Cellular decompositions and Chebyshev interpolants for real algebraic curves

Silviana Amethyst[*] Jonathan D. Hauenstein[†] Charles W. Wampler[‡]

February 10, 2022

Abstract

A cellular decomposition of a real algebraic curve consists of a collection of vertices and edges which have a smooth interior. A numerical cellular decomposition represents an edge via one interior point and a homotopy that permits tracking along the edge. This homotopy yields a parameterization of the edge that can be used for performing various operations such as membership testing, computing winding numbers at each boundary vertex, and generating sample points. By combining numerical cellular decomposition with Chebyshev interpolants along each edge, we develop an approach that can efficiently perform computations such as optimizing an analytic or nonanalytic function over real algebraic curves. Examples utilizing `Bertini_real` to compute a numerical cellular decomposition and `Chebfun` to compute Chebyshev interpolants are provided.

1 Introduction

A common problem arising in various scientific and engineering fields is to perform non-algebraic computations on real algebraic curves, that is, one may wish to intersect a one-real-dimensional solution set of a system of polynomial equations with a non-algebraic set or find extrema of a function evaluated on the set. For example, in an engineering design problem, one might have a set of polynomial equality constraints that leave a one-dimensional real set of designs meeting basic requirements. After computing a description of that set, the engineer may search for the best design in the set by posing a fitness function to optimize. Often the fitness function is not algebraic, and even so, we would like assurance that we will find the global optimum. For a particular example, in Section 6.2 we consider a mechanism design problem wherein a one-dimensional algebraic curve of degree 72 in 10 variables describes all four-bars meeting specified motion requirements, and a subsequent optimization step finds all four-bars on this curve that minimize the sum of the leg lengths as in (12).

[*]Department of Mathematics, University of Wisconsin-Eau Claire, Eau Claire, WI 54701 (`amethyst@uwec.edu`, `silviana.org`).

[†]Department of Applied and Computational Mathematics and Statistics, University of Notre Dame, Notre Dame, IN 46556 (`hauenstein@nd.edu`, `www.nd.edu/~jhauenst`).

[‡]General Motors R&D, MC 480-106-RA2, 30500 Mound Road, Warren, MI 48092 (`charles.w.wampler@gm.com`, `www.nd.edu/~cwample1/`).

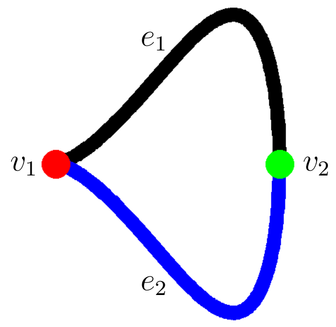

Figure 1: A cellular decomposition of the real algebraic curve defined by $x^6 - x^3 + y^2 = 0$

The approach of this paper is to couple two mathematical ideas to solve this problem: numerical cellular decomposition of the curve (implicit representation) together with Chebyshev interpolants of each edge (explicit representation) in the decomposition. This allows for computations on the real algebraic curve to be performed efficiently using the interpolants.

A cellular decomposition is a description of the curve as a finite union of vertices and edges that have a smooth interior. For example, Figure 1 illustrates a cellular decomposition for the real algebraic curve defined by $x^6 - x^3 + y^2 = 0$ consisting of two vertices, v_1 and v_2, and two edges, e_1 and e_2. A numerical cellular decomposition builds on methods from numerical algebraic geometry, e.g., see [2, 14], to describe an edge via one interior point together with a homotopy-based approach for tracking along the edge starting from this interior point. Such a decomposition was first described in [13] (see also [3, 5]) and is implemented in `Bertini_real` [4]. By utilizing a homotopy to describe each edge, this numerical description provides an implicit parametric representation of each edge which can be useful for deciding membership and sampling points on each edge.

Some computations, such as plotting and optimization, can be performed more efficiently on each edge given an explicit parametric representation. For simplicity, we always consider representing a bounded part of a real algebraic curve. We show how to compute an analytic reparameterization of each edge by using a numerical cellular decomposition to compute winding numbers at each boundary vertex. The analytic reparameterization permits efficient approximation of each edge using Chebyshev interpolants. The `Matlab` package `Chebfun` [9] includes out-of-the-box methods for performing computations using Chebyshev interpolants, including plotting and optimization.

The structure of this paper is as follows. Section 2 describes using numerical algebraic geometry to represent a complex algebraic curves and a cellular decomposition to represent a real algebraic curve. Sections 3 and 4 describe numerical aspects involving a numerical cellular decomposition (implicit representation) and Chebyshev interpolant (explicit representation), respectively. Section 5 provides a short description our proof-of-concept software implementation combining `Bertini_real` and `Chebfun` which is then demonstrated on some examples in Section 6, including kinematic synthesis of a four-bar mechanism.

2 Witness sets and cellular decompositions

Since computing a cellular decomposition of a real algebraic curve performs computations over the complex numbers, we first start with representing a complex curve using witness sets in numerical algebraic geometry. More details about witness sets are provided in [2, 14]. This is followed by a description of a cellular decomposition.

2.1 Witness sets

Consider a system of n polynomial equations having complex coefficients in N variables:

$$f(x) = f(x_1, \ldots, x_N) = \begin{bmatrix} f_1(x_1, \ldots, x_N) \\ \vdots \\ f_n(x_1, \ldots, x_N) \end{bmatrix} = 0.$$

The solution set, namely $\mathcal{V}(f) = \{x \in \mathbb{C}^N \mid f(x) = 0\}$, is called a *complex algebraic set* or simply a *variety.* A variety $V \subset \mathbb{C}^N$ is *irreducible* if, whenever $V_1, V_2 \subset \mathbb{C}^N$ are varieties such that $V = V_1 \cup V_2$, then either $V = V_1$ or $V = V_2$. If a variety is not irreducible, it is *reducible.*

For an irreducible variety $V \subset \mathbb{C}^N$, the *dimension* of V, denoted $\dim V$, is equal to the minimum of the dimension of the tangent space at any point on the variety. A point $p \in V$ is called a *smooth* point if the dimension of the tangent space of V at p is equal to $\dim V$. If $p \in V$ is not a smooth point, it is called a *singular point.* For irreducible varieties, the set of smooth points is path connected. If every point is smooth, the variety is a *manifold.*

Every variety V has an *irreducible decomposition* consisting of finitely many irreducible components, namely

$$V = \bigcup_{i=1}^{k} V_i$$

where each V_i is irreducible and $V_i \not\subset V_j$ for $i \neq j$. An irreducible decomposition is unique up to relabeling. Then, $\dim V = \max_i \dim V_i$ and, for $j = 0, \ldots, \dim V$, let

$$Z_j = \bigcup_{\dim V_i = j} V_i.$$

The variety Z_j is called the *pure j-dimensional component* of V.

Our focus is on varieties $C \subset \mathbb{C}^N$ which are pure 1-dimensional, called *complex algebraic curves* or simply *curves.* A curve is represented in numerical algebraic geometry (e.g., see [2, 14]) by a *witness set* which is a triple $\{f, \mathcal{H}, W\}$ constructed as follows. By intersecting C with a *hyperplane*, which is a variety defined by a single linear polynomial, say, $\ell(x)$ with $\mathcal{H} = \mathcal{V}(\ell)$, one considers the resulting set $C \cap \mathcal{H}$. The *degree* of C, denoted $\deg C$, is the maximum number of points in $C \cap \mathcal{H}$ such that $\mathcal{H}$ is a hyperplane and $\dim(C \cap \mathcal{H}) = 0$. A hyperplane $\mathcal{H}$ is called *general* with respect to C if $\deg C = \#(C \cap \mathcal{H})$. With this setup, the three elements in the witness set for C are:

- f is a polynomial system, called a *witness system*, such that each irreducible component of C is an irreducible component of $\mathcal{V}(f)$;

- $\mathcal{H}$ is a general hyperplane, called a *witness slice*; and
- $W = C \cap \mathcal{H}$, called a *witness point set*, which consists of $\deg C$ points.

Since one can always handle the reducible case by considering each irreducible component separately, we assume that C is irreducible. Then, each witness point $w \in W$ for C is a smooth point, i.e., the tangent space of C at w is one dimensional. Let $Jf(w)$ be the Jacobian matrix of f evaluated at w. Trivially, we know $\dim \operatorname{null} Jf(w) \geq 1$. If $\dim \operatorname{null} Jf(w) = 1$, then C is said to be *generically reduced* or, equivalently, has *multiplicity 1*, with respect to f. Otherwise, C is *generically nonreduced* or, equivalently, has *multiplicity* > 1, with respect to f. One can always assume that C is generically reduced with respect f by replacing f with, for example, an isosingular deflation [10] of f.

Finally, since f has n polynomials, we trivially know that $n \geq N-1$. We will always assume that $n = N-1$, i.e., a well-constrained system, which can be accomplished by replacing f with a randomization $R \cdot f$ where $R \in \mathbb{C}^{(N-1)\times n}$ is general. The system $R \cdot f$ is also a witness system due to Bertini's theorem, e.g., see [14, Thm. A.8.7].

Example 1. The *twisted cubic curve* $C = \{(t, t^2, t^3) \mid t \in \mathbb{C}\} \subset \mathbb{C}^3$ is irreducible with $\deg C = 3$. The following polynomial systems are three examples of witness systems for C:

$$f = \begin{bmatrix} y - x^2 \\ z - xy \\ xz - y^2 \end{bmatrix}, \quad g = \begin{bmatrix} y - x^2 + 3(xz - y^2) \\ z - xy - 2(xz - y^2) \end{bmatrix}, \quad h = \begin{bmatrix} (y - x^2)(z - xy) \\ xz - y^2 \end{bmatrix}.$$

The curve C is generically reduced with respect to f and g, but generically nonreduced with respect to h. Moreover, $\mathcal{V}(f) = C$ while $\mathcal{V}(g)$ and $\mathcal{V}(h)$ contain other irreducible components. For $\mathcal{H} = \mathcal{V}(3x - 2y + z - 4)$, $W = C \cap \mathcal{H}$ consists of 3 points. To 4 decimal places, we have

$$W = \{(1.6506, 2.7246, 4.4973), (0.1747 \pm 1.5469i, -2.3623 \pm 0.5404i, -1.2486 \mp 3.5597i)\}$$

where $i = \sqrt{-1}$. Therefore, the witness set $\{g, \mathcal{H}, W\}$ satisfies the above assumptions since g consists of 2 polynomials in 3 variables such that C is generically reduced with respect to g.

2.2 Cellular decomposition

Suppose that $C \subset \mathbb{C}^N$ is a complex algebraic curve and consider the real points of C, namely $C \cap \mathbb{R}^N$. The set $C \cap \mathbb{R}^N$ is called a *real algebraic curve* or simply a *real curve*. That is, a real curve is simply the real points of a complex algebraic curve which consists of at most finitely many isolated real points and at most finitely many one-dimensional real arcs. This suggests that a cellular decomposition of a real curve must have two ingredients:

- a *vertex* which is simply a point in $\mathbb{R}^N$; and
- an *edge* which is subset of $\mathbb{R}^N$ that is diffeomorphic to an open interval in $\mathbb{R}$.

Hence, an edge is a *one-dimensional real manifold* whose endpoints are vertices. In particular, the set of vertices V must contain the set of singular points of C that are real, which includes the isolated real points in $C\cap\mathbb{R}^N$. With this setup, a *cellular decomposition* of $C\cap\mathbb{R}^N$ is a collection of vertices $V=\{v_1,\ldots,v_k\}$ and edges $E=\{e_1,\ldots,e_\ell\}$ such that

$$C\cap\mathbb{R}^N=\bigsqcup_{i=1}^{k} v_i \sqcup \bigsqcup_{j=1}^{\ell} e_j.$$

Example 2. A cellular decomposition for the real part of the twisted cubic curve C in Ex. 1 consists of a single edge since $C\cap\mathbb{R}^3=\{(t,t^2,t^3)\mid t\in\mathbb{R}\}$ is a one-dimensional real manifold that is diffeomorphic to $\mathbb{R}$.

Example 3. Consider the real part of the curve $C=\mathcal{V}(x^6-x^3+y^2)$ from [7, Ex. 5], which is shown in Figure 1. A cellular decomposition of this real curve consists of the following:

- vertices $v_1=(0,0)$ and $v_2=(1,0)$;
- edges $e_1=\left\{\left(x,\sqrt{x^3-x^6}\right)\;\middle|\; x\in(0,1)\right\}$ and $e_2=\left\{\left(x,-\sqrt{x^3-x^6}\right)\;\middle|\; x\in(0,1)\right\}$.

3 Numerical cellular decomposition and analytic reparameterization

Since symbolic expressions for the edges such as those listed in Ex. 3 are typically not available due to the Abel–Ruffini theorem and Galois theory, e.g., see [8, 15], a numerical description of an edge, creating a *numerical cellular decomposition*, was developed in [5, 13] where each edge is represented by an interior point and a homotopy that allows one to track from the interior point along the edge. Also, as demonstrated in Ex. 3, edges need not be analytic at the endpoints, which are vertices, analytic reparameterization is utilized so that an edge can be represented near each endpoint using a power series expansion.

3.1 Numerical cellular decomposition

Following the setup in Section 2, let $\{f,\mathcal{H},W\}$ be a witness set for a curve $C\subset\mathbb{C}^N$ such that f consists of $N-1$ polynomials and C is generically reduced with respect to f. The approach in [5, 13] uses this given information to construct a numerical cellular decomposition for a real curve $C\cap\mathbb{R}^N$ using the following process consisting of 4 steps followed by 2 optional post-processing steps, which is computed with respect to a linear projection map $\pi:\mathbb{R}^N\to\mathbb{R}$ such that $C\cap\pi^{-1}(t)$ consists of finitely many points for every $t\in\mathbb{C}$. This property holds for a general linear projection map.

1. **Compute critical points**
   Compute the finitely many points $x\in C\cap\mathbb{R}^N$ such that the Jacobian matrix of $\{f(x),\pi(x)\}$ is rank deficient.
2. **Bound infinite behavior**
   To replace infinite-length edges with edges of finite length, compute the intersection points of $C\cap\mathbb{R}^N$ with a sphere that encompasses all of the critical points from Step 1. Append the resulting intersection points to the list of critical points.

3. **Slice between critical points**
   Since the topological behavior of $\pi^{-1}(v) \cap C \cap \mathbb{R}^N$ can only change at a critical value $v = \pi(c)$ where c is a critical point, compute $\pi^{-1}(t) \cap C \cap \mathbb{R}^N$ for values of t between successive critical values yielding interior points on the edges.

4. **Connect the dots**
   Using the homotopy
   $$H(x,t) = \left[\begin{array}{c} f(x) \\ \pi(x) - t \end{array} \right] = 0, \tag{1}$$
   starting at the interior points computed in Step 3, one can vary t to attempt to connect previously computed points.

5. **Merge (optional)**
   To reduce the number of edges, one can merge edges which connect at smooth points.

6. **Smooth (optional)**
   One can provide a smooth sampling of each edge by collecting points obtained by tracking along the edge using the homotopy (1).

In the end, a numerical cellular decomposition consists of a finite collection of vertices $V \subset C \cap \mathbb{R}^N$ containing the real isolated points of $C \cap \mathbb{R}^N$ and the endpoints of the edges. Each edge is represented numerically by three points: a “left” vertex $v_\ell \in V$, a “right” vertex $v_r \in V$, and an interior point $p \in C \cap \mathbb{R}^N$. Hence, if $\pi_\ell = \pi(v_\ell)$ and $\pi_r = \pi(v_r)$, then $\pi_\ell < \pi(p) < \pi_r$ and one can track along the edge via the homotopy (1) starting at $(x,t) = (p, \pi(p))$ and letting t vary in the interval (π_ℓ, π_r).

Example 4. To illustrate computing a numerical cellular decomposition, consider the real plane curve defined by

$$\begin{aligned} 0 = {} & 16x_1^8x_2^4 + 64x_1^7x_2^5 + 96x_1^6x_2^6 + 64x_1^5x_2^7 + 16x_1^4x_2^8 + 64x_1^{10} \\ & + 320x_1^9x_2 + 584x_1^8x_2^2 + 416x_1^7x_2^3 - 128x_1^6x_2^4 - 496x_1^5x_2^5 \\ & - 128x_1^4x_2^6 + 416x_1^3x_2^7 + 584x_1^2x_2^8 + 320x_1x_2^9 + 64x_2^{10} - 399x_1^8 \\ & - 1596x_1^7x_2 - 934x_1^6x_2^2 + 2784x_1^5x_2^3 + 4643x_1^4x_2^4 + 2784x_1^3x_2^5 \\ & - 934x_1^2x_2^6 - 1596x_1x_2^7 - 399x_2^8 + 840x_1^6 + 2520x_1^5x_2 - 772x_1^4x_2^2 \\ & - 5744x_1^3x_2^3 - 772x_1^2x_2^4 + 2520x_1x_2^5 + 840x_2^6 - 766x_1^4 - 1532x_1^3x_2 \\ & - 2298x_1^2x_2^2 - 1532x_1x_2^3 - 766x_2^4 + 288x_1^2 + 288x_1x_2 + 288x_2^2 - 27 \end{aligned}$$

which arises as the discriminant [11, Fig. 9] of the Kuramoto model [12] with three coupled oscillators. An illustration of computing a numerical cellular decomposition of this curve is provided in Figure 2 where each edge is depicted with its own arbitrary color. The vertices in the numerical cellular decomposition are indicated by circular dots and the interior points for the edges are indicated with diamonds.

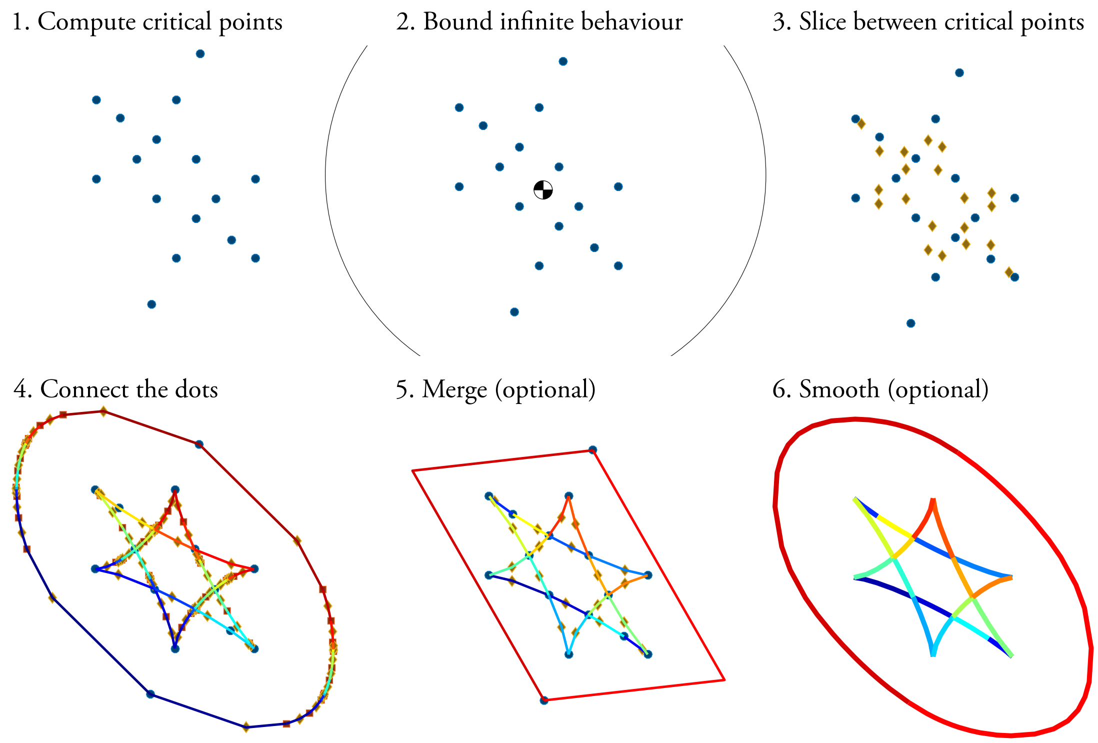
1. Compute critical points
2. Bound infinite behaviour
3. Slice between critical points
4. Connect the dots
5. Merge (optional)
6. Smooth (optional)

Figure 2: Pictorial representation of computing a numerical cellular decomposition in four steps. Merging (Step 5) simplifies the computed decomposition, which is particularly useful when performing additional computations. Smoothing (Step 6) is for visualization purposes.

Example 5. The cellular decomposition of the curve $C = \mathcal{V}(x^6 - x^3 + y^2)$ in Ex. 3 presented in Figure 1 is based on the projection $\pi(x, y) = x$. Hence, a numerical cellular decomposition with homotopy

$$H(x, y, t) = \left[\begin{array}{c} x^6 - x^3 + y^2 \\ x - t \end{array}\right] = 0$$

consists of

- vertices $v_1 = (0, 0)$ and $v_2 = (1, 0)$;
- edges e_1 with "left" vertex $v_\ell = v_1 = (0, 0)$, "right" vertex $v_r = v_2 = (1, 0)$, and interior point $p = (1/2, \sqrt{7}/8)$, and e_2 with "left" vertex $v_\ell = v_1 = (0, 0)$, "right" vertex $v_r = v_2 = (1, 0)$, and interior point $p = (1/2, -\sqrt{7}/8)$.

3.2 Analytic reparameterization

Viewing each edge as the union of two arcs, one from the interior point p to the "left" vertex v_ℓ and the other from the interior point p to the "right" vertex v_r, each arc can be reparameterized to be analytic at the endpoints and diffeomorphic to the interval $[-1, 1]$. Considering the "left" arc, let $\pi_\ell = \pi(v_\ell)$ and $\pi_p = \pi(p)$. Thus, the first reparameterization is to replace $t \in (\pi_\ell, \pi_p]$ in (1) by

$$t(\tau) = \tau \cdot \pi_p + (1-\tau) \cdot \pi_\ell = \pi_\ell + \tau \cdot (\pi_p - \pi_\ell)$$

where $\tau \in (0,1]$. Therefore, the "left" arc corresponds to $x(\tau)$ for $\tau \in (0,1]$ satisfying $x(1) = p$ and $H(x(\tau), t(\tau)) \equiv 0$ where H as in (1). By local uniformization [14, § 10.2], there exists a positive integer c_ℓ, called the *cycle number* or *winding number* such that, for $\tau(s) = s^{c_\ell}$, the "left" arc $x(s) := x(\tau(s))$ is analytic for $s \in [0,1]$. The cycle number c_ℓ can be computed using singular endgames [14, § 10.2] which are typically already employed in Step 4 of Section 3.1 when connecting the dots.

Finally, a linear transformation is used to map from $[0,1]$ to $[-1,1]$ via $\mu(s) = 2 \cdot s - 1$ so that $s = (\mu+1)/2$. In the end, one has that $x(\mu)$ for $\mu \in [-1,1]$ satisfying $x(1) = p$ and

$$H_\ell(x,\mu) = \left[\begin{array}{c} f(x) \\ \pi(x) - \left(\pi_\ell + \left(\frac{\mu+1}{2}\right)^{c_\ell} \cdot (\pi_p - \pi_\ell)\right) \end{array} \right] = 0. \qquad (2)$$

One can apply a similar construction to the "right" arc yielding a cycle number c_r, potentially different than c_ℓ, with corresponding homotopy

$$H_r(x,\mu) = \left[\begin{array}{c} f(x) \\ \pi(x) - \left(\pi_r + \left(\frac{\mu+1}{2}\right)^{c_r} \cdot (\pi_p - \pi_r)\right) \end{array} \right] = 0 \qquad (3)$$

where $\pi_r = \pi(v_r)$.

Example 6. Reconsidering the setup from Ex. 5, all cycle numbers for both arcs of each edge are 2. In particular, for the "left" arc of edge e_1, $\pi_\ell = 0$, $\pi_p = 1/2$, and $c_\ell = 2$ yielding

$$H_\ell(x,\mu) = \left[\begin{array}{c} x^6 - x^3 + y^2 \\ x - (\mu+1)^2/8 \end{array} \right] = 0.$$

With $(x(1), y(1)) = (1/2, \sqrt{7}/8)$, one has

$$(x(\mu), y(\mu)) = \left(\frac{(\mu+1)^2}{8}, \frac{(\mu+1)^3}{512} \sqrt{512 - (\mu+1)^6} \right)$$

which is clearly analytic for $\mu \in [-1,1]$. Similarly, for the "right" arc of edge e_1, $\pi_r = 1$, $\pi_p = 1/2$, and $c_r = 2$ yielding

$$H_r(x,s) = \left[\begin{array}{c} x^6 - x^3 + y^2 \\ x - (1 - (\mu+1)^2/8) \end{array} \right] = 0.$$

With $(x(1), y(1)) = (1/2, \sqrt{7}/8)$, one has

$$(x(\mu), y(\mu)) = \\ \left(1 - \frac{(\mu+1)^2}{8}, \frac{(\mu+1)(7-2\mu-\mu^2)}{512} \sqrt{(7-2\mu-\mu^2)(\mu^4 + 4\mu^3 - 18\mu^2 - 44\mu + 169)} \right)$$

which is clearly analytic for $\mu \in [-1,1]$.

4 Chebyshev interpolants

The outcome of Section 3 is a collection of implicitly defined arcs $x(\mu)$ for $\mu \in [-1,1]$ which are analytic. Hence, one is able to perform computations on $x(\mu)$ by path tracking using the corresponding homotopy (2) or (3). Rather than perform computations implicitly using a homotopy, another approach for performing additional computations on each arc is to construct an explicit representation in the form of a Chebyshev interpolant approximating the arc. In particular, one can construct a Chebyshev interpolant for each coordinate $x_i(\mu)$ of $x(\mu)$ which then easily facilitates many efficient computations on the arc, such as plotting and optimization as implemented in the `Matlab` package `Chebfun` [9].

For $N \geq 1$, the *Chebyshev points of the second kind* are

$$\mu_j = -\cos(j \cdot \pi/N) \in (-1,1) \quad \text{for} \quad j = 0,\ldots,N. \tag{4}$$

Given a function $f(\mu)$, let $f_j = f(\mu_j)$. Then, there is a unique polynomial $p_N(x)$, called the *Chebyshev interpolant*, with $\deg p_N \leq N$ such that $p(x_j) = f_j$ for $j = 0,\ldots,N$. By selecting an appropriate value of N, one is able to control the error

$$\|p_N - f\|_\infty = \max_{-1\leq\mu\leq 1} |p_N(\mu) - f(\mu)|$$

thereby creating an explicit polynomial approximation of the function for $\mu \in [-1,1]$. In `Chebfun`, the value of N is selected for the error to be roughly machine precision.

Applied to the problem of approximating $x(\mu)$, each coordinate $x_i(\mu)$ is approximated using its own Chebyshev interpolant so the value of N utilized can be different for each coordinate. As stated above, each $x(\mu_j)$ is computed via path tracking using the corresponding homotopy (2) or (3).

5 Implementation details

The software `Bertini_real` [4] implements the numerical cellular decomposition as described in [3, 5, 13]. The cycle numbers, as described in Section 3.2, are naturally computed and recorded as part of this decomposition. The `Matlab` package `Chebfun` [9] implements the computation of Chebyshev interpolants along with a whole host of operations one can perform on them, including optimization and visualization. The advantage of combining `Bertini_real` and `Chebfun` is to allow fast and efficient computations over real curves in any ambient dimension as exemplified in Section 6.

Since a Chebyshev interpolant is constructed for each coordinate for the "left" and "right" arcs for each edge, some implementation improvements are available. First, we note that this was described theoretically for simplicity of presentation. However, one is not compelled to do this. Using the least common multiple of the cycle numbers for the "left" and "right" arcs, a common regularization can be used.

Second, since `Chebfub` uses deterministic sampling based on the Chebyshev points (4), we can prevent repeated evaluation at the same point on the curve by using a hashtable to provide a cache. Fortunately, `Matlab` provides a suitable container and the hash function is merely the value in $[-1,1]$.

Third, since the computation of points on the arcs using the homotopies (2) and (3) is merely a parameter homotopy, one can perform the path tracking in parallel, offering significant speedup, via `Paramotopy` [1].

Fourth, adjusting the tolerances used in `Chebfun` for deciding when a Chebyshev interpolant is sufficiently accurate can drastically reduce the number of points which are needed to be computed along the arcs. Since computing points on the arcs requires path tracking, reducing the number of paths to be tracked can yield significant computational savings.

Fifth, all of the computations on the edges are local. For example, if all of the points on an edge are known to not be physically meaningful for the particular problem, that edge can trivially be ignored providing computational savings.

Finally, it is often the case that one wants to perform many computations over the same real algebraic curve. Once the computational expense of obtaining the Chebyshev interpolants of the arcs has been performed, this decomposition can be reused to perform many efficient computations over the same real algebraic curve thereby amortizing the one-time cost of computing the Chebyshev interpolants over many computations.

6 Applications

The underlying motivation of using Chebyshev interpolants to approximate real algebraic curves is to quickly and efficiently perform not necessarily algebraic computations on the real algebraic curves, which is exemplified by optimizing a non-algebraic objective function over real algebraic curves. This is demonstrated on two plane curves in Section 6.1 followed by synthesizing a four-bar linkage in Section 6.2.

6.1 Plane curves

The first plane curve under consideration is the "asteroid" curve defined by

$$f(x, y) = (x^2 + y^2 - 1)^3 + 27x^2y^2 = 0$$

whose real part is depicted in Figure 3. This sextic curve has four singular points at $(\pm 1, 0)$ and $(0, \pm 1)$ which are connected by four edges. Consider minimizing the objective function

$$\Omega(x, y) = \sin(7x) + \sin(5y)$$

over the real part of the "asteroid" curve. Since both x and y as well as trigonometric functions of x and y are utilized, this is not an algebraic problem.

Once Chebyshev interpolants are computed for each arc, the minimization computation is performed with ease resulting in six local minima as shown in Figure 3.

As a more challenging example, consider the cubic "alpha" curve defined by

$$f(x, y) = y^2 - x^2(x + 1) = 0$$

whose real part is depicted in Figure 4. This real curve has one singular point at the origin and is unbounded so Step 2 in Section 3.1 is used to bound the infinite behavior.

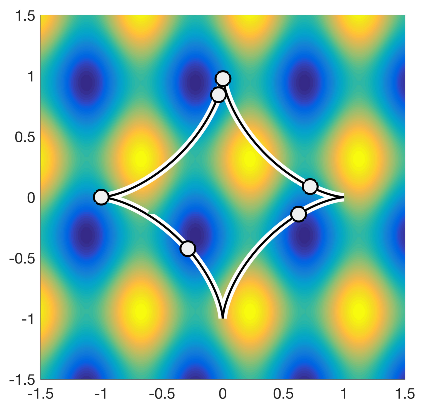

Figure 3: Visualization of minimizing $\Omega(x, y) = \sin(7x) + \sin(5y)$ over the "asteroid" curve $(x^2 + y^2 - 1)^3 + 27x^2y^2 = 0$ including the six local minima.

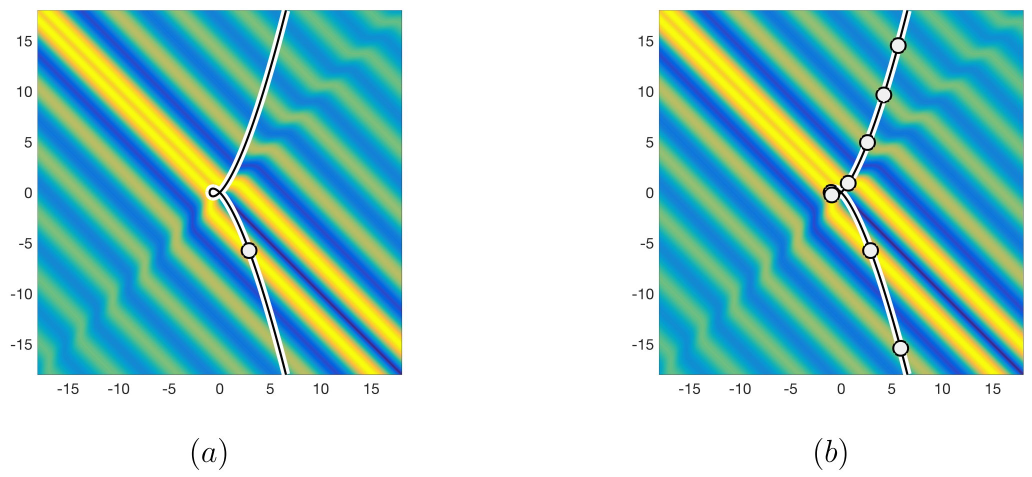

Figure 4: Visualization of maximizing $\Omega(x, y) = \text{besselj}(1, \text{abs}(x + y) + \text{erfc}(x - y) - 1)$ over the "alpha" curve $y^2 - x^2(x + 1) = 0$. (a) Global maximum. (b) Several local maxima.

On the real points of the "alpha" curve, consider maximizing

$$\Omega(x, y) = \text{besselj}(1, \text{abs}(x + y) + \text{erfc}(x - y) - 1)$$

written using `Matlab` notation. In particular, besselj() is the Bessel function of the first kind, abs() is the absolute value, and erfc() is complementary error function. Clearly, $\Omega(x, y)$ is non-analytic and defined in terms of integrals. Nonetheless, by using Chebyshev interpolants of the arcs of the "alpha" curve, numerical optimization can be easily performed yielding the global maximum as shown in Figure 4(a) and several local maxima as shown in Figure 4(b).

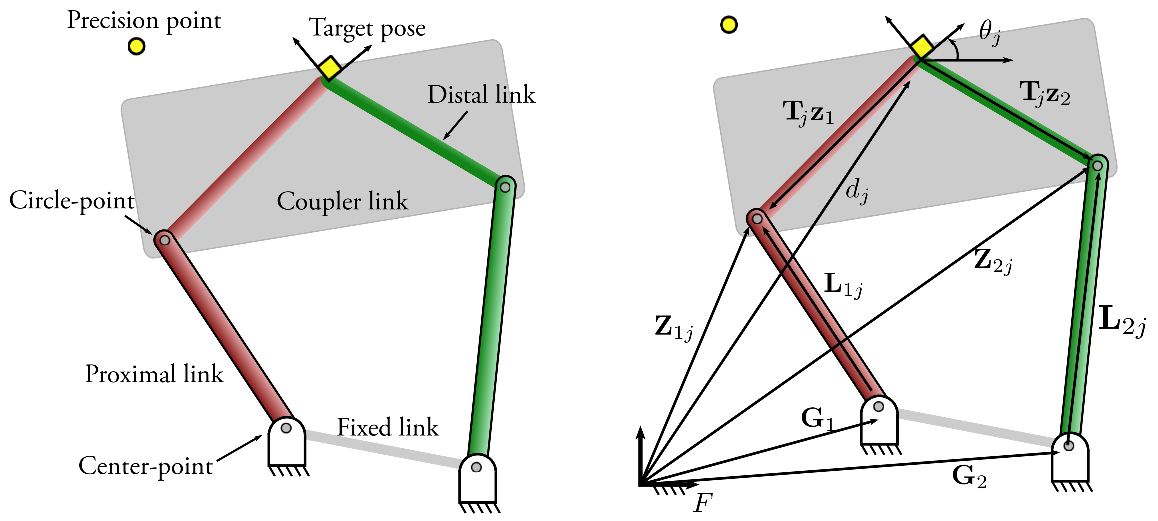

Figure 5: A four-bar planar mechanism, terminology on the left and with vectors as the links on the right. The subscript j ranges over the precision points and poses.

6.2 Planar four-bar mechanisms

The final example arises from synthesizing four-bar mechanisms as depecited in Figure 5. A four-bar mechanism is composed of two dyads (commonly referred to as an RR linkage) shown in red and green in Figure 5. The base of each *proximal link* is fixed in space at a *center-point* with a revolute joint. The two center-points connected by a virtual *fixed link*. A *distal link* is joined to the proximal link at a *circle-point* with a revolute joint. The distal links are connected rigidly to form the *coupler link* with a marked *coupler-point* at the intersection of the distal links. Hence, each circle-point traces a (subset of a) circle around its respective center-point as the four-bar is driven and the coupler-point traces the *coupler curve*.

One can synthesize four-bar mechanisms by enforcing constraints based on *precision points* and *precision poses*. A precision point is a point that the coupler curve must pass through. A precision pose is a precision point together with a pose that specifies how the orientation of the coupler link as it passes through the point. In particular, precision points provide a single constraint while precision poses provide two constraints. Assuming rigid links, synthesizing four-bar mechanisms amounts to solving a polynomial system. Depending on the number of precision points and precision poses specified, the dimension and degree of the solution set varies [6].

One approach to formulating the polynomial system is by utilizing isotropic coordinates $(x, \bar{x}) \in \mathbb{C}^2$ following [6]. In isotropic coordinates, the point $(x, \bar{x})$ corresponds with a real point if x and $\bar{x}$ are complex conjugates of each other. Let $i = \sqrt{-1}$. An angle θ_j is converted into isotropic coordinates $(\mathbf{T}_j, \overline{\mathbf{T}}_j)$ where

$$\mathbf{T}_j = \cos\theta_j + i\sin\theta_j = e^{i\theta_j} \quad \text{and} \quad \overline{\mathbf{T}}_j = \cos\theta_j - i\sin\theta_j = e^{-i\theta_j}$$

In particular, $\mathbf{T}_j \cdot \overline{\mathbf{T}}_j = 1$. When j corresponds with a precision pose, $\mathbf{T}_j$ and $\overline{\mathbf{T}}_j$ are known since the angle θ_j is prescribed. When j corresponds with a precision point, $\mathbf{T}_j$ and $\overline{\mathbf{T}}_j$ become variables with the constraint $\mathbf{T}_j \cdot \overline{\mathbf{T}}_j = 1$. A loop equation involving the coupler-point is

$$\mathbf{Z}_{1j} = \mathbf{T}_j \mathbf{z}_1 + d_j \tag{5}$$
$$\overline{\mathbf{Z}}_{1j} = \overline{\mathbf{T}}_j \overline{\mathbf{z}}_1 + \bar{d}_j \tag{6}$$

while loop equations for the dyads are

$$\mathbf{L}_{1j} = \mathbf{T}_j \mathbf{z}_1 + d_j - \mathbf{G}_1, \tag{7}$$
$$\overline{\mathbf{L}}_{1j} = \overline{\mathbf{T}}_j \overline{\mathbf{z}}_1 + \bar{d}_j - \overline{\mathbf{G}}_1, \tag{8}$$
$$\mathbf{L}_{2j} = \mathbf{T}_j \mathbf{z}_2 + d_j - \mathbf{G}_2, \tag{9}$$
$$\overline{\mathbf{L}}_{2j} = \overline{\mathbf{T}}_j \overline{\mathbf{z}}_2 + \bar{d}_j - \overline{\mathbf{G}}_2. \tag{10}$$

These equations require that the proximal link remains constant in length:

$$f_j = \mathbf{L}_{1j}\overline{\mathbf{L}}_{1j} - \mathbf{L}_{11}\overline{\mathbf{L}}_{11} = 0 \qquad \text{for each precision point and precision pose}$$
$$g_j = \mathbf{L}_{2j}\overline{\mathbf{L}}_{2j} - \mathbf{L}_{21}\overline{\mathbf{L}}_{21} = 0 \qquad \text{for each precision point and precision pose}$$

while the variables $(\mathbf{T}_j, \overline{\mathbf{T}}_j)$ for each precision point satisfy

$$h_j = \mathbf{T}_j\overline{\mathbf{T}}_j - 1 = 0 \qquad \text{for each precision point.}$$

It often happens that constraining a synthesis problem so that there are only finitely many solutions is unproductive since many of the solutions turn out to be nonphysical (such as links having non-real lengths), useless (such as a linkage which visits the precision points in an unintended order or on two different branches of the coupler curve), or impractical (such as a linkage having poor force transmission so that it is susceptible to jamming or having links too long to fit within geometric constraints). In such circumstances, a useful tactic is to remove a constraint, thereby allowing the designer to search over a real algebraic curve to find a desirable answer. One way to automate such a search is for the designer to specify an optimization criterion and it is often the case that the criterion is not algebraic.

We will demonstrate this by synthesizing a four-bar linkage based on specifying one precision point and four poses summarized in Table 1. This setup produced a curve of degree 72 in the 10 variables

$$(\mathbf{G}_1, \overline{\mathbf{G}}_1, \mathbf{z}_1, \overline{\mathbf{z}}_1, \mathbf{G}_2, \overline{\mathbf{G}}_2, \mathbf{z}_2, \overline{\mathbf{z}}_2, \mathbf{T}_3, \overline{\mathbf{T}}_3)$$

for which the real part is readily decomposed by `Bertini_real`. Note that there is a trivial relabeling of the two dyads yielding a two-way correspondence

$$(\mathbf{G}_1, \overline{\mathbf{G}}_1, \mathbf{z}_1, \overline{\mathbf{z}}_1, \mathbf{G}_2, \overline{\mathbf{G}}_2, \mathbf{z}_2, \overline{\mathbf{z}}_2, \mathbf{T}_3, \overline{\mathbf{T}}_3) \longleftrightarrow (\mathbf{G}_2, \overline{\mathbf{G}}_2, \mathbf{z}_2, \overline{\mathbf{z}}_2, \mathbf{G}_1, \overline{\mathbf{G}}_1, \mathbf{z}_1, \overline{\mathbf{z}}_1, \mathbf{T}_3, \overline{\mathbf{T}}_3). \tag{11}$$

In our particular run for this paper, the numerical cellular decomposition of the curve resulted in 1247 edges (this number is not an invariant as it depends on the random projection utilized). Figure 6 shows a projection of the real points of this curve onto $\mathbf{G}_1$.

Aiming to find an optimal four-bar mechanism based on the specifications in Table 1, we consider minimizing the sum of the leg lengths:

$$\Omega = |L_{11}| + |L_{21}| + |z_1| + |z_2|. \tag{12}$$

Table 1: Task specification with four precision poses and one precision point

j	d_{j_x}	d_{j_y}	θ_j (degrees)
1	0.79	0.87	-11.76
2	1.02	0.53	-33.56
3	0.90	1.00	–
4	1.28	0.33	-35.06
5	1.30	0.71	-8.14

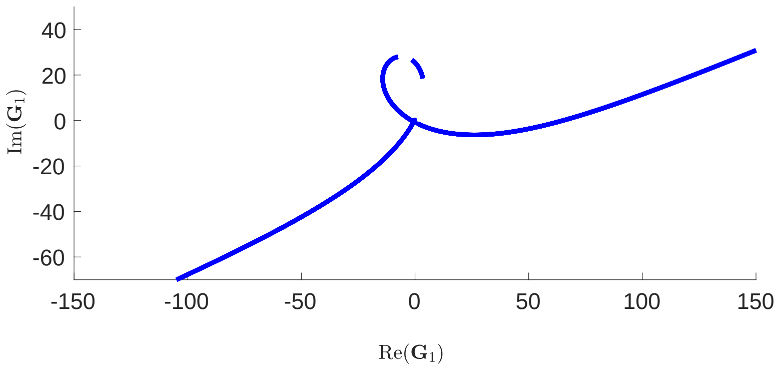

Figure 6: Projection of a one-dimensional family of four-bar mechanisms onto $\mathbf{G}_1$.

After computing Chebyshev interpolants for the arcs, `Chebfun` determined there were 28 local minima over the real algebraic curve restricted to the sphere centered at the origin of radius $2.744 \cdot 10^3$. Due to the symmetry highlighted in (11), the 28 local minima arise in 14 pairs. Of these, 9 yield nondegenerate four-bar mechanisms while the other 5 are degenerate. In Figure 7, parts (a) and (b) contains coupler curves for two distinct nondenerate local minima; in part (a), the other branch of the coupler curve is out of the viewport and has neither precision poses nor points on it. Part (c) of the figure is an example of a coupler curve of a mechanism which has a branch defect since some precision points and precision poses lie on both branches.

7 Conclusion

We have shown how numerical cellular decomposition and Chebyshev interpolants can be combined to represent real algebraic curves in a form where subsequent operations, such as optimizing an analytic or nonanalytic function over the curve, can be carried out efficiently to high precision. Winding numbers computed during cell decomposition allow desingularization so that arcs can be represented to machine precision even as they approach singularities. In our implementation, after using `Bertini_real` to compute a numerical cellular decomposition and `Chebfun` to compute a Chebyshev interpolant for each arc, all the functionality that `Chebfun` provides for functions on a real interval become available on the entirety of a real algebraic curve. We have illustrated this capability on several examples, including a mechanical design problem involving four-bar linkages.

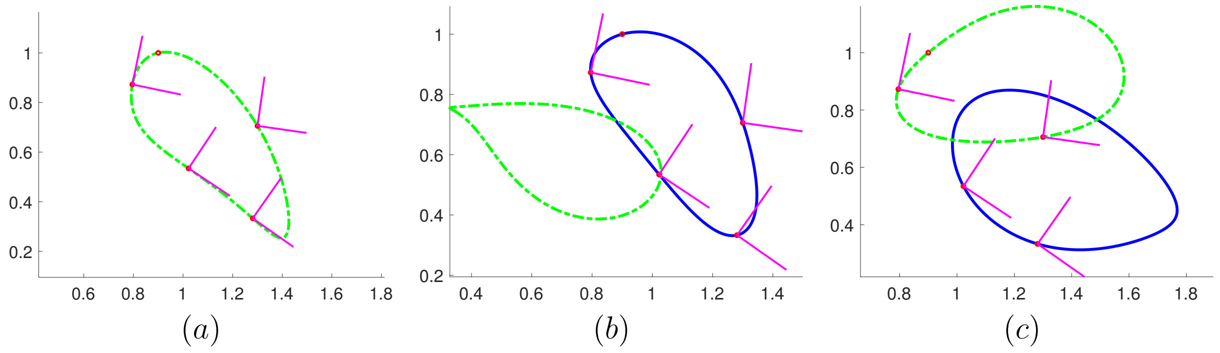

Figure 7: Coupler curves for three local minima of the sum of leg lengths of four-bar mechanisms satisfying the conditions of Table 1. (a),(b) Coupler curves of two nondegenerate four-bar mechanisms which are mechanically desirable. (c) Coupler curve of four-bar mechanism which has a branch defect.

Acknowledgments

The authors thank Nick Trefethen for helpful discussions regarding `Chebfun`. Support for this research was provided in part by NSF CCF 1812746.

References

[1] D. Bates, D. Brake, and M. Niemerg. Paramotopy: Parameter homotopies in parallel. In J. H. Davenport, M. Kauers, G. Labahn, and J. Urban, editors, *Mathematical Software – ICMS 2018*, pages 28–35, Cham, 2018. Springer International Publishing.

[2] D. J. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler. *Numerically solving polynomial systems with Bertini*, volume 25 of *Software, Environments, and Tools*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2013.

[3] G. Besana, S. Di Rocco, J. Hauenstein, A. Sommese, and C. Wampler. Cell decomposition of almost smooth real algebraic surfaces. *Numerical Algorithms*, 63(4):645–678, 2013.

[4] D. A. Brake, D. J. Bates, W. Hao, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler. Bertini_real: Software for real algebraic sets. Available at bertinireal.com.

[5] D. A. Brake, D. J. Bates, W. Hao, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler. Algorithm 976: Bertini_real: Numerical decomposition of real algebraic curves and surfaces. *ACM Transactions on Mathematical Software (TOMS)*, 44(1):10, 2017.

[6] D. A. Brake, J. D. Hauenstein, A. P. Murray, D. H. Myszka, and C. W. Wampler. The complete solution of alt–burmester synthesis problems for four-bar linkages. *Journal of Mechanisms and Robotics*, 8(4):041018, 2016.

[7] D. A. Brake, J. D. Hauenstein, and C. Vinzant. Computing complex and real tropical curves using monodromy. *Journal of Pure and Applied Algebra*, 223(12):5232–5250, 2019.

[8] D. A. Cox. *Galois theory.* Pure and Applied Mathematics (Hoboken). John Wiley & Sons, Inc., Hoboken, NJ, second edition, 2012.

[9] T. A. Driscoll, N. Hale, and L. N. Trefethen. Chebfun guide. 2014.

[10] J. D. Hauenstein and C. W. Wampler. Isosingular sets and deflation. *Found. Comput. Math.*, 13(3):371–403, 2013.

[11] I. Hiskens and R. Davy. Exploring the power flow solution space boundary. *IEEE Transactions on Power Systems*, 16(3):389–395, 2001.

[12] Y. Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. *Lect. Notes Phys.*, 39:420–422, 1975.

[13] Y. Lu, D. Bates, A. Sommese, and C. Wampler. Finding all real points of a complex curve. *Contemporary Mathematics*, 448:183–205, 2007.

[14] A. J. Sommese and C. W. Wampler, II. *The numerical solution of systems of polynomials arising in engineering and science.* World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2005.

[15] J.-P. Tignol. *Galois' theory of algebraic equations.* World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, second edition, 2016.