Recognition of computationally constructed loci

Peter Lebmeir and Jürgen Richter-Gebert

Technical University of Munich, Department of Mathematics,
Boltzmannstr. 3, 85748 Garching, Germany

lebmeir@ma.tum.de,
WWW home page (German):
http://www-m10.ma.tum.de/bin/view/Lehrstuhl/PeterLebmeir

richter@ma.tum.de,
WWW home page (German):
http://www-m10.ma.tum.de/bin/view/Lehrstuhl/RichterGebert

Abstract. We propose an algorithm for automated recognition of computationally constructed curves and discuss several aspects of the recognition problem. Recognizing loci means determining a single implicit polynomial equation and geometric invariants, characterizing an algebraic curve which is given by a discrete set of sample points. Starting with these discrete samples arising from a geometric ruler and compass construction an eigenvalue analysis of a matrix derived from the data leads to proposed curve parameters. Utilizing the construction itself, with its free and dependent geometric elements, further specifications of the type of constructed curves under generitcity assumptions are made. This is done by a second eigenvalue analysis of parameters of several generically generated curves.

1 Introduction

The generation of loci is one of the central applications in todays computer geometry programs. In abstract terms a *locus* consists of all locations that a specific point of a geometric configuration can take, while one parameter of the configuration may vary. For instance the locus of all points that are at a fixed distance from a fixed point is simply a circle. Typically the locus data generated by a geometry program does not consist of a symbolic description of the locus. Rather than that a more or less dense collection of sample points on the locus is generated. In many cases the user of such a program can identify the underlying curve visually by simply looking at it or by a pre-knowledge of the underlying construction or by a combination of both. However, for several applications (like automatic assistance, geometric expert systems, etc.) it is highly desirable to recognize these curves automatically. After a geometric construction for a locus is specified such a recognition procedure can be carried out on two different levels. First, one is interested in which locus is generated for a *concrete* instance (i.e. position of free elements) of the construction. Second, one is interested in

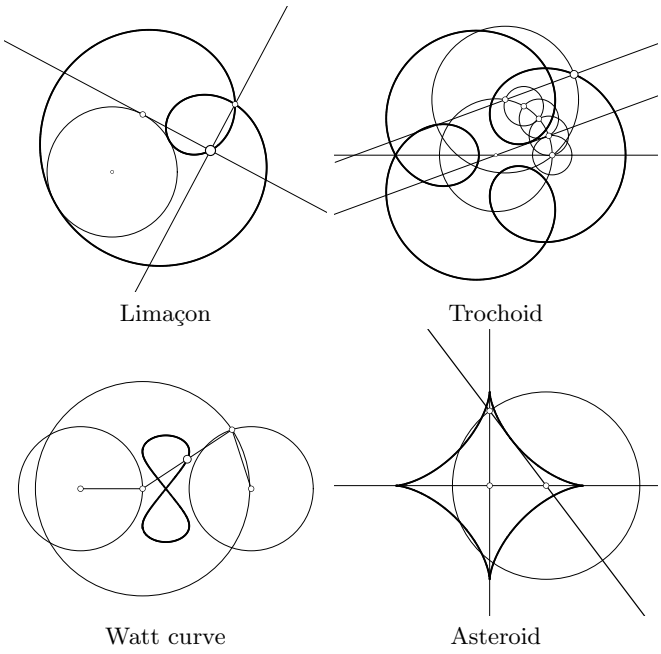

Fig. 1. geometric constructions with of loci. Each curve can be interpreted as zero set of an algebraic equation.

invariants of *all* the loci that can be generated by a specific construction. We refer to this set as the *type* of the locus. Very often in the literature *types of loci* (which are algebraic curves in our setup) are closely related to *names* of the corresponding curves like *cardioide, limaçon, lemniscate, Watt curve, conic,* etc. One of the ultimate goals of a locus recognition algorithm would be an output like:

> "You constructed a limaçon with parameters $r = 0.54$ and $s = 0.72$. A limaçon is given by the equation $(x^2 + y^2 - 2rx)^2 = s^2(x^2 + y^2)$. Your construction will generically generate a limaçon."

If one treats the geometric construction as a black box which takes positions of the free elements of a construction as input and produces sample points of the generated locus as output, the above demands imply the necessity of a curve recognition algorithm based on discrete sets of sample points.

In our constructions we only allow primitive operations coming from ruler and compass operations. In this case a locus turns out to be contained in a zero set of a polynomial equation for any specific instance of a construction. Fixing the free construction elements in a way that only one degree of freedom is left, a dependent point is restricted to an algebraic curve. The locus data is given by a discrete set of sample points $\mathcal{P}$ on this curve, constructed for example via

a dynamic geometry program. In general, these programs trace a single point and thus return a set $\mathcal{P}$ contained in a single real branch of the zero set of an algebraic equation. Therefore we presume $\mathcal{P}$ to be of that kind.

In our case, curve recognition for a specific instance of a construction means extracting a single algebraic *equation* of minimal degree from a discrete point set $\mathcal{P}$ and determining its degree. In a second step we focus on the set $\mathcal{B}$ of curves obtainable by *all* instances of a specific construction – the *type* of the curve. We presume that all coordinates of dependent elements in a construction are analytic functions in the free parameters of the free elements of a construction. Under this assumption, an examination of the degree of the curves in $\mathcal{B}$ and geometric invariant extraction is possible. In this way we can, at least for simple curves, associate a *name* with $\mathcal{B}$, characterizing the type of the contained curves. In particular the minimal degree itself turns out to be an invariant property.

Recognizing implicit multivariate polynomials has been investigated also by other research groups most often in different contexts and with different side constraints. A genuine feature of the setup treated in this article is that the sample points usually come with a high arithmetic precision and that it is known in advance that they belong to an algebraic curve with an a priori upper degree bound (usually the real degree will be much lower than this bound). This is in contrast to related research work in computer vision or patter recognition where one is interested to approximate camera pictures of geometric shapes by algebraic curves in order to recognize these shapes. There the shape data does not a priori belong to an algebraic curve and one is only interested in relatively roughly approximating curves of fixed degree (compare for instance [6],[9][10])

Another related setup was treated and implemented in the dynamic geometry program Cabri Géomètre [3]. There a first locus recognition algorithm was implemented. Unfortunately no algorithmic details are available and practical experiments show that the algorithm used there is relatively instable in particular under rigid transformations. (We will deal with this particular issue later in Section 5.2).

In this article we introduce a concept of locus recognition, that deals with the construction on two different levels. On both levels the construction itself is treated as a black box that generates sample points on the locus for specific parameters of the free elements of the construction. In the first step we propose an algorithm that takes the sample points of one locus as sole input data and reconstructs the parameters and degree of the underlying algebraic curve for this specific instance. In a second step we allow the free construction elements to vary and thus produce a whole collection of sample point sets $\mathcal{P}_1, \mathcal{P}_2, \ldots$ each of which represents locus data of the same type. Randomization techniques are then used to finally extract common features of these loci and determine the type of the locus (with high probability).

2 Computationally constructed curves

Dynamic geometry programs facilitate ruler and compass constructions in a plane: Elementary construction steps consist of placing free points in the plane, joining two points by a line, constructing circles by midpoint and perimeter and intersecting lines with lines, lines with circles or circles with circles. Furthermore, points can be restricted to already constructed elements like a line or a circle. We will call such points *semi-free* as they have only one degree of freedom. Lines (circles) are zero sets of linear (quadratic) equations and can be computationally represented by the parameters of these equations. Semi-free points can be described by a one-parametric solution manifold of an equation. All-in-all (for details see [1]) any instance of a geometric construction corresponds to the solution set of a finite set of polynomial equations in the parameters generated by the free and semi-free elements. Allowing only finitely many ruler and compass construction steps, every position of a dependent point can be described by algebraic equations depending on the position of the free points.

While moving a free point in a dynamic geometry program, one can watch the dependent points move consistently with the construction. Fixing all (semi-) free elements except for one semi-free point in a construction, a single dependent point p gets restricted to a zero set of an algebraic equation. Tracing p with a dynamic geometry program under the movement of a single semi-free point (a *mover*), a single branch B of an algebraic curve is revealed. The algebraic curve can be described by a polynomial equation $b(p) = 0$ of minimal degree. B is the set of locations of the traced point p. Unless otherwise stated all points and curves are assumed to be given in homogeneous coordinates. Under these assumptions a constructed branch B is contained in the zero set $Z(b)$ of a homogeneous polynomial function $b : \mathbb{RP}^2 \to \mathbb{R}$ of minimal degree d_b:

$$Z(b) = \left\{ \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \in \mathbb{RP}^2 \;\middle|\; b(x_1, x_2, x_3) = \sum_{i+j+k=d_b} \beta_{i,j,k} \cdot x_1^i x_2^j x_3^k = 0 \right\} \quad (1)$$

From a construction in a dynamic geometry program we get a discrete set $\mathcal{P}$ containing points lying almost exactly on a branch B of some $Z(b)$. Using algebraic methods, we could in principle determine a corresponding curve of very high degree by the following approach: In a first step each elementary construction step is described as a polynomial relation in the coordinates of geometric elements. In a second step elimination techniques (for instance based on Gröbner Bases or Ritt's algebraic decomposition method) are used to find an implicit representation of the possible position of the locus generating point. These type of approach to curve recognition is purely symbolic but it significantly suffers from combinatorial explosion of the algebraic structures. Even for small constructions one could in general not hope for results in reasonable computation time. In contrast we use the construction as a "black box" and consider only the sample point data of the computationally constructed branch B. Our aim is to derive its describing algebraic equation b of minimal degree. Now our driving questions are:

1. Which plane algebraic curve of minimal degree is "reasonably" fitting the numerical locus data? (Which b fits $\mathcal{P}$ "reasonably" and what is d_b?)
2. Regarding a ruler and compass construction, which type of plane algebraic curves are constructed? (What is the degree d_b of almost all curves B in $\mathcal{B}$ and what name characterizes $\mathcal{B}$?)

Clearly the problem needs some mathematic modeling which preserves quality criteria. The next section will deal with this issue. Additionally the first part of our work, i.e. the parameter extraction, can be applied to any curve or even surface recognition problem given sufficiently many points of sufficiently high precession.

3 Curve recognition

Given a finite set of data points $\mathcal{P} = \{p_1, p_2, ..., p_m\}$, e.g. a set of samples in homogeneous coordinates on a locus from a ruler and compass construction. The problem of fitting an algebraic curve B to the data set $\mathcal{P}$ is usually cast as minimizing the mean square distance

$$\frac{1}{m}\sum_{i=1}^{m} dist(p_i, Z(b))^2 \tag{2}$$

from the data points to the curve (see [5]). This is a function in the coefficients β of the polynomial b, where $dist(p, Z(b))$ denotes the Euclidean distance from a point p to the zero set $Z(b)$.

Unfortunately, there is in general no closed form for $dist$. In principle $dist$ could be approximated by iterative calculations. However, they turn out to be expensive and very clumsy for the needed optimization procedure. Therefore we will use other approximation. Without any numerical noise $b(p) = 0$ for all $p \in Z(b)$ the first choice to replace (2) is

$$\frac{1}{m}\sum_{i=1}^{m} b(p_i)^2. \tag{3}$$

We will use this formula as central ansatz for modeling the distance of curve and sample points. This is adequate since our sample points are given with high arithmetic precision.

We now will describe how the coefficents of b that minimize (3) can be calculated. For this let $T_1(x), T_2(x), ..., T_k(x)$ be a basis of the linear space of all homogeneous polynomials of degree d in three variables. If $d = 2$ one could take, for instance, the monomial basis

$$T_1(x) = x_1^2, T_2(x) = x_1x_2, T_3(x) = x_1x_3, T_4(x) = x_2^2, T_5(x) = x_2x_3, T_6(x) = x_3^2.$$

In general we have $k = \frac{1}{2}(d+1)(d+2)$ basis polynomials. Furthermore let

$$\tau_d = (T_1, T_2, \ldots, T_k) : \mathbb{RP}^2 \to \mathbb{R}^k$$

be the function that associates a point $p \in \mathbb{RP}^2$ with its evaluation $\tau_d(p) \in \mathbb{R}^k$ of all the basis functions of degree d.

Assuming we have a discrete set of n points $\mathcal{P} = \{p_1, \ldots, p_n\}$ from an underlying construction and assuming we know d_b, the degree of the computationally constructed branch B, we can fix τ_d for polynomials of degree d_b. Furthermore, we can form an $n \times k$ matrix $P := P_{d,\mathcal{P}}$ with row vectors $\tau_d(p_i)$ of τ_d-transformed points $p_i \in \mathcal{P}$. Thus minimizing (3) is equivalent to minimizing $\|P\beta\|_2$, with the unknown parameter vector β. So far the minimum can easily be obtained by setting $\beta = (0, \ldots, 0)$. This comes from the homogenicity of the problem setup and the fact that β and $\lambda \cdot \beta$ for $\lambda \in \mathbb{R}\backslash\{0\}$ define the same curve. We overcome this problem by forcing additional side-constraints on β and require $\|\beta\|_2 = 1$. In relation to our ansatz (3) this side constraint has two advantages: First of all it is geometrically reasonable, since it ensures a bound on each coefficient of the polynomial. Second, it leads to a nicely solvable minimization problem. All in all curve recognition in our case is stated as

$$\min_{\|\beta\|_2=1} \|P\beta\|_2 \tag{4}$$

The above minimization problem leads to a nicely structured Eigenvector-Eigenvalue analysis as the following considerations show. We have

$$\min_{\|\beta\|_2=1} \|P\beta\|_2 = \min_{\|\beta\|_2=1} \sqrt{\beta^T P^T P \beta},$$

which is minimized by an eigenvector of $P^T P$ corresponding to an eigenvalue λ of minimal absolute value. The minimum itself is this very eigenvalue λ. In terms of a singular value decomposition of P, λ is no more than a singular value of P of minimal absolute value and (4) is minimized by a corresponding right singular vector of P.

Proof: $P^T P$ is symmetric and features an orthogonal diagonalization employing eigenvectors. Thus $\exists Q \in \mathcal{O}(k) : \; Q^T P^T P Q = \Lambda$ with a diagonal matrix $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_k)$ exhibiting the eigenvalues λ_i $(1 \leq i \leq k)$ of $P^T P$. The j-th column of Q is the eigenvector corresponding to the j-th diagonal entry of Λ. Abbreviating $Q^T \beta$ by γ, we get:

$$\min_{\|\beta\|_2=1} \|P\beta\|_2 = \min_{\|\gamma\|_2=1} \sqrt{\gamma^T \Lambda \gamma}.$$

Let λ_j be an eigenvalue of minimal absolute value $|\lambda_j| = \min\{|\lambda_j| \mid 1 \leq j \leq k\}$ and let $e_j^T = (0, ..., 0, 1, 0, ..., 0)$ be the j-th canonical basis vector. Then

$$\min_{\|\gamma\|_2=1} \sqrt{\gamma^T \Lambda \gamma} = \sqrt{e_j^T \Lambda e_j} = \sqrt{\lambda_j}.$$

By undoing the transformation from β to $\gamma = e_j$, we get $\beta^T = (q_{1,j}, q_{2,j}, ..., q_{k,j})$, the j-th eigenvector of $P^T P$. If the minimal absolute eigenvalue is a single eigenvalue, then we get a unique eigenvector minimizing eigenvector β. If not, then

β may be any vector in the span of all eigenvectors to the eigenvalues with an absolute value of λ_j. Since a singular value of P is the root of an eigenvalue of P^TP, the right singular vector corresponds to the eigenvectors from above. $\square$

In practice with large $\mathcal{P}$ the minimal absolute eigenvalue of P^TP will always be unique. In cases, where $\mathcal{P}$ does not specify a singe curve and a hole bunch of curves could have $\mathcal{P}$ as sample point set, we can not expect an algorithm to always return the correct curve.

Let us assume having the sample points on a branch B of a yet unknown degree d_b in a no-numerical-noise environment. If we choose an arbitrary testing degree $d \in \mathbb{N}\backslash\{0\}$ and examine the behavior of P and (4), we have the following cases:

$d = d_b$: As we investigate in homogeneous polynomials, the length of the parameter vector β is $k_d = \frac{1}{2}(d+1)(d+2)$, but the corresponding curve is determined by $k_d - 1$ points in general position. Per assumption, $\mathcal{P}$ contains this many points in general position and thus $rank(P) = k_d - 1 = rank(P^TP)$ and P^TP has a single vanishing eigenvalue $\lambda = 0$. The corresponding eigenvector is the desired parameter vector of the curve.

$d < d_b$: An analysis of the corresponding P yields: P is of maximal rank and P^TP has no vanishing eigenvalue.

$d > d_b$: Let $d = d_b + r$ for some $r \in \mathbb{N}$. An analogous observation shows that P^TP has $k_r = \frac{1}{2}(r+1)(r+2)$ vanishing eigenvalues. This is a reasonable behavior because it shows that the resulting curves are degenerate: One component is the curve of degree d_b fixed by $\mathcal{P}$ and the other component is a more or less random curve of degree r.

This shows that for a given $\mathcal{P}$ the product P^TP has a vanishing eigenvalue $\lambda = 0$ if and only if the testing degree is greater than or equals to d_b. Thus for constructed curves, we can compute the eigenvalues λ of P^TP of minimal absolute value for testing degrees $d = 1$, $d = 2$, ..., successively. This leads to a point where λ is zero. At that point we reached the desired degree.

In practice, where the sample points are disturbed by a small numerical noise, we will not observe vanishing eigenvalues but a significant drop (usually a few orders of magnitude) of the absolute minimal eigenvalues (see section 5). Thus we are able to say:

"You constructed a curve of degree d_b."

4 Type of constructed loci

Let $\mathcal{B}$ be the whole continuum of constructed branches B and let $d_{\mathcal{B}}$ denote the maximum of all degrees d_b of constructed curves B contained in $\mathcal{B}$. Since we deal with finite constructions, $d_{\mathcal{B}}$ is bounded. $\mathcal{P}$ is a discrete set of points originating from discrete positions of a semi-free mover of a construction. In the last section we have seen that P^TP becoming (nearly) singular indicates that the degree of a curve is equal to or below a certain value. Due to the analyticity

(omitting the details here[1]) almost any curve contained in $\mathcal{B}$ is of degree $d_{\mathcal{B}}$. Therefore we can select a generic parameter set in the parameter space of a specific construction and get a generic curve B. The degree d_b of B equals $d_{\mathcal{B}}$ with probability one. This means, we can detect $d_{\mathcal{B}}$ and prompt the user of a dynamic geometry program:

> "You constructed a curve of degree d_b. Your construction will generically generate a curve of degree $d_{\mathcal{B}}$."

To further specify $\mathcal{B}$ with associated degree $d_{\mathcal{B}}$, we can calculate invariants with respect to either projective, affine or Euclidean transformations. If all curves that belong to a given construction show the same invariants stored in a database, a name like circles, conchoids or limaçon can be associated with $\mathcal{B}$.

One possibility to achieve this is to examine the orthogonal space $\mathcal{B}^{\perp}$. This is the space of all $k_{d_{\mathcal{B}}} = \frac{1}{2}(d_{\mathcal{B}}+2)(d_{\mathcal{B}}+1)$ dimensional vectors orthogonal to any coefficient vector of any curve in $\mathcal{B}$. Therefore $\mathcal{B}^{\perp}$ is an invariant of the underlying construction. Generating a matrix V containing at least $k_{d_{\mathcal{B}}}$ parameter vectors to generic curves of $\mathcal{B}$, $\mathcal{B}^{\perp}$ is the null space of V. It can be determined by an eigenvalue/eigenvector analysis: $\mathcal{B}^{\perp}$ is spanned by the eigenvectors of V^TV to eigenvalues equal to zero.

The remaining task for a graduation of $\mathcal{B}$ would be to correlate the calculated orthogonal space with a database and retrieve a corresponding name. Then we can say something like:

> "Your construction will generically generate a circle."

The idea of how to avoid costly calculations of $\mathcal{B}^{\perp}$ is to simply look up some orthogonal spaces for curve classes of degree $d_{\mathcal{B}}$ in a database and perform a multiplication with V. Getting a zero result we know $\mathcal{B}$ to be a subset of all curves with the orthogonal space taken.

As a simple example let $d_{\mathcal{B}} = 2$ and let $\mathcal{B}$ be a set of circles. We assume that all coefficient calculations are performed with the monomial basis for quadrics (see example after equation (3)). Looking in a database for orthogonal spaces to curve types of degree two, we will find *circles* with an associated orthogonal space $\mathcal{B}^{\perp} = span((1,0,0,-1,0,0),(0,1,0,0,0,0)) =: span(o_1, o_2)$, because circles are characterized by the following properties properties

1. the coefficient of x^2 equals the coefficient of y^2:
   $(1,0,0,-1,0,0)\cdot\tau_2(x,y,z) = x^2 - y^2 = 0$, $\forall(x,y,z)$ on a circle.
2. the coefficient of xy equals zero:
   $(0,1,0,0,0,0)\cdot\tau_2(x,y,z) = xy = 0$, $\forall(x,y,z)$ on a circle.

If in fact all members of $\mathcal{B}$ are circles, $V\cdot o_1$ and $V\cdot o_2$ are zero or almost zero due to numerical effects.

To specify a more complex $\mathcal{B}$, a linear test like matrix multiplication may not suffice. It has to be tested wether potentially non-linear expressions hold for

[1] see [1] for an in-depth analysis

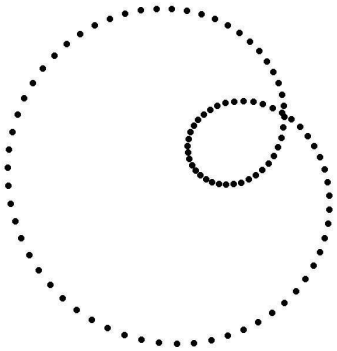

Fig. 2. Raw data of a degree-four-curve

$k_{d_{\mathcal{B}}}$ generic curves out of $\mathcal{B}$. Additional difficulties arise when $\mathcal{B}$ is tested to be a subset of a parametrized class, like the class of conchoids[2]. Here, research is in progress to unify and simplify the tests of type affiliations. Focusing on Euclidean graduation, a promising approach seems to be to calculate an intrinsic rotation invariant center of a curve using all curve parameters as introduced in [7]. Then mapping $\mathcal{B}$ to a class of curves in a database by transformations is reduced to comparing invariants under rotations and scaling.

5 Experimental results

Constructing a curve C in a dynamic geometry program and applying a curve recognition algorithm based on minimizing (4) yields some curve parameters. This corresponding curve should fit the sample data if the corresponding eigenvalue is significantly small. We present some data of numerical experiments with loci whose sample points have been calculated with the program Cinderella [4].

5.1 Finding the degree

We start with the example of a limaçon corresponding to the first picture in Figure 1. The corresponding sample data $\mathcal{P}$ of the locus corresponds to the cloud of points given in Figure 2. The numerical precision of the data is approximately 14 digits. The Euclidean coordinates of points $p \in \mathcal{P}$ range from -10 to 10. Figure 3. shows a sequence of plots for the estimated curves of degree $d = 2, \ldots, 6$. The smallest absolute values of eigenvalues λ_d in these five situations are given below the pictures. The sample points are given for reference. One observes a significant drop of the eigenvalues from $d = 3$ to $d = 4$. The

[2] A conchoid symmetric to the x-axis and with the singularity in the Euclidean origin can be written as $b(x, y, z) = (x - \sigma)^2(x^2 + y^2) - \rho^2 x^2 = 0$ for some parameters σ, ρ.

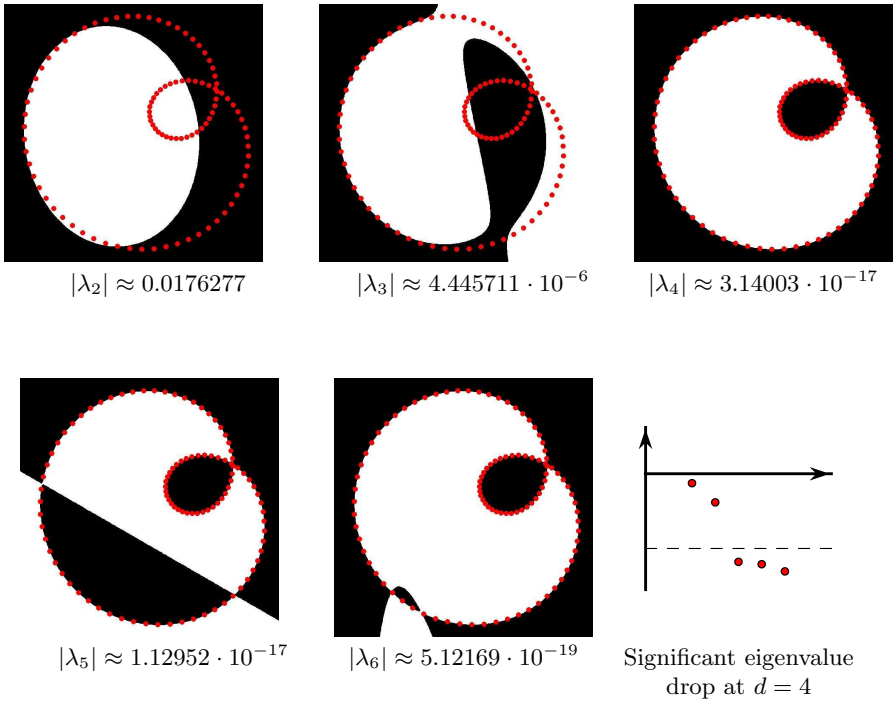

Fig. 3. Estimated curves for degrees 2, 3, 4, 5 and 6 with logarithmic plot of the minimal eigenvalues against the tested degrees and threshold-line corresponding to $\lambda = 10^{-14}$

picture in the bottom right shows the logarithm of the minimal eigenvalues with a threshold-line at $\ln(\lambda) = \ln(10^{-14})$ marked vertically against the tested degree $d \in \{2, 3, 4, 5, 6\}$. Moreover for $d = 5$ we obtain (as expected) three absolute eigenvalues below $\lambda = 10^{-14}$ and for $d = 6$ we obtain six absolute eigenvalues below 10^{-14}. The next larger absolute eigenvalues are around 10^{-6}. This suggests that the curve under investigation is of degree $d = 4$. For the degree five approximation the plot unveils an additional component of degree one. And for degree six an additional component of degree two shows up.

As a second example we examine a parametrizable Epicycloid

$$f(t) = \begin{pmatrix} r \cdot \cos(k \cdot t) - s \cdot \cos(l \cdot t) \\ -r \cdot \sin(k \cdot t) - s \cdot \sin(l \cdot t) \end{pmatrix} \quad . \tag{5}$$

If we choose $r = 2.4$, $s = 3$, $k = 2$ and $l = 5$ we get samples depending on the range and sampling rate for t. Three instances of sample point sets are shown in Figure 4. In the top row $t \in [0; 2\pi)$, in the middle row $t \in [0; 40\pi)$ and in the bottom row $t \in [0, 10\pi)$. In any case 72 equidistant values are used. Thus we have an ordered sequence of sample points given by our parametrization. In dynamic geometry programs ordered samples can be calculated even if the

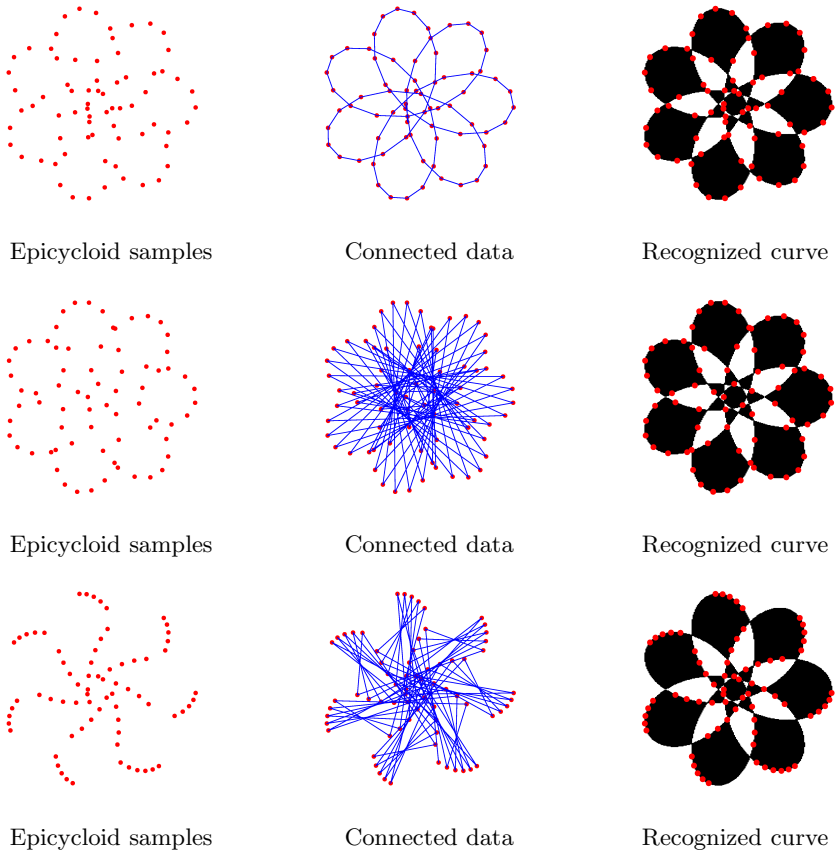

Fig. 4. Reconition of Epicycloids (5)

constructed curve is not rationally parametrizable[3]. A good idea of what the curve belonging to the data looks like can be provided by connecting the ordered samples linearly. The middle column of Figure 4 shows these line segments. In the case of $t \in [0; 40\pi)$ in the second row, the sampling rate is too low to show the curve itself but the picture gives a good impression of the contour. In case where the data is not scattered along the curve but locally concentrated, more intuition is needed when looking at the connected data. In either case our algorithm detects the correct curve of degree ten (ignoring roundoff errors).

As a further example, we take a Epicycloid of degree six with parameters $r = 1$, $s = 6$, $k = 12$ and $l = 4$. The top left image of Figure 5 shows it,

[3] More precisely a curve can be parameterized by rational functions if it comes from a construction that uses exclusively ruler constructions. Constructions use ruler *and* compass may lead to more general (still algebraic curves).

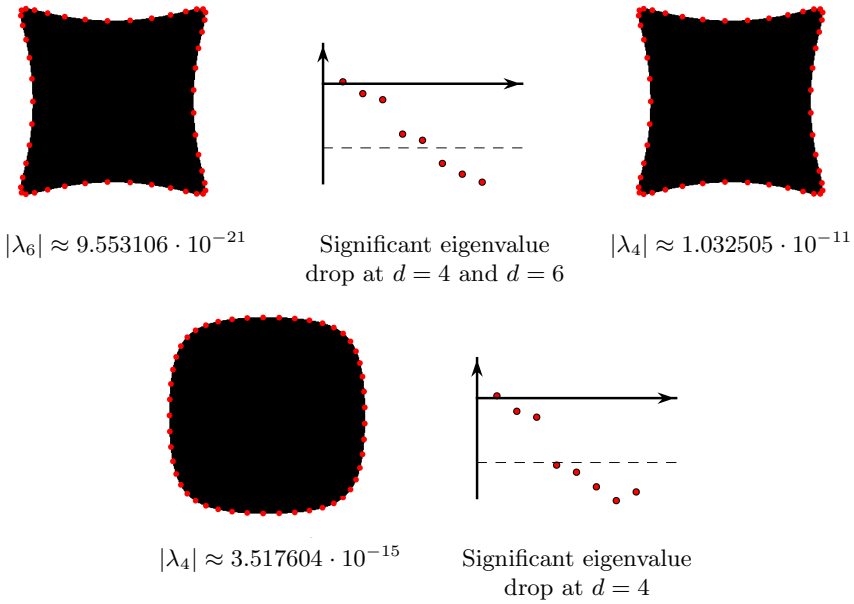

Fig. 5. Reconition of Epicycloids (5)

correctly recognized by our algorithm. The corresponding minimal eigenvalue is sufficiently small, i.e. below the dashed threshold. The logarithmic plot of the minimal eigenvalues against the tested degree in the subsequent middle picture reveals another significant eigenvalue-drop. It suggests that the curve looks quite like a curve of degree four. There is in fact such a curve of degree $d = 4$, approximating the sample points very good. It can be seen in the rightmost picture in the top row of Figure 5. We could have guessed this only by looking at the parameter vector of the correctly recognized curve: All parameters corresponding to the $x_1^i x_2^j x_3^k$-terms with $i + j \in \{5, 6\}$ are very small or vanish totally (x_3 is the coordinate for homogenization). This curve of degree four can be discarded because of a (second) significant drop in the minimal eigenvalues when switching to a testing degree of six. An acceptable threshold, e.g. 10^{-14}, may be chosen to tell these curves apart. By altering the Epicycloid's parameter r to $r = 0.4$, (5) still provides us with a Epicycloid of degree six. In this case there is only one significant drop in the minimal eigenvalues. λ_4 is already in the range of roundoff errors. Thus our presented algorithm falsely assumes that the data belongs to a curve of degree four instead of six.

Our algorithm makes a false degree guess more often for curves of relatively high degree. The may be recognized as curves of lower degree. Usually in most of these cases the approximation by the low degree curve is so good that it visually fits the sample data extremely well. The situation is qualitatively the same as

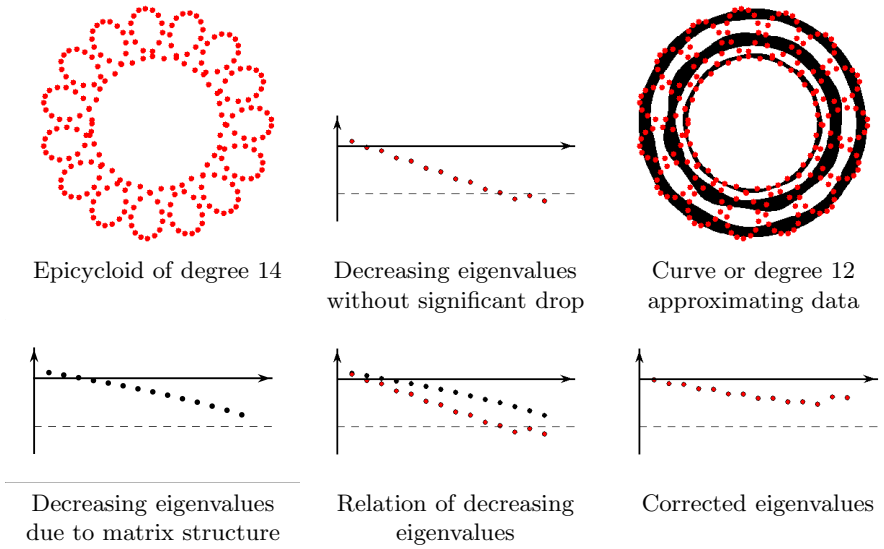

Fig. 6. Problematic reconition of an Epicycloid of degree 14 and structural decreasing eigenvalues

with the Epicycloid in Figure 5. For the example in the top row of the sample data is already very nicely approximated by an algebraic curve of degree 4.

In general one can observe that the absolute value of the smaller eigenvalue becomes smaller the higher the degree gets. Let us take for example the Epicycloid with a degree of 28 from Figure 6. It has the parameters $r = 1.4$, $s = 5$, $k = 14$ and $l = 1$ and it comes up with a decreasing minimal eigenvalue the higher the selected tested degree is. This is not only the case with Epicycloids but with any curve. The minimal eigenvalues stop decreasing once they reach the region of roundoff errors. A reason for this behavior is in the structure of the matrix P. P is used to determine the degree in the minimization (4). It is built up rowwise by $\tau_d(p)$ with p being a sample point. Thus the columns of P consist of values $x^i y^j z^k$, where x, y and z are the coordinates of a sample point. Since the number of free parameters grows quadratically with the degree d curves of higher degree can simply approximates a set of sample points much better. We did not analyze this effect qualitatively but experimental results exploit a roughly exponential behavior. For this we we took a sample set of 300 random points within the range of -4 and 4 for x and y. With probability one these points will not be contained in a curve of degree 20 or lower. We can do this a hundred times and thereby calculate the mean minimal eigenvalues for our testing degrees. The resulting diminishing eigenvalues due to our matrix structure is shown in a logarithmic plot in the bottom left of Figure 6. In this logarithmic scale it is almost linear. The next picture (middle of lower row) compares this generic effect with the behavior for the Epicycloid. We see that at least part of

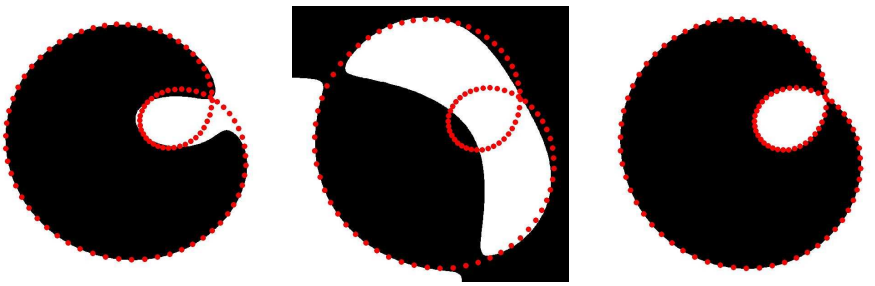

Fig. 7. Experiments with shifted sample data for shifts $(10, 10)$, $(30, 30)$ and a shift $(30, 30)$ followed by scaling

the systematic eigenvalue fall can be explained by this generic effect. Knowing the structural diminution approximately, we can introduce a correction term in our calculations. The bottom right plot of Figure 6 reflects the outcome. This means that in case of our Epicycloid we have small eigenvalues for high testing degrees but not a single significant drop in the eigenvalues. The fact that λ_{12} in itself is lower than our chosen threshold is no indicator that a curve and the correct degree was found. If we would accept the curve at the testing degree $d = 12$, we would get the curve shown in the right of Figure 6. It is obviously not a good approximation to the sample points. In relation to the structural diminution, the minimal eigenvalues of the Epicycloid decrease even more. We interpret this as follows: The higher the testing degree gets, the better the data may be approximated. The increasing values in the corrected plot are due to roundoff errors.

5.2 Shifting and preconditioning

Our method as presented here has one significant drawback: It is very sensitive to the location of the sample points. This is due to the following effect. If, for instance, in our example we shift the Euclidean sample points by a vector $(100, 100)$, then the corresponding parameters in $\tau_d(p_i)$ exhibit very high parameter values since the shifts are amplified by the large exponents in the polynomial bases. This results in the fact that our matrix P becomes more and more numerically ill-conditioned when the sample points are far away from the origin. The first two pictures of Figure 4 show how the estimated curve becomes more and more inaccurate the further away all sample points are form the origin.

So far we do not have a unified method to attack these numerical problems however we have several reasonable heuristics that work fine in parctise. As a first step one could a priory investigate the data and translate it so that it is not

too far away from the origin. One could also scale the data. After calculating the parameter vector this vector has to be transformed correctly to match the original data points again. We call this process preconditioning. The third picture of Figure 7 demonstrates the result of this method. There the data points have first been translated by a vector $(30, 30)$. This would normally result in a very badly conditioned matrix and the estimated curve would be far off the sample points (see middle picture). However now the whole data set is scaled by a factor of 0.01 which again moves the data points close to the origin. The last picture shows that after this preconditioning the estimated curve nicely matches the sample data again.

A more subtle method to attack numerical instabilities can be achieved by using topological curve invariants. In our example the middle picture could never stem from a locus generated by a dynamic geometry program: The curve obviously breaks up in (at least) two branches and $\mathcal{P}$ contains points on different branches. This is also true for the wrongly recognized Epicycloid of Figure 6. Using, for example, a Bernstein Basis (with respect to the appropriate bounded domain in each case) for representing and dealing with the involved polynomials could help to improve the conditioning problem, too.

5.3 Investigating curve invariants

The parameter vector of the curve is the eigenvector that corresponds to the smallest absolute eigenvalue. A search for curve invariants as proposed in Section 4 would require the generation of many instances of similar curves and the calculation of the orthogonal space. The orthogonal space would hint to specific dependencies on the curve's parameters. However, very often such dependencies also are indicated if one investigates only in one such parameter vector. In our example of Figure 2 after multiplying by a suitable factor the eigenvector of the degree four approximation takes the following form (four digits after the decimal point are shown):

$$\begin{aligned}\beta = (\ & 1.0,\ -9.4034 \cdot 10^{-8},\ 2.0,\ -8.8127 \cdot 10^{-8},\ 1.0,\ -28.08,\ -0.55999,\ -28.08,\\ & -0.55999,\ 237.3311,\ 7.8623,\ 40.2880,\ -423.9761,\ 66.5167,\ -1396.3979)\end{aligned}$$

Looking at these values one may suspect that the curve has the characteristic properties

$$\beta_2 = \beta_4 = 0, \qquad 2\beta_1 = \beta_3 = 2\beta_5, \qquad \beta_6 = \beta_8, \qquad \beta_7 = \beta_9.$$

In fact, comparing these conjectures with the coefficients of a generic limaçon generated by a computer algebra system shows that the above relations indeed hold for the general case. Using techniques like the PSLQ algorithm [8] that is able to "guess" integral relations between real numbers one could also use a single eigenvector to derive many more reasonable conjectures about the underlying curve type. Further research in this direction is in progress.

6 Conclusions

With a given set of points $\mathcal{P}$ representing an algebraic curve approximately, we can determine the curve's coefficients by computing eigenvalues and eigenvectors only. This article presented the first steps along this road. Still there are many open questions and problems to attack. The main research problems currently are:

- What are good ways of preconditioning?
- How to deal with curve types that depend on parameters?
- How to derive geometric transformations that map a curve to a kind of standard representation?
- How can one derive a reliable measure for the quality of the result?
- To what extent can randomization techniques be used to speed up the calculations?
- How to use topological curve characteristics to restrict the search space of potential curves?
- Can similar approaches be used within the more special class of rational algebraic functions?

References

1. Kortenkamp U. and Richter-Gebert, J.: *Grundlagen dynamischer Geometrie*, Zeichnung - Figur - Zugfigur (2001) 123-144
2. Kortenkamp U. and Richter-Gebert, J.: *Complexity issues in dynamic geometry* In *Festschrift in the honor of Stephen Smale's 70th birthday,* M. Rojas, F. Cucker (eds.), World Scientific, pp 355–404, (2002).
3. Laborde, J.-M. and Bellemain, F.: *Cabri-Geometry II*, Texas Instruments, 1993–1998.
4. Richter-Gebert J. and Kortenkamp U.: *Cinderella - The interactive geometry software*, Springer 1999; see also `http://www.cinderella.de`.
5. Taubin, G.: *An improved algorithm for algebraic curve and surface fitting*, Proc. ICCV'93, Berlin, Germany (may 1993) 658–665.
6. Taubin, G.: *Estimation of Planar Curves, Surfaces and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (1991) 1115-1138.
7. Tarel, J.-P. and Cooper, D. B.: *The Complex Representation of Algebraic Curves and its Simple Exploitation for Pose Estimation and Invariant Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 663-674.
8. H.R.P. Ferguson, D.H. Bailey and S. Arno: *Analysis of PSLQ, An Integer Relation Finding Algorithm*, Mathematics of Computation, Vol 68, No. 225 (Jan 1999) 351-369.
9. Z. Lei and T. Tasdizen and D.B. Cooper: *PIMs and Invariant Parts for Shape Recognition*, Proceedings of Sixth International Conference on Computer Vision (ICCV'98) (Jan 1998) 827-832.
10. Tasdizen, T. and Tarel, J.-P. and Cooper, D.B.: *Algebraic Curves That Work Better*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR'99) (June 1999) 35-41.