Generalized Principal Component Analysis for Algebraic Varieties

Approaching a Robust Method for Learning Vanishing Ideals

Matthias Himmelmann
Matriculation number: 4853871

Master's thesis
presented for gaining the degree
Master of Science in Mathematics

Supervisors: Prof. Dr. Bernd Sturmfels and
Prof. Dr. Christian Haase

Advisor: M.Sc. Sascha Timme

Department of Mathematics and Computer Science
Free University of Berlin
December 30, 2020

Table of Contents

1 Introduction

For the past centuries, an important aspect in applied mathematics has been the interpolation of $n+1$ pairs of complex numbers by a univariate polynomial. Numerous famous mathematicians such as Horner, Gauß, Lagrange and Newton worked on related problems. They have found out that $n+1$ distinct numbers with corresponding grid points can be perfectly approximated. A univariate polynomial of degree n is capable of attaining these numbers at the grid points, as can be seen in Figure 1.

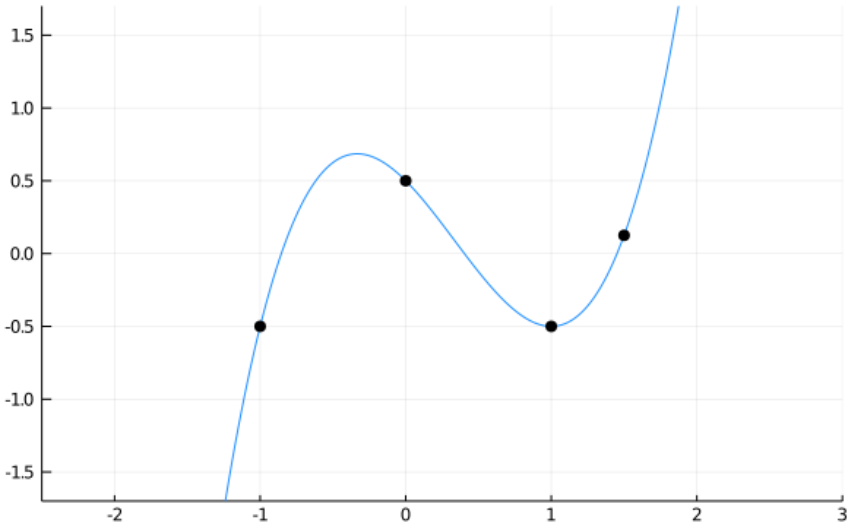

Figure 1: Interpolation of 4 points in the plane by $f(x) = x^3 - x^2 - x + 0.5$.

However, in real-life applications we regularly deal with many noisy data points. The problem's condition number can make it infeasible to find a polynomial of degree n that represents the data well. Plenty statistical literature deals with solving this issue that is also known as overfitting. It corresponds to using more parameters than necessary to approximate the data.

In more recent years, effort was made to generalize interpolation to multivariate polynomials. The Buchberger-Möller algorithm (cf. Möller and Buchberger [27]) computes the ideal of all polynomials vanishing on the sample points and has proven to be reliable in exact arithmetics (cf. Abbott et al. [1]). Extending this method to noisy data (cf. Heldt et al. [22]), the zero-dimensionality of the data points' ideal remains a drawback.

A different approach is given by Breiding et al. [5]. They use a multivariate Vandermonde matrix to calculate generators for the algebraic variety's vanishing ideal from point samples, truly generalizing interpolation. Indeed, the Vandermonde matrix has often been applied in numerical analysis, calculating polynomials vanishing on a set of samples (cf. Olver [28]). The fundamental difference to the Buchberger-Möller algorithm is calculating polynomials that approximate the topological and geometric structure of the data, other than finding all polynomials vanishing on the data points.

Imagine we are given 66 point samples as displayed in Figure 2. After visualizing the data, one can guess that these samples belong to a planar cubic. With this information, we try to calculate the equation that generated these points using a vector in the kernel of the multivariate Vandermonde matrix. In this case, the equation we are looking for is $y^2 - x^3 + x = 0$. Nonetheless, there is an issue: The multivariate Vandermonde matrix is ill-conditioned (cf. Pan [29]), making it impractical to use this method for complex, confluent and particularly noisy data.

Due to Ma et al. [24], a noise-resistent method for learning subspace arrangements, i.e. unions

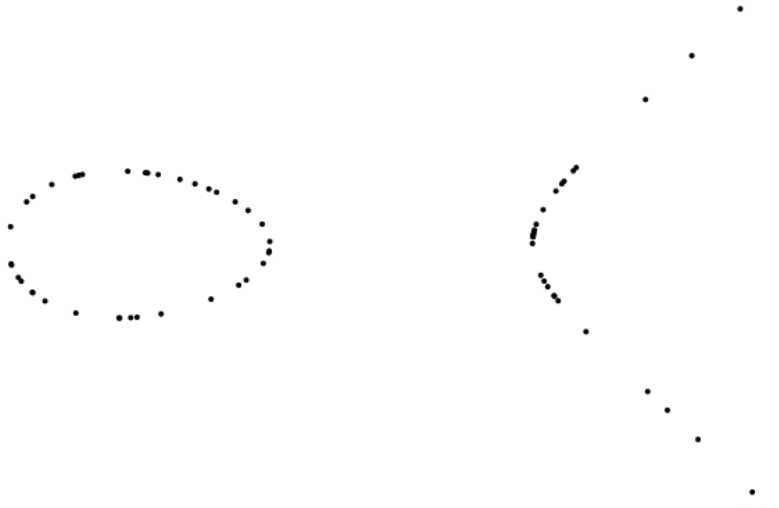

Figure 2: 66 Samples from the elliptic curve $y^2 - x^3 + x = 0$.

of linear spaces, is available. It involves the use of algebro-geometric techniques, since subspace arrangements can be described by the vanishing set of polynomial equations. We will make use of these ideas to generalize the previously proposed methods to arbitrary algebraic varieties. After introducing concepts from algebraic geometry in Chapter 2, the generalized principal component analysis (cf. Vidal et al. [35]) will be examined in Chapter 3. The initially strong assumptions about the underlying subspaces' properties will be mitigated in the process. The result will be a robust algorithm for learning subspace arrangements.

Similarly, for learning equations from arbitrary algebraic varieties, in Chapter 4 we initially assume noise-free data. A priori knowledge about the codimension of the variety and the maximal occuring degree among the vanishing ideal's generators is expected. For quantifying, how well the proposed algorithms perform compared to similar methods, suitable error functions are discussed. After having identified existing issues of the method `FindEquations` (cf. Breiding et al. [5]), a different algorithm for learning polynomial equations less prone to perturbations will be presented. As a performance measure, we can exploit the error functions to detect potential improvements, enabling us to present a variation of the method with higher accuracy. Prior to conducting experiments with different algebraic varieties, the issue of overfitting will be dealt with. Adjusting the algorithm reduces the impact of high variance.

In Chapter 5, the Cayley-Bacharach theorem imposes conditions on degree 3 curves passing through points coming from the intersection of two cubics. It is demonstrated that our method is capable of abiding by these rules. Conclusively, an outlook for future work will be given. Statistical learning theory will be the natural next step in the algorithm's development. It is conceivable that a stopping criterion involving regularization with respect to the complexity of the search space and the degree of the variety will allow our algorithm to run without prior knowledge.

2 Algebro-Geometric Basics

In this thesis, rudimentary knowledge about commutative algebra is assumed. Especially, polynomial rings and ideals are important tools, as summarized in Cox-Little-O'Shea [10]. Let us begin by introducing useful concepts from algebraic geometry. First and foremost, it is fundamental to understand what an algebraic variety is. We recall the definition from Michałek and Sturmfels [25, p. 19]:

Definition 2.1. *Let k be a field, let $k[x_1, \dots, x_D]$ be the polynomial ring over k in D variables and let $S \subset k[x_1, \dots, x_D]$ be a family of polynomials. The variety or vanishing locus of these polynomials is defined to be*

$$\mathcal{V}(S) = \{\boldsymbol{p} = (p_1, \dots, p_D) \in k^D \ : \ f(\boldsymbol{p}) = 0 \ \forall f \in S\}.$$

Given any subset $W \subset k^D$, we can define the vanishing ideal of W by

$$\mathcal{I}(W) = \{f \in k[x_1, \dots, x_D] \ : \ f(\boldsymbol{p}) = 0 \ \forall \boldsymbol{p} \in W\}.$$

Many textbooks require irreducibility for an algebraic variety, i.e. it cannot be written as the union of two properly contained, non-trivial varieties. The concept that was introduced in Definition 2.1 is called an algebraic set there. Any algebraic set can be decomposed into the union of algebraic varieties and irreducible algebraic sets correspond to vanishing ideals that are prime. For a proof of the latter, compare Proposition 2.3 in Michałek and Sturmfels [25, p. 20]. In this thesis, we will use the term algebraic variety for an algebraic set, as in the applied literature this definition is more prevalent.

If two polynomials f_1 and f_2 vanish on $W \subset k^D$, so do their sum and products with arbitrary elements of the underlying ring. Hence, $\mathcal{V}(S) = \mathcal{V}(\langle S \rangle)$ where $\langle S \rangle$ is the ideal generated by S. According to Hilbert's basis theorem (cf. Cox et al. [10, p. 76]), the ring $k[x_1, \dots, x_D]$ is Noetherian, so the inputs of the operator $\mathcal{V}$ are finitely generated ideals. Furthermore, the ideal $\mathcal{I}(W)$ is radical which can be seen in the following way. If $z \in W$ and $f(z)^m = 0$, it also holds that $f(z) = 0$, as k is a field. Since z is chosen arbitrarily, $f \in \mathcal{I}(W)$ and the ideal is radical. The restriction of the operators $\mathcal{V}$ and $\mathcal{I}$ is further justified by the following remark.

Remark 2.2. *Let k be an algebraically closed field. Then there is a 1-to-1 correspondence between the set of all varieties in k^D and the set of all radical ideals in $k[x_1, \dots, x_D]$ given by the operators $\mathcal{V}$ and $\mathcal{I}$ respectively. This fact is also known as Hilbert's Nullstellensatz. For a proof consult for example Michałek and Sturmfels [25, pp. 90-91].*

Considering the fields $k = \mathbb{R}$ and $k = \mathbb{C}$, we want to intersect a variety $V \subset \mathbb{C}^D$ with $\mathbb{R}^D$, especially when $D \leq 3$. As a consequence, we are able to visualize the vanishing set. Later on, intersecting and taking the union of varieties will be useful. Thus, we want to endow k^D with additional structure, namely a topology. The Zariski topology on k^D is defined by the algebraic varieties as closed sets, which is proven by the following proposition that uses aggregated statements from Gathmann [18, pp. 14-15].

Proposition 2.3. *The following properties hold:*

1. *The operators $\mathcal{V}$ and $\mathcal{I}$ reverse inclusions.*
2. *Let I_j for $j \in J$ be a family of ideals in $k[x_1, \dots, x_D]$. It holds that*

$$\mathcal{V}(\langle \bigcup_{j \in J} I_j \rangle) = \mathcal{V}(\sum_{j \in J} I_j) = \bigcap_{j \in J} \mathcal{V}(I_j).$$

3. *If $I_1, I_2 \subset k[x_1, \ldots, x_D]$ are ideals, then*

$$\mathcal{V}(I_1 \cap I_2) = \mathcal{V}(I_1 \cdot I_2) = \mathcal{V}(I_1) \cup \mathcal{V}(I_2).$$

Proof. In proving the statements, Definition 2.1 is useful.

1. For the operator $\mathcal{V}$, let $I_1 \subset I_2$ be ideals in $k[x_1, \ldots, x_D]$ and let $z \in \mathcal{V}(I_2)$. Therefore, any $f \in I_2$ satisfies $f(z) = 0$ and $g(z) = 0$ for each $g \in I_1$, because $I_1 \subset I_2$, implying that $z \in \mathcal{V}(I_1)$. Ultimately, $\mathcal{V}(I_1) \supset \mathcal{V}(I_2)$.

   For $\mathcal{I}$, let $Z_1 \subset Z_2$ be arbitrary sets in k^D. Choose any $f \in \mathcal{I}(Z_2)$. By definition, $f(z) = 0$ for each $z \in Z_2$. As $Z_1 \subset Z_2$, this yields that $f \in \mathcal{I}(Z_1)$. Thus, $\mathcal{I}(Z_1) \supset \mathcal{I}(Z_2)$ follows.

2. Since $\mathcal{V}(\bigcup_{j\in J} I_j) \subset \mathcal{V}(I_j)$ for each $j \in J$ and $\mathcal{V}(\sum_{j\in J} I_j) \subset \mathcal{V}(I_j)$ by Proposition 2.2.1 as $0 \in I_j$ for each $j \in J$, one has

$$\mathcal{V}(\langle \bigcup_{j\in J} I_j \rangle),\ \mathcal{V}(\sum_{j\in J} I_j) \subset \bigcap_{j\in J} \mathcal{V}(I_j).$$

   For the other inclusion, let $\mathbf{p} \in \bigcap_{j\in J} \mathcal{V}(I_j)$. This implies that for any $j \in J$, each $f \in I_j$ satisfies $f(\mathbf{p}) = 0$. Equivalently, any $f \in \bigcup_{j\in J} I_j$ vanishes on $\mathbf{p}$, which proves one part of the claim. Recall that $\sum_{j\in J} I_j$ consists of finite sums of the form $\sum_{j\in J} f_j$ with $f_j \in I_j$. Since $f_j(\mathbf{p}) = 0$ for each f_j in I_j, the sum of such polynomials also vanishes on $\mathbf{p}$.

3. On the one hand, if $\mathbf{p} \in \mathcal{V}(I_1) \cup \mathcal{V}(I_2)$, then $f(\mathbf{p}) = 0$ for all $f \in I_1$ or $g(\mathbf{p}) = 0$ for all $g \in I_2$. In any case, $h(\mathbf{p}) = 0$ for all $h \in I_1 \cap I_2$ and $(f \cdot g)(\mathbf{p}) = 0$, implying that

$$\mathcal{V}(I_1 \cap I_2),\ \mathcal{V}(I_1 \cdot I_2) \supset \mathcal{V}(I_1) \cup \mathcal{V}(I_2).$$

   On the other hand, let $\mathbf{p} \notin \mathcal{V}(I_1) \cup \mathcal{V}(I_2)$, so there exists a polynomial f in I_1 and a polynomial g in I_2 that do not vanish on $\mathbf{p}$. As a consequence, the product fg does not vanish on $\mathbf{p}$, because k contains no zero divisors. Therefore, $\mathcal{V}(I_1 \cdot I_2) = \mathcal{V}(I_1) \cup \mathcal{V}(I_2)$. By Proposition 2.2.1, one has

$$I_1 \cdot I_2 \subset I_1 \cap I_2 \subset I_1,\ I_2 \qquad \Rightarrow \qquad \mathcal{V}(I_1 \cdot I_2) \supset \mathcal{V}(I_1 \cap I_2) \supset \mathcal{V}(I_1),\ \mathcal{V}(I_2).$$

   Having already shown that $\mathcal{V}(I_1 \cdot I_2)$ and $\mathcal{V}(I_1) \cup \mathcal{V}(I_2)$ coincide, the above containment yields that $\mathcal{V}(I_1) \cup \mathcal{V}(I_2) = \mathcal{V}(I_1 \cap I_2)$.

□

Parts 2 and 3 of the above proposition prove that finite unions and arbitrary intersections of closed sets are again closed. Furthermore, $\mathcal{V}(0) = k^D$ and $\mathcal{V}(1) = \emptyset$, so the Zariski topology is indeed a topology.

There is another notion of varieties in projective space. To distinguish both definitions, we call varieties from Definition 2.1 affine varieties. Contrary to affine space k^D, the D-dimensional projective space $\mathbb{P}_k^D$ ecompasses points in $k^{D+1} \setminus \{0\}$ with the equivalence relation $a \sim \lambda a$ for $\lambda \in k \setminus \{0\}$ and $a \in k^{D+1}$. Let $[a_0 : \cdots : a_D]$ denote a projective point to emphasize the difference between points in affine and projective space.

Definition 2.4. *A projective variety is an algebraic set in $\mathbb{P}_k^D$ of the form*

$$\mathcal{V}(S) = \{\boldsymbol{a} = [a_0 : \cdots : a_D] \in \mathbb{P}_k^D : f(\boldsymbol{a}) = 0 \text{ for } f \in S\}$$

for a family of homogeneous polynomials $S \subset k[x_0, \ldots, x_D]$.

The polynomials' homogeneity is necessary, as otherwise Definition 2.4 is not well-defined. In $\mathbb{P}_k^D$, the equivalence relation $a = \lambda a$ for each $\lambda \in k \setminus \{0\}$ and any $a \in \mathbb{P}_k^D$ applies. Conversely, the evaluation of an arbitrary polynomial f in a point $a \in \mathbb{P}_k^D$ depends on the scaling. As a consequence, while a polynomial f vanishes on λa for some $\lambda \in k \setminus \{0\}$, possibly it does not vanish on a. An example is $x - 2$.

Homogeneous polynomials of degree n have the desireable property that

$$f(\lambda a_0, \ldots, \lambda a_D) = \lambda^n f(a_0, \ldots, a_D),$$

implying that f vanishes on any representative of $[a_0 : \cdots : a_D] \in \mathbb{P}_k^D$ if and only if it vanishes on some representative. Now, as there is an inclusion from k^D to $\mathbb{P}_k^D$ via any of the standard affine charts $\{z_i \neq 0\}$ of $\mathbb{P}_k^D$, we can embed any affine variety V into $\mathbb{P}_k^D$. However, it might not be a projective variety, so we consider its projective closure $\overline{V}$, defined by the homogenization of all polynomials in $\mathcal{I}(V)$. We denote the homogenization of an ideal I by I^h, implying $\overline{V} = \mathcal{V}((\mathcal{I}(V))^h) \subset \mathbb{P}_k^D$.

As an example, let us homogenize the polynomial $f = x_1^2 + x_1 + x_2 - 1 \in \mathbb{C}[x_1, x_2]$. Its homogenization lives inside the coordinate ring $\mathbb{C}[x_0, x_1, x_2]$ of $\mathbb{P}_{\mathbb{C}}^2$. As the total degree of f is 2, every term in its homogenization needs to have degree 2. By multiplying powers of x_0 whenever necessary, f becomes $f^h = x_1^2 + x_0 x_1 + x_0 x_2 - x_0^2$. The inverse operation is given by selecting an affine chart in $\mathbb{P}_{\mathbb{C}}^2$, for example $\{z_0 \neq 0 \ : \ [z_0 : z_1 : z_2] \in \mathbb{P}_{\mathbb{C}}^2\}$. Subsequently, the corresponding variable in the coordinate ring $\mathbb{C}[x_0, x_1, x_2]$ is mapped to 1, so $x_0 \mapsto 1$. Choosing the same affine chart the polynomial f lives in returns the original polynomial. For instance, $f(x_1, \ldots, x_D) = f^h(1, x_1, \ldots, x_D)$.

Remark 2.5. *It is not sufficient to homogenize only the generators of an ideal. For example, every generator in $I = \langle x - z^2, y - z^2 \rangle$ has total degree 2. However, $x - y \in I^h$ because $x - y \in I$. Nevertheless, $x - y \notin J = \langle \omega x - z^2, \omega y - z^2 \rangle$, so $J \subsetneq I^h$. Selecting elements of I, whose homogeneization generates I^h is a problem. The reduced Gröbner basis of I is a family of polynomials $\mathcal{G}$ that satisfies $\langle \mathcal{G}^h \rangle = I^h$, solving this issue (cf. Michałek and Sturmfels [25, pp. 6-10]).*

After choosing an embedding, important invariants of an algebraic variety are its dimension and degree. This thesis relies on a definition from Hartshorne [21, p. 5], because a vector space as a special case gives a good intuition for it.

Definition 2.6 (Dimension and Degree)**.** *Let V be an algebraic variety. V's dimension is defined by the maximum length d of chains $V_0 \subsetneq \cdots \subsetneq V_d$, comprising irreducible, non-empty subvarieties V_i of V. Allowing $\overline{k}^D$ as domain, we calculate the vanishing set of $\mathcal{I}(V)$ in its algebraically closed ambient space $\overline{k}^D$. Let L be an affine subspace of dimension $D - \dim(V)$ in general position. The degree of V is defined as the amount of points in $\overline{V} \cap \overline{L} \subset \mathbb{P}_{\overline{k}}^D$.*

The notion of degree does not distinguish between the varieties $\mathcal{V}(x)$ and $\mathcal{V}(x^2)$ viewed in $\mathbb{C}^1$. Only the radical of the vanishing ideal is considered. Intuitively, in the latter there is a double point in 0. The rich theory of schemes addresses this behavior (cf. Hartshorne [21, §II]).

Example 2.7. Consider the union of a line and a plane in $\mathbb{C}^3$ given by the vanishing set of the ideal $\langle x_1 x_3, x_2 x_3 \rangle$. In the plane, we can find the (maximal) chain of prime ideals

$$\{0, 0, 0\} = \mathcal{V}(x_1, x_2, x_3) \subsetneq \mathcal{V}(x_1, x_3) \subsetneq \mathcal{V}(x_3).$$

The only irreducible subvarieties of the line are the points. Therefore, the dimension of this variety is 2. For the degree, a generic line meets $\mathcal{V}(x_1, x_2)$ in a point in $\mathbb{P}_{\mathbb{C}}^3$ that corresponds to

∞ with probability 1. Choosing the embedding

$$\iota : \mathbb{C}^3 \hookrightarrow \mathbb{P}^3_{\mathbb{C}}, \qquad (a_1, a_2, a_3) \mapsto [a_1 : a_2 : a_3 : 1],$$

this point of intersection corresponds to $[0 : 0 : 1 : c]$ for some $c \in \mathbb{C}$ or $[0 : 0 : 0 : 1]$. Consequently, the degree of $\mathcal{V}(x_1x_3, x_2x_3)$ is 2, as a generic affine line intersects a plane in exactly 1 point.

This paves the way for a famous theorem about the number of intersection points of two plane curves. The following generalization to hypersurfaces does not involve counting multiplicities of the intersections and consequently avoids a scheme-theoretic language.

Theorem 2.8 (Bézout's Theorem)**.** *Let $\mathcal{V}(f_1), \ldots, \mathcal{V}(f_m) \subset \mathbb{P}^D_{\overline{k}}$ be irreducible, non-trivial hypersurfaces in relative general position and let $m \leq D$. Then, for the projective variety $V = \mathcal{V}(f_1, \ldots, f_m)$ it holds that*

$$\dim(V) = D - m \qquad \textit{and} \qquad \deg(V) = \prod_{i=1}^{m} \deg(f_i)$$

where f_i is the generator of $\mathcal{I}(\mathcal{V}(f_i))$ for $i \in \{1, \ldots, m\}$. Omitting the assumption that the varieties are irreducible, lie in relative general position and that f_i generates $\mathcal{I}(\mathcal{V}(f_i))$, the Bézout bound yields an upper bound on V's degree:

$$\deg(V) \leq \prod_{i=1}^{m} \deg(f_i).$$

Proof. A proof of Bézout's theorem can be found in Shafarevich [34, §IV.2]. By Krull's principal ideal theorem, $\dim(V) \geq D - m$. Equality holds, because each f_i is not a zero divisor in the ring $\overline{k}[x_0, \ldots, x_D]/\langle f_1, \ldots, f_{i-1} \rangle$ by the relative generality of the hypersurfaces (cf. Michałek and Sturmfels [25, p. 26]). □

The Bézout bound holds in this generality, because we can decompose V into its irreducible components and apply the first part of the theorem to each combination. However, intersection points with higher multiplicity are possible, because $\langle f_i \rangle$ is not necessarily a prime ideal and the hypersurfaces are not irreducible. Also, nontrivial subvarieties possibly coincide, since the hypersurfaces do not lie in relative general position. Therefore, only an inequality is achievable.

3 Generalized Principal Component Analysis

In this chapter, we shift to a data-scientific viewpoint. Assume we are given N samples from the union of several linear spaces. The segmentation problem deals with assigning each of the sample points to a corresponding linear space. In doing so, the samples are partitioned. Our ultimate goal is to learn the generating equations of an algebraic variety. Choosing the intermediate step of treating subspace arrangements first, will prove useful in the development of an appropriate method. We will soon see that subspace arrangements are varieties as well.

The generalized principal component analysis, thoroughly laid out in both Ma et al. [24] and Vidal et al. [35], relies on algebro-geometric techniques to solve the aforementioned segmentation problem. In the presence of noise, statistical techniques help to find a relatively close arrangement of subspaces. We approach this in three steps, making different assumptions at each one:

3.1 First, we assume that our data points are free of disturbances and that we know the amount of occuring linear spaces and their corresponding dimensions. The algorithm generalized principal component analysis (GPCA) is developed in the process.

3.2 Afterwards, we relax the conditions of having a priori knowledge about the subspaces; the method recursive GPCA emerges.

3.3 Finally, we assume that our samples are corrupted with noise. Here, we once again assume that we know the number and dimensions of the linear spaces. Arguably, this is the major achievement in the line of GPCA methods.

This step-by-step program paves the way for a more general method similar to GPCA that works for arbitrary algebraic varieties.

3.1 The Noise-Free Case

Fix a field k. As mentioned earlier, we want to use $k = \mathbb{R}$ or $k = \mathbb{C}$. Let us denote by k^D the D-dimensional ambient space. According to Ma et al. [24], this enables us to formulate the following definition:

Definition 3.1. *A subspace arrangement is defined to be $\mathcal{A} = V_1 \cup \cdots \cup V_n \subseteq k^D$ for linear subspaces $V_i \subseteq k^D$. The dimension of the subspaces is labeled by $d_i = \dim V_i$ and the codimension by $c_i = D - d_i$ respectively. For $S \subseteq \{1, \ldots, n\}$, let us denote by $V_S = \bigcap_{s \in S} V_s$ the intersection over all indices in S. We call the dimension of this intersection d_S and the codimension c_S. Finally, we call $\mathcal{A}$ transversal, if for any choice of index set $S \subseteq \{1, \ldots, n\}$ the equality $c_S = \min(D, \sum_{i \in S} c_i)$ holds.*

Taking into account Proposition 2.3 and since any linear space is the vanishing set of linear forms with $a_{i,j} \in k^D$ and $c_i \leq D$, any subspace arrangement is an algebraic variety:

$$\begin{aligned} \mathcal{A} = V_1 \cup \cdots \cup V_n &= \mathcal{V}\left(a_{1,1} \cdot \mathbf{x},\ \ldots,\ a_{1,c_1} \cdot \mathbf{x}\right) \cup \cdots \cup \mathcal{V}\left(a_{n,1} \cdot \mathbf{x},\ \ldots,\ a_{n,c_n} \cdot \mathbf{x}\right) \\ &= \mathcal{V}\left(\bigcap_{i=1}^{n} \langle a_{i,1} \cdot \mathbf{x}, \ldots, a_{i,c_i} \cdot \mathbf{x} \rangle\right) \\ &= \mathcal{V}\Bigg(\underbrace{\prod_{i=1}^{n} \langle a_{i,1} \cdot \mathbf{x}, \ldots, a_{i,c_i} \cdot \mathbf{x} \rangle}_{=\mathcal{P}(\mathcal{A})}\Bigg) \end{aligned}$$

Here, the associated polynomial ring is $k[\mathbf{x}] = k[x_1, \ldots, x_D]$. The product of $\mathcal{A}$'s individual linear spaces' vanishing ideals is denoted by $\mathcal{P}(\mathcal{A})$. Since every factor in $\mathcal{P}(\mathcal{A})$ is a homogeneous

ideal, the entire product of ideals is homogeneous and so is its radical $\mathcal{I}(\mathcal{A})$. The polynomials that are vanishing on $\mathcal{A}$ potentially have a strictly lower degree than the number of subspaces n, e.g. a transversal arrangement $\mathcal{B}$ of two lines and a plane in $\mathbb{R}^3$. $\mathcal{B}$ can be embedded into the transversal union of two planes $\mathcal{D}$, as every two lines span a plane. As $\mathcal{D}$ has polynomials of degree 2 that vanish on it by the construction of $\mathcal{P}(\mathcal{D})$, these polynomials also vanish on the transversal arrangement $\mathcal{B}$ because $\mathcal{B} \subset \mathcal{D}$.

We can deduce that the vanishing ideal $\mathcal{I}(\mathcal{A})$ is a direct sum

$$\mathcal{I}(\mathcal{A}) = I_m \oplus I_{m+1} \oplus \cdots \oplus I_n \oplus I_{n+1} \oplus \dots$$

of homogeneous components I_ℓ and the smallest $m \leq n$ so that $I_m \neq \{0\}$. Alternatively, $I_\ell = \mathcal{I}(\mathcal{A}) \cap k[\mathbf{x}]_\ell$, the latter set denoting the homogeneous component of degree ℓ in the graded ring $k[\mathbf{x}]$. The finite-dimensional vector space $k[\mathbf{x}]_\ell$ has a basis given by the monomials of degree ℓ. Its dimension is given by

$$\dim k[\mathbf{x}]_\ell = M_\ell^D = \begin{pmatrix} \ell + D - 1 \\ D - 1 \end{pmatrix}.$$

An option is to compute the equations in each homogeneous degree. Computing the equations in the lowest homogeneous component of

$$\mathcal{P}(\mathcal{A}) = P_n \oplus P_{n+1} \oplus \dots$$

is sufficient, as $\mathcal{V}(\mathcal{I}(\mathcal{A})) = \mathcal{V}(\mathcal{P}(\mathcal{A}))$ by Lemma 2.8 in Ma et al. [24]. In doing so, we introduce the Veronese embedding of degree h:

$$\begin{aligned} \nu_\ell : k^D &\rightarrow k^{M_\ell^D} \\ \nu_\ell \begin{pmatrix} x_1 \\ \vdots \\ x_D \end{pmatrix} &= \begin{pmatrix} x_1^\ell \\ x_1^{\ell-1} x_2 \\ \vdots \\ x_D^\ell \end{pmatrix}. \end{aligned}$$

The strategy employed in GPCA relies on the Veronese embedding ν_ℓ: Assume we are given a set of noise-free samples $\Omega = \{z_1, \dots, z_N\} \subset \mathcal{A} \subset k^D$ from a transversal subspace arrangement $\mathcal{A}$ and assume we know the dimensions d_i and the number of subspaces n. First, we compute a Matrix L_n, consisting of the n-th Veronese embeddings of these samples.

$$L_n = \begin{pmatrix} \nu_n(z_1)^T \\ \vdots \\ \nu_n(z_N)^T \end{pmatrix} \in k^{N \times M_n^D}$$

Computing the kernel of this matrix yields all homogeneous polynomials of degree n that vanish on Ω. For example, this can be accomplished using the singular value decomposition (SVD). Let C be the matrix whose columns form a basis of the kernel prescribed by the kernel-finding method of our choice. Then, let

$$Q(\mathbf{x}) = (q_1, \dots, q_m) = C^T \nu_n(\mathbf{x})$$

be the generating polynomials of $\mathcal{I}(\Omega) \cap k[\mathbf{x}]_n$. The following theorem on the sampling of an algebraic set will prove useful in justifying the choice of generating polynomials in Q.

Theorem 3.2 (Theorem 2.9 in Ma et al. [24]). *Consider a set $\emptyset \subsetneq S \subset k^D$ with vanishing ideal $\mathcal{I}(S)$ generated by polynomials in $k[x_1, \ldots, x_D]_{\leq n} = k[\boldsymbol{x}]_0 \oplus \cdots \oplus k[\boldsymbol{x}]_n$. There is a finite set $\Omega = \{z_1, z_2, \ldots, z_N\}$ such that $\mathcal{I}(\Omega) \cap k[\boldsymbol{x}]_{\leq n}$ generates $\mathcal{I}(S)$.*

Proof. The proof of this can be found in Ma et al. [24]. □

Specifically, for sufficiently large choices of Ω, $\mathcal{I}(\mathcal{A})$ can be generated by

$$\mathcal{I}(\Omega) \cap k[\mathbf{x}]_{\leq n} \subseteq k[\mathbf{x}]_0 \oplus k[\mathbf{x}]_1 \oplus \cdots \oplus k[\mathbf{x}]_n = k[\mathbf{x}]_{\leq n}.$$

By Hilbert's Nullstellensatz, $\mathcal{I}(\mathcal{A}) = \sqrt{\mathcal{P}(\mathcal{A})}$. Consequently, $\mathcal{I}(\Omega) \supseteq \mathcal{I}(\mathcal{A}) \supseteq \mathcal{P}(\mathcal{A})$ by Proposition 2.3, as $\Omega \subset \mathcal{A}$. Consider the following theorem on the Hilbert functions of $\mathcal{I}(A)$ and $\mathcal{P}(\mathcal{A})$:

Theorem 3.3 (4.7 in Derksen [12]). *For a transversal arrangement $\mathcal{A}$ and for $i \geq n$ it holds that*

$$h_{\mathcal{I}(\mathcal{A})}(i) = h_{\mathcal{P}(\mathcal{A})}(i).$$

Here, h_K denotes the Hilbert function of K with $h_K(i) = \dim(K_i)$ for the homogeneous component K_i of degree i.

Proof. For the concise development and a proof of the statement, compare Derksen [12]. □

Hence, the dimensions of the finite-dimensional vector spaces I_ℓ and P_ℓ are identical for $\ell \geq n$. Since $\mathcal{I}(\mathcal{A}) \supseteq \mathcal{P}(\mathcal{A})$ are homogeneous ideals, each of their homogeneous components satisfies $I_\ell \supseteq P_\ell$. As the dimensions of I_n and P_n agree by Theorem 3.3, the equality $I_n = P_n$ follows. Taking into account Theorem 3.2, for appropriate choices of Ω, the n-th graded part $\mathcal{I}(\Omega) \cap k[\mathbf{x}]_n$ generates $\mathcal{P}(\mathcal{A})$. The previous observation $\mathcal{P}(\mathcal{A}) = \sqrt{\mathcal{I}(\mathcal{A})}$ implies

$$\mathcal{V}(Q) = \mathcal{V}\left(\mathcal{I}(\Omega) \cap k[\mathrm{x}]_n\right) = \mathcal{V}\left(\mathcal{P}(\mathcal{A})\right) = \mathcal{V}\left(\mathcal{I}(\mathcal{A})\right)$$

which justifies our construction of Q.

First, the generalized principal component analysis computes the bases of the tangent spaces at n different sample points using the jacobian matrix

$$\mathcal{J}(Q)(z_i) = C^T \nabla \nu_n(z_i)$$

at an arbitrary point $z_i \in \Omega$. Its nullspace is then calculated by a kernel method, such as the singular value decomposition. This procedure yields a basis $B_i = (b_1, \ldots, b_{d_i})$ of the linear space V_i containing z_i. Since the column vectors of $\mathcal{J}(Q)(z_i)$ span the orthogonal complement of $\mathcal{V}(Q)$'s tangent space at z_i, its kernel spans the tangent space.

As a second step, all sample points z_j that satisfy $B_i^T z_j = 0$ are assigned to the space V_i. Already assigned samples are excluded from finding further bases. Subsequently, a new sample point is chosen from Ω and the first step is repeated until there are no points left to segment.

This method works especially well for transversal subspace arrangements, as the tangent space at a non-origin point yields the entire corresponding linear subspace. In particular, a linear space's tangent space is equal at every point. Assuming that $\mathcal{A}$ is transversal, almost all sample points lie on a unique subspace of $\mathcal{A}$, if Ω is sampled randomly. However, the tangent space is a local property of a variety. Therefore, the segmentation of the samples using tangent spaces is not as useful for arbitrary algebraic varieties.

3.2 Recursive GPCA

The GPCA algorithm presented in the previous section depends on idealized circumstances. For example, it assumes that both, the number of subspaces and their dimensions are known. If any of these two assumptions is not met, adjustments need to be made. While the samples are still noise-free, in the following we neither know the amount of linear spaces, nor their dimension.

According to Vidal et al. [35], a method called recursive GPCA solves this problem. Starting with $n = 1$, GPCA is recursively applied to find out whether L_n is rank-deficient. If so, the polynomials Q are computed from L_n's kernel and the bases B_i corresponding to the n linear spaces V_i are calculated via Q's jacobian, as was done in the original algorithm.

All points z_j that satisfy $B_i^T z_j = 0$ are added to the subspace with basis B_i and are omitted from Ω. These particular samples z_j are aggregated in a set X_i. For this new input of samples and V_i as new ambient space, the algorithm is applied again. In doing so, we find out whether there are lower-dimensional subspaces that fit this set of points. Afterwards, n is increased by 1. The procedure is repeated until each X_i can no longer be segmented into lower-dimensional subspaces or a prescribed maximum number of subspaces $n_{\max}$ is reached. To gain more insights into the algorithm, let us use the following example.

Example 3.4. Figure 3 depicts the transversal arrangement of two lines and a plane. The corresponding vanishing ideal is given by $\langle yz,\ x^2z - z^3 \rangle \subset \mathbb{R}[x, y, z]$. Assume a sample set that contains sufficient noise-free points from each linear space. Accordingly, let 5 non-zero random points from each line and 10 non-colinear and non-zero random points from the plane be given, so $\Omega = \{z_1, \ldots, z_{20}\}$. We want to employ the recursive GPCA method with $n_{\max} = 3$.

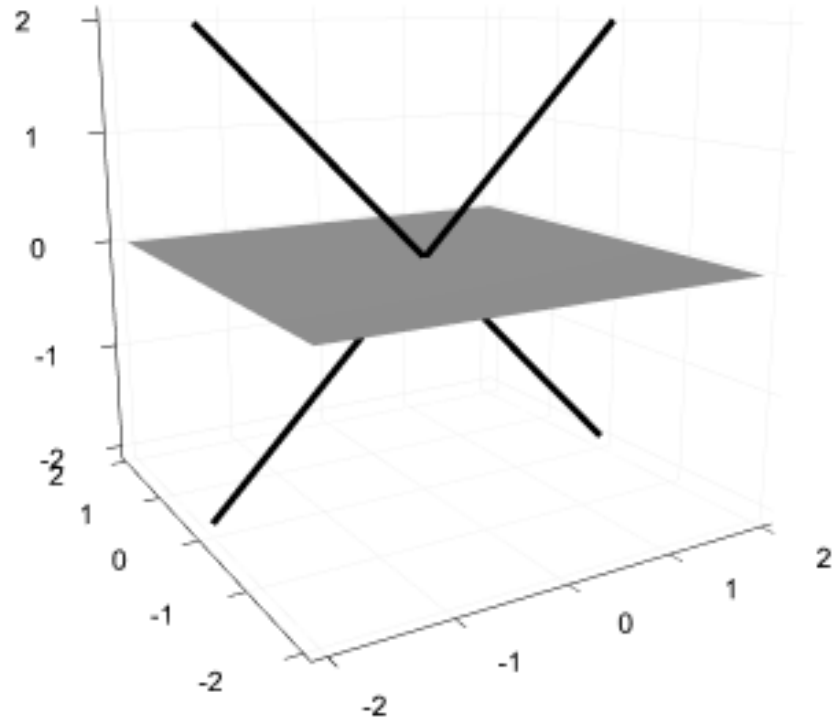

Figure 3: Transversal arrangement of a plane $\{z = 0\}$ and two lines defined in $\{y = 0\}$.

The algorithm starts with $n = 1$. There is no linear form that vanishes on all sample points, as the data points do not lie in a common plane by construction. Consequently, the kernel of L_1 is trivial, corresponding to the fact that the only linear subspace containing the arrangement is $\mathbb{R}^3$ itself. Proceeding with $n = 2$, the matrix L_2 is rank-deficient and its kernel contains a vector

corresponding to a polynomial of degree 2. This polynomial is given by yz and describes the transversal arrangements of the two planes $E_1 = \{y = 0\}$ and $E_2 = \{z = 0\}$. Subsequently, we partition all sample points Ω to the corresponding plane by calculating the kernel of $\mathcal{J}(yz)(z_i)$ for some z_i. By construction, 10 points are matched with E_1 while 10 points are matched with E_2, creating two separate sets of samples X_i corresponding to the plane E_i. Performing the algorithm again with X_1 in the ambient space E_1, yields a matrix L_n with full rank for $n \leq n_{\max}$, because the polynomials are viewed in $\mathbb{R}[x, y, z]/\mathcal{I}(E_1) = \mathbb{R}[x, y]$. Hence, the points in X_1 cannot be segmented further and E_1 is the final result.

Consider X_2 in the ambient space E_2. Notice that the corresponding coordinate ring satisfies $\mathbb{R}[x, y, z]/\mathcal{I}(E_2) = \mathbb{R}[x, z]$, so the y-coordinate is omitted. Thus, the kernel of L_1 is trivial: Only the degree 1 polynomial y vanishes on the sample X_2, because the points are assumed to lie on two distinct lines. In L_2, we find a vector in the kernel, which corresponds to $(x-z)(x+z) = x^2-z^2$. In each case, this enables us to assign 5 points each to lines ℓ_1 and ℓ_2 by calculating the tangent space at an arbitrary point $z_i \in X_2$ and segmenting all points that are orthogonal to the kernel of $\mathcal{J}(x^2 - z^2)(z_i)$. After picking a $z_i \in \Omega$ that has not been segmented yet, we repeat the process. In the resulting coordinate ring $x = z$ or $x = -z$, so these points cannot be segmented any further. As a result, the algorithm partitions Ω into 3 parts, corresponding to ℓ_1, ℓ_2 and E_1.

In other words, the recursive GPCA algorithm finds two lines and a plane. The sample points Ω are segmented according to the space they correspond to by computing the inner product of each point with the basis vectors of the orthogonal complement. If the inner product vanishes for each basis element of a linear space, the sample point belongs to that space. We are optimistic that this procedure can be applied to our method for learning an algebraic variety's vanishing ideal. A comparable approach is discussed in Chapter 4.

3.3 Noisy Samples

From now on, let us assume that our samples $\Omega = \{z_1, \ldots, z_N\}$ are corrupted with isotropic Gaussian noise. That is,

$$z_i = \hat{a}_i + n_i \quad \text{for} \quad i = 1, \ldots, N.$$

Here, $\hat{a}_i$ is a point on $\mathcal{A}$ that depends on the coefficient matrix C and n_i is an independent Gaussian random noise with covariance matrix given by the identity matrix times a positive constant. This notion stems from the ideas presented in Ma et al. [24].

In this section, we will assume that the number of subspaces n and the corresponding dimensions $d_1, \ldots, d_n$ are known. The tangent space is a local property of an algebraic variety. In a subspace arrangement, the tangent space is constant on each linear subspace. This property can be used to segment sample points. In general, this procedure does not work, so the tangent space is not of interest for us.

Now, even when knowing the correct number of subspaces n and their dimensions d_i, noise in the samples will almost always lead to L_n having full rank and it is therefore not mathematically clear, which singular vectors to pick. An applicable heuristic is to choose the smallest few singular values of L_n and the corresponding singular vectors. However, we do not know a suitable treshold a priori. An option is to find out up to what number we want to pick the singular values. An alternative is knowing, how many singular values we want to use.

We try to solve this problem by using a least square fitting approach. The idea is to minimize the mean squared distance

$$\min_{\hat{z}_i \in \mathcal{A}} \frac{1}{N} \sum_{i=1}^{N} ||z_i - \hat{z}_i||^2.$$

Optimizing the above proves to be a difficult task, as the closest point $\hat{z}_i$ on the variety is a complicated polynomial function in the coefficient matrix C. Using Taylor's theorem, we can avoid this issue by calculating a first-order approximation of the term $z_i - \hat{z}_i$, condensed in the following proposition.

Proposition 3.5. *Assume that the polynomials in Q are linearly independent. Let $z \in k^D$ and denote the closest point to z on the algebraic variety $\mathcal{A} = \mathcal{V}(Q)$ by $\hat{z}$. Then the Sampson distance (cf. Sampson [33])*

$$Q(z)^T \left([\mathcal{J}(Q)(z)]^T \mathcal{J}(Q)(z) \right)^\dagger Q(z)$$

is a first-order approximation of the squared euclidian distance $||z - \hat{z}||^2 = dist(z, \mathcal{A})^2$.

Proof. Calculating the Taylor expansion of $Q(\hat{z})$ demonstrates

$$Q(\hat{z}) = Q(z) + \mathcal{J}(Q)(z)(\hat{z} - z) + \mathcal{O}(||\hat{z} - z||^2).$$

Q contains generators of $\mathcal{A}$'s vanishing ideal and $\hat{z} \in \mathcal{A}$, so $Q(\hat{z}) = 0$. Omitting the terms of higher order indicates that

$$\begin{aligned} [\mathcal{J}(Q)(z)]^T \mathcal{J}(Q)(z) \cdot (z - \hat{z}) &\approx [\mathcal{J}(Q)(z)]^T Q(z) \\ \Rightarrow (z - \hat{z}) &\approx \left([\mathcal{J}(Q)(z)]^T \mathcal{J}(Q)(z) \right)^\dagger [\mathcal{J}(Q)(z)]^T Q(z). \end{aligned}$$

Here, the symbol † signifies the Moore-Penrose inverse (cf. Penrose [30]). This theory is employed, as we do not know, whether the $m \times m$ product of jacobians has full rank. In this case, m is the number of equations in Q. Finally, the above equations transform to

$$\begin{aligned} ||z - \hat{z}||^2 &\approx Q(z)^T \mathcal{J}(Q)(z) \left(\left([\mathcal{J}(Q)(z)]^T \mathcal{J}(Q)(z) \right)^\dagger \right)^2 [\mathcal{J}(Q)(z)]^T Q(z) \\ &= Q(z)^T \left(\mathcal{J}(Q)(z) [\mathcal{J}(Q)(z)]^T \right)^\dagger Q(z), \end{aligned}$$

concluding the proof. □

Using the above proposition, we can approximate the mean euclidian distance from our data points Ω to the subspace arrangement $\mathcal{A}$ by the Sampson distance $\mathcal{L}_S(C; \Omega, n)$

$$\frac{1}{N} \sum_{i=1}^{N} ||\hat{z}_i - z_i||^2 \approx \frac{1}{N} \sum_{i=1}^{N} Q(z_i)^T \left(\mathcal{J}(Q)(z_i) [\mathcal{J}(Q)(z_i)]^T \right)^\dagger Q(z_i) = \mathcal{L}_S(C; \Omega, n) \tag{1}$$

for $Q(\mathbf{x}) = C^T \nu_n(\mathbf{x})$ with coefficient matrix $C \in k^{m \times M_n^D}$. Geometrically, in a non-singular point z_i the jacobian $\mathcal{J}(Q)(z_i)$ contains m linear independent vectors from the orthogonal complement of the tangent space of $\mathcal{A}$ at z_i. The number m is equivalent to the number of linear independent polynomials in P_n. Consequently, m is equal to the dimension of I_n by Theorem 3.3. Thinking further, our goal is to calculate the euclidian distance $||z_i - \hat{z}_i||^2$ without knowing $\hat{z}_i$, which is a fundamental difficulty. To understand why this is difficult, let us consider Figure 4 that shows a parabola under the naïve objective function $||Q(\mathbf{x})||^2$.

The dashed lines depict the contour lines $Q(x, y) = \varepsilon$ and $Q(x, y) = -\varepsilon$. Notice that the points a, b have the same measure $||Q(a)|| = ||Q(b)||$ from the parabola. While d' represents the euclidian distance from the points to the original parabola, d'' is calculated using a line orthogonal to the contour line $Q(x, y) = \pm\varepsilon$. In the points of high curvature, i.e. points close to the origin, the contour lines are further away from the original curve than where the parabola is almost linear. In the euclidian distance, a and b hardly have the same distance to our parabola. Only the special choice of distance function $||\cdot||^2$ makes this behaviour possible.

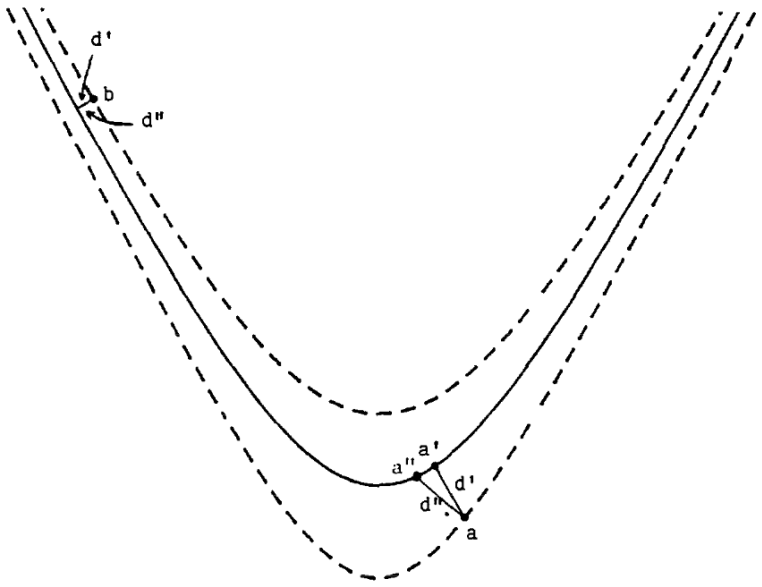

Figure 4: Three Contour Lines of $Q(\mathbf{x})^T Q(\mathbf{x}) = ||Q(\mathbf{x})||^2$, where $Q(x, y) = y - x^2$.
This image was taken from Sampson [33]

As a consequence, points far away from the origin are more heavily weighted than points close to the point of highest curvature in the mean squared error $\frac{1}{N}\sum_{i=1}^{N} ||Q(z_i)||^2$. To combat this, Sampson [33] suggests a method for planar curves: To calculate d'' the paper uses a first-order approximation of Q. In most situations, this is a good approximation of the desired euclidian distance d', which is

$$d' = ||a' - a|| \approx d'' = \frac{||Q(a)||}{||\nabla Q(a)||}$$

at a point a. In the case $m = 1$ of Q only containing a single polynomial, our objective function (1) is equivalent to the above distance. The Sampson distance is the natural extension of this idea: At any point z_i, our goal is to calculate the distance to the variety $\mathcal{A}$. The euclidian distance from z_i to $\hat{z}_i$, i.e. the closest point on $\mathcal{A}$ to z_i, is given by an orthogonal line on $\mathcal{A}$ through z_i. Calculating the orthogonal complement to $\mathcal{A}$'s tangent space at $\hat{z}_i$ produces the desired result. By a previous observation, a generating system of the orthogonal complement in regular points is given by the row vectors of $\mathcal{J}(Q)(\hat{z}_i)$. Since we do not have access to this, we use a linear approximation of Q in $\hat{z}_i$ which is exactly how we constructed (1).

Remark 3.6. *In our setting, the subspace arrangement $\mathcal{A}$ is calculated via the vector of polynomials Q. If the naïve loss function $||Q(\boldsymbol{z})||^2$ is used, points far away from the origin are usually weighted more heavily than points close to the origin. Assuming that there is a polynomial of degree greater than 1 in Q, $||Q(\boldsymbol{z})||^2$ becomes larger, the farther away $\boldsymbol{z}$ is from the origin. This phenomenon even happens, when the euclidian distance of $\boldsymbol{z}$ from the variety $\mathcal{A}$ remains the same.*

The Sampson distance (1) *thus weighs the points in our sample set Ω more evenly, as it compensates for the norm of the vectors that span the orthogonal complement of $\mathcal{A}$ in z.*

However, minimizing the Sampson distance contains a redundancy. As the ideal that is generated by $Q(\mathbf{x})$ is closed under addition and multiplication with ring elements and all polynomials in Q have the same degree by assumption, its zero set is the same as the zero set of $MQ(\mathbf{x})$ for any regular matrix $M \in \mathbb{R}^{m \times m}$, because . Mutual linear independence between the polynomials in Q is maintained this way. Moreover, the Sampson distance is invariant under nonsingular linear transformation M, so the polynomials in $Q(\mathbf{x})$ that minimize the Sampson distance are not

unique. Therefore, we impose conditions on the coefficients of the polynomials in $Q(\mathbf{x})$. Ma et al. [24] consider the $m \times m$ matrix $\frac{1}{N}\sum_{i=1}^{N} \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T$ for this purpose, as constructed in Proposition 3.7.

Proposition 3.7. *If there are no lower-degree polynomials than those occuring in Q that vanish on Ω, then the matrix $\frac{1}{N}\sum_{i=1}^{N} \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T$ is symmetric positive definite.*

Proof. The above matrix is a real symmetric positive semi-definite matrix by construction. If the product of jacobians $\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T \in \mathbb{R}^{m\times m}$ was rank-deficient, then so would $\mathcal{J}(Q)(z_i)$ be. Consequently, there is a linear relation between the rows of $\mathcal{J}(Q)(z_i)$, i.e. the gradients of the individual polynomials that occur in Q. As each entry of the gradient has a strictly lower degree than its corresponding polynomial, we can construct a lower-degree polynomial that vanishes on z_i. However, assuming that the matrix from the proposition is rank-deficient, there is a singular vector $v \in \mathbb{R}^m \setminus \{0\}$ such that

$$v^T \left(\frac{1}{N}\sum_{i=1}^{N} \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T \right) v = \frac{1}{N}\sum_{i=1}^{N} \underbrace{\left(v^T \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T v\right)}_{\geq 0} = 0,$$

as every individual matrix $J_i = \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T$ is positive semi-definite. For this to hold, each $v^T \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T v$ needs to be 0. Consequently, the matrix J_i is singular, resulting in a linear relation between the rows and by that a lower-degree polynomial that vanishes on Ω. This is in contradiction to our assumption, proving that the average of the jacobian products at the data points Ω has full rank. □

When employing the recursive GPCA algorithm discussed in Section 3.2, assuming that there are no lower-degree polynomials vanishing on Ω is sensible. The lowest-degree polynomials that occur in the vanishing ideal of $\mathcal{A}$ are found first, demonstrating that in each degree there are no lower degree polynomials than the ones currently considered vanishing on the data set.

As Proposition 3.7 shows, the jacobian products' average in the sample points is symmetric positive definite, meaning that it can be diagonalized with strictly positive eigenvalues λ_i. Since $MQ(\mathbf{x})$ has the same vanishing locus as $Q(\mathbf{x})$ for a regular linear transformation M, we can scale the resulting diagonalized jacobian product using a diagonal matrix M with entries $\frac{1}{\sqrt{\lambda_i}}$, maintaining the same vanishing set. Analogous to Ma et al. [24], this procedure's outcome is the identity matrix, leading to the desired constraint

$$\frac{1}{N}\sum_{i=1}^{N} \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T = I_{m\times m},$$

with m denoting the number of polynomials in $Q(\mathbf{x})$. Ultimately, we formulate the following constrained nonlinear optimization problem:

$$Q^\star = \arg\min_{Q} \frac{1}{N}\sum_{i=1}^{N} Q(z_i)^T \left(\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T\right)^\dagger Q(z_i) \tag{2}$$

$$\text{s.t. } \frac{1}{N}\sum_{i=1}^{N} \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T = I_{m\times m} \tag{3}$$

Optimizing the above objective function is possible with one of the myriad nonlinear optimization algorithms, for example an iterative gradient descent technique. However, a sufficiently good initialization is needed for any such method to converge to a global minimum quickly. Finding such an initialization is the next step in the algorithm's development.

Both, the least square fitting error $||Q(z)||^2$ and the constraint (3) are invariant under unitary transformations $R \in \mathbb{R}^{m\times m}$ with $R^TR = I_{m\times m}$, demonstrating there still is a redundancy in the optimization problem (2). Similar to Ma et al. [24], the arithmetic mean of the matrices $\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T$ is required to be $I_{m\times m}$, so we can approximate each of these matrix products with the identity. With this in mind, the sum

$$\frac{1}{N}\sum_{i=1}^{N} Q(z_i)^T Q(z_i) = \frac{1}{N}\sum_{i=1}^{N} ||Q(z_i)||^2$$

approximates the Sampson distance (1). The following proposition then paves the way for an efficient algorithm that finds a good initialization to our nonlinear optimization problem (2).

Proposition 3.8. *Given $C = (c_1, \ldots, c_m)$ and $Q(\boldsymbol{x}) = C^T\nu_n(\boldsymbol{x})$, we can transform the least squares problem*

$$Q^\star = \arg\min_Q \frac{1}{N}\sum_{i=1}^{N} ||Q(z_i)||^2 \quad \textit{subject to} \quad \frac{1}{N}\sum_{i=1}^{N} \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T = I_{m\times m} \tag{4}$$

to an eigenvalue problem of the form

$$\Gamma^{-1}\Sigma c_i^\star = \lambda_i c_i^\star \qquad \textit{for} \quad i = 1, \ldots, m \tag{5}$$

with $C^\star = (c_1^\star, \ldots, c_m^\star)$ representing the optimal solution to the constraint program (4) and matrices given by

$$\Sigma = \frac{1}{N}\sum_{i=1}^{N} \nu_n(z_i)\nu_n(z_i)^T \quad \textit{and} \quad \Gamma = \frac{1}{N}\sum_{i=1}^{N} \nabla\nu_n(z_i)\nabla\nu_n(z_i)^T.$$

Proof. Let $Q = (q_1, \ldots, q_m) = C^T\nu_n(z_i)$. We deduce

$$||Q(z_i)||^2 = \sum_{j=1}^{m} q_j(z_i)^2 = \sum_{j=1}^{m} (c_j^T\nu_n(z_i)\nu_n(z_i)^T c_j) = \operatorname{tr}\left(C^T\nu_n(z_i)\nu_n(z_i)^T C\right).$$

As the trace commutes with addition and scalar-multiplication and $\mathcal{J}(Q)(z_i) = C^T\nabla\nu_n(z_i)$, the constraint programming problem (4) is equivalent to

$$C^\star = \arg\min_C \operatorname{tr}(C^T\Sigma C) \quad \text{subject to} \quad C^T\Gamma C = I_{m\times m}. \tag{6}$$

Subsequently, we use Lagrange multipliers to transform the optimization problem (6) into an eigenvalue problem. To do so, we reformulate (6) as follows:

$$\begin{aligned} C^\star \quad &= \quad \arg\min_C \sum_{j=1}^{m} c_j^T\Sigma c_j \\ &\text{s.t.} \quad C^T\Gamma C - I_{m\times m} = 0. \end{aligned}$$

According to Ma et al. [24], Lagrange multipliers enable us to set up the following loss function with $\Lambda = (\lambda_{i,j})_{i,j=1}^m$:

$$\mathcal{L}(C, \Lambda) = \sum_{j=1}^{m} c_j^T\Sigma c_j - \sum_{i,j=1}^{m} \lambda_{i,j}\left(c_i^T\Gamma c_j - \delta_{i,j}\right).$$

Here, $\delta_{i,j}$ is the Kronecker delta. As the necessary condition for optima requires, $\mathcal{L}$'s derivative

is 0 in optimal solutions. Accordingly,

$$\begin{aligned} 0 = \frac{\partial \mathcal{L}(C,\Lambda)}{\partial c_i} &= 2\Sigma c_i - 2\sum_{j=1}^{m} \lambda_{i,j} \Gamma c_j \qquad \text{for } i = 1,\ \ldots,\ m \qquad \text{and} \\ 0 = \frac{\partial \mathcal{L}(C,\Lambda)}{\partial \lambda_{i,j}} &= c_i^T \Gamma c_j - \delta_{i,j} \qquad\qquad \text{for } i,j = 1,\ \ldots,\ m. \end{aligned}$$

Notice that $c_j^T \Gamma c_i = 0$ for $i \neq j$. As a result, we omit $c_j^T \Gamma c_i$ in the loss function $\mathcal{L}(C,\Lambda)$, made possible by maintaining the mixed constraints. Deriving the new loss function reveals

$$\begin{aligned} 0 &= 2\Sigma c_i - 2\lambda_{i,i} \Gamma c_i \qquad \text{for } i = 1,\ \ldots,\ m \qquad \text{and} \\ 0 &= c_i^T \Gamma c_j - \delta_{i,j} \qquad \text{for } i,j = 1,\ \ldots,\ m. \end{aligned}$$

The first line is a generalized eigenvalue problem. We denote $\lambda_{i,i}$ by λ_i and subsequently consider $0 \leq \lambda_1 \leq \cdots \leq \lambda_m$ as the eigenvalues corresponding to c_i in ascending order. We know that they are non-negative, as both Σ and Γ are symmetric positive semi-definite matrices by construction. With noisy data and no polynomials of degree lower than n vanishing on the subspace arrangement $\mathcal{A}$, Γ has full rank. As Γ is a square matrix, it is invertible and consequently, the generalized eigenvalue problem $\Sigma c_i = \lambda_i \Gamma c_i$ is equivalent to the eigenvalue problem

$$\Gamma^{-1} \Sigma c_i = \lambda_i c_i. \tag{7}$$

A $m \times m$ matrix is diagonizable if and only if there is a basis of m eigenvectors. By Proposition 3.7, the matrix Γ and its inverse Γ^{-1} are symmetric positive definite. Accordingly, the invertible square root $\Gamma^{-\frac{1}{2}}$ of Γ^{-1} exists, enabling us to write

$$\Gamma^{\frac{1}{2}} \Gamma^{-1} \Sigma \Gamma^{-\frac{1}{2}} = \Gamma^{-\frac{1}{2}} \Sigma \Gamma^{-\frac{1}{2}}.$$

The latter expression is symmetric because $\Gamma^{-\frac{1}{2}}$ and Σ are both symmetric, so $\Gamma^{-1}\Sigma$ is similar to a symmetric matrix. As a result, $\Gamma^{-1}\Sigma$ is diagonizable, implying this matrix has m linearly independent eigenvectors. These vectors can be chosen to respect the orthonormality constraints in $C^T \Gamma C = I_{m \times m}$, as proven in the following.

Let c_i and c_j be two eigenvectors from the eigenvalue problem (7) with distinct eigenvalues λ_i and λ_j. By the symmetry of Σ,

$$\lambda_j c_i^T \Gamma c_j = c_i^T \Sigma c_j = c_i^T \Sigma^T c_j = \underbrace{\left(c_j^T \Sigma^T c_i\right)^T}_{\in k} = c_j^T \Sigma c_i = \lambda_i c_j^T \Gamma c_i. \tag{8}$$

Since Γ is symmetric positive definite, it defines an inner product $\langle \text{-}, \text{-} \rangle_\Gamma$ and $c_j^T \Gamma c_i = c_i^T \Gamma c_j$ holds. As the eigenvalues were chosen to be distinct, the equations (8) show $c_j^T \Gamma c_i = 0$. The Gram-Schmidt theorem (cf. Preston [32, p. 158]) demonstrates for eigenvectors with the same associated eigenvalue λ, that we can choose an orthogonal basis of the eigenspace with respect to $\langle \text{-}, \text{-} \rangle_\Gamma$. By normalizing c_i with $\langle c_i, c_i \rangle_\Gamma^{-1}$, a solution $C = (c_1, \ldots, c_m)$ to the eigenvalue problem (7) automatically satisfies the constraint from (4), proving the claim. □

Let $C^\star = (c_1^\star, \ldots, c_m^\star)$ be a solution to the eigenvalue problem (5) in Proposition 3.8. According to Ma et al. [24], the polynomials $Q^\star(\mathbf{x}) = (C_i^\star)^T \nu_n(\mathbf{x})$ lead to a good initialization to the constraint programming problem (2). Typically, a reasonable gradient descent technique only needs a few iterations to converge to a global minimum.

4 Learning Equations of Algebraic Varieties

In Constrast to Chapter 3, which introduces a way to learn polynomials vanishing on a subspace arrangement, in this chapter, these methods are generalized to arbitrary algebraic varieties V. To begin with, existing methods for learning polynomials that approximate sample points under given restrictions are investigated. While the Buchberger-Möller algorithm (cf. Möller and Buchberger [27]) has proven to work well in exact arithmetics (cf. Abbott et al. [1]) and has been extended to noisy data (cf. Heldt et al. [22]), it outputs the ideal of all polynomials vanishing on the samples. The resulting vanishing ideal is necessarily zero-dimensional, ignoring the data's topological and geometric structure. As a first considerable advance in learning polynomials that respect the data's geometry, we mention the article *Learning Algebraic Varieties from Samples* by Breiding et al. [5]. In the following, the algorithm that is introduced in Breiding's paper is reproduced. We extend it in later sections, using the statistical methods Ma et al. and Vidal et al. [24, 35] establish with the generalized principal component analysis (cf. Chapter 3).

4.1 Presentation of the Basic Method

Assume for this section that $\Omega = \{z_1, \dots, z_N\} \subset k^D$ is a noise-free sample set from an algebraic variety V that is a complete intersection. Equivalently, its codimension $\operatorname{codim}(V)$ is equal to the minimum number of generating polynomials. Generally, not all generators of the vanishing ideal $\mathcal{I}(V)$ are homogeneous polynomials, if V is not a projective variety. Thus, we need a different concept than the Veronese embedding to compute the vanishing ideal $\mathcal{I}(\Omega)$. Such a concept is presented in Breiding et al. [5]. For a finite, linearly independent subset $\mathcal{M}$ of $\mathcal{R} = \mathbb{R}[x_1, \dots, x_D]$, we write $U_{\mathcal{M}}(\Omega)$ for the $N \times |\mathcal{M}|$ matrix whose i-th row consists of $\mathcal{M}$'s evaluation in the sample point z_i. To understand $U_{\mathcal{M}}(\Omega)$ better, consider some special choices of $\mathcal{M}$.

Picking $\mathcal{M} = \mathcal{R}_n$, the set of monomials of degree n, results in a matrix L_n that we have already seen in Chapter 3. Recall that L_n's rows consist of the n-th Veronese embeddings of the points in Ω. For the set $\mathcal{M} = \mathcal{R}_{\leq n}$ that consists of the monomials of degree up to n, we call $U_{\mathcal{M}}(\Omega)$ the multivariate Vandermonde matrix. Its name derives from the Vandermonde matrix that is used in various articles on interpolation (cf. Björck and Pereyra [2]). In our setting, it can be realized with $D = 1$. Conveniently, the kernel of $U_{\mathcal{M}}(\Omega)$ equals the finite-dimensional vector space $\mathcal{I}(\Omega) \cap \mathcal{R}_{\mathcal{M}}$. Hence, when the polynomial basis $\mathcal{M}$ is chosen well enough, we can find the vanishing ideal $\mathcal{I}(V)$ of the variety V. The question remains, what a suitable choice of $\mathcal{M}$ is. Without prior knowledge about the variety, it seems reasonable to assume that $\mathcal{R}_{\leq n}$ is a good choice for $\mathcal{M}$. If n is suffciently large, all polynomials in $\mathcal{I}(V)$ can be generated from $\mathcal{I}(\Omega) \cap \mathcal{R}_{\leq n}$. The following propositions helps in quantifying our choice of $\mathcal{M}$.

Lemma 4.1 (5.2 in Breiding et al. [5]). *If the inclusion of $\mathcal{I}(\mathcal{V}) \cap \mathcal{R}_{\mathcal{M}}$ into its superspace $\mathcal{I}(\Omega) \cap \mathcal{R}_{\mathcal{M}} = \ker(U_{\mathcal{M}}(\Omega))$ is an equality, then*

$$|\Omega| = N \geq |\mathcal{M}| - \dim(\mathcal{I}(V) \cap \mathcal{R}_{\mathcal{M}}).$$

Proof. By assumption, $\mathcal{I}(V) \cap \mathcal{R}_{\mathcal{M}} = \ker(U_{\mathcal{M}}(\Omega))$. Since $U_{\mathcal{M}}(\Omega)$ consists of N rows, N is an upper bound on the matrix' rank. The statement of the lemma follows, as the rank of a matrix is equal to the number of columns minus the dimension of its kernel. □

The samples Ω are fixed, leaving $\mathcal{M}$ as the only variable. There are some interesting propositions, both in Breiding et al. [5] and Ma et al. [24], on a dynamic selection of $\mathcal{M}$. Nevertheless, we only deal with a known number of linear independent equations m and known maximal occuring degree n of $\mathcal{I}(V)$'s generating equations. In the special case $\mathcal{R}_{\mathcal{M}} = \mathcal{R}_{\leq n}$, Lemma 4.1 demonstrates that at least as many samples as the difference $\binom{n+D}{n} - \sum_{j=1}^{n} \dim(\mathcal{I}(V)_j)$ are

necessary. This expression has an upper bound given by $\binom{n+D}{n}$. Since the dimension of the ambient space D is fixed, the degree n has to attain a value such that

$$\binom{n+D}{n} - \sum_{j=1}^{n} \dim(\mathcal{I}(V)_j) \;\leq\; \binom{n+D}{D} = \frac{(n+D)\cdots(n+1)}{D!} \;\leq\; N.$$

As no further information about the properties of V is available, we take advantage of the matrix $U_{\mathcal{R}_{\leq n}}(\Omega)$ to calculate those equations by computing its kernel. Breiding et al. [5] presents three methods for finding a kernel: The singular value decomposition SVD, the R from a QR-decomposition or the reduced row echelon form RREF. Making use of either of these algorithms, `FindEquations` has proven to perform well on samples coming from special varieties (cf. Breiding et al. [5]). Nonetheless, the method shows several issues. First and foremost, noise in the data points Ω leads to false results. To understand that claim, consider Figure 5.

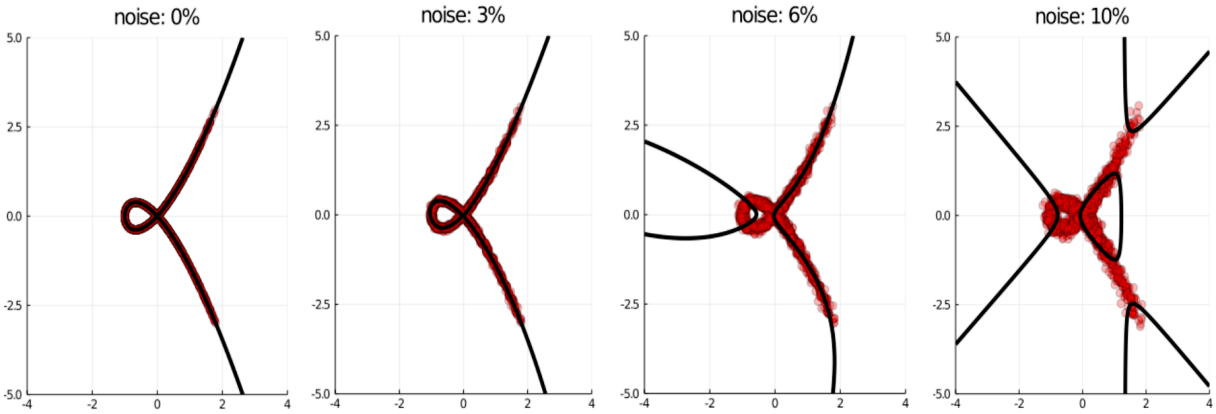

Figure 5: `FindEquations` applied to points sampled from $\langle y^2 - x^3 - x^2 \rangle$ with varying noise.

In this figure, the displayed samples are acquired from the ideal $\langle y^2 - x^3 - x^2 \rangle$ using the rejection sampling technique explained in Breiding and Marigliano [4]. These data points are corrupted with isotropic Gaussian noise ranging from 0 to 10%. The curves depicted in these images are calculated using the algorithm `FindEquations`. Thus, the first picture on the left resembles the original nodal cubic.

Visually, the variety that `FindEquations` learns does not fit the depicted sample points well when the noise is increased. Intuitively, even above 6% Gaussian noise, the variety that optimally fits the data points should closely resemble the nodal elliptic curve, containing a singularity in 0. Even so, `FindEquations` works reasonably well until more than 3% noise is added. This noise threshold is bound to drastically decrease when dealing with more complex systems. In particular, a higher-degree variety in an ambient space with more dimensions than 2 already succeeds in breaking the algorithm, as illustrated in Section 4.6. This observation shows that an algorithm more robust against perturbations is necessary.

A second problem is that the multivariate Vandermonde matrix is severely ill-conditioned. In Breiding et al. [5], methods to increase the algorithm's stability are suggested. Nevertheless, improving the stability of `FindEquations` is not the focus of the thesis, as the following sections avoid the Vandermonde matrix entirely.

Once multiple equations are learned ($m > 1$), sparsity becomes an issue. In real-life applications, we are used to a sparse presentation of polynomial ideal generators. This way, the ideal is vastly more efficient to store. In contrast, the singular value decomposition yields an orthonormal basis for $\mathcal{I}(\Omega) \cap \mathcal{R}_{\leq n}$. Section 5.2 and Example 5.3 in Breiding et al. [5] explain this issue in greater

detail, proposing that the QR decomposition results in a sparse basis. However, in Breiding's paper it is laid out that there is a trade-off between sparsity and the stability of computations. Consequently, the computation of a sparse basis is usually less accurate in the presence of noise than the computation of a dense basis.

4.2 Choosing a Suitable Error Function

As we have seen in the previous section, the curve that `FindEquations` learns does not fit the samples in the presence of noise (cf. Figure 5). It is essential to quantify the result of our algorithm. In particular, an error function is necessary to evaluate how well the learned variety approximates the data points Ω. A naïve approach is to use the mean squared error

$$E_{MS}(Q;\Omega) = \frac{1}{N}\sum_{i=1}^{N} ||Q(z_i)||^2.$$

After all, our goal is that $Q(z_i) = 0$ for each $z_i \in \Omega$, because then the points in Ω lie on the vanishing set generated by the polynomials in Q. Either way, this error function depends on the scaling of Q's coefficients. In Fitzgibbon et al. [16], several scalings are recommended to solve this issue. For example, requiring $c_1 + \frac{1}{2}c_2 + \frac{1}{2}c_3 = 1$, with c_i the i-th column in the coefficient matrix C is recommended. Alternatively, a normalization of the row vectors in C is proposed.

For arbitrary varieties, we have already seen an example in Section 3.3. The loss function E_{MS} does not describe the discrepancy between samples and variety accurately in points with high curvature. In addition, E_{MS} is biased towards methods based on least squares formulations. To overcome this obstacle, we introduce the Sampson distance $\mathcal{L}_S(C;\Omega,n)$ corresponding to

$$E_{SD}(Q;\Omega) = \frac{1}{N}\sum_{i=1}^{N} Q(z_i)^T \left(\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T\right)^{\dagger} Q(z_i).$$

As geometrically elaborated in Section 3.3 and further justified in the following section, E_{SD} is a good fit for our problem's objective function. In demonstrating that our suggested method performs well, using the error E_{SD} is a self-fulfilling prophecy: Any loss function is biased towards an algorithm that optimizes the same objective function.

Instead, taking the difference between the original and the learned variety seems to be a good idea. This is infeasible for several reasons. Firstly, the integral over the difference of two vanishing sets easily becomes infinitely large. Instead, a Monte Carlo approach seems promising. For instance, sampling points y_i and $\hat{y}_i$ on both varieties lying on the same hyperplane and calculating the smallest occuring euclidian distances is a suitable method. Let us assume that there are k and $\hat{k}$ respectively of these points sampled from either variety. This procedure can be repeated for $\ell \in \mathbb{N}$ times, resulting in

$$E_{MC}(Q;\Omega) = \frac{1}{\ell}\sum_{i=1}^{\ell}\frac{1}{\hat{k}}\sum_{\hat{j}=1}^{\hat{k}} \min_{j\in[k]} ||y_{i,j} - \hat{y}_{i,\hat{j}}||^2.$$

Yet, there is no guarantee for the accuracy of this method. A small change in the hyperplane results in a large change in error, as we are dealing with polynomials of degree n. On a further note, we do not necessarily know the original equation in real-world applications. It is sensible to choose an error function that does not depend on prior knowledge of the equations.

Ultimately, the desired error function turns out to be the euclidian distance of the points to the

variety that was learned by the algorithm,

$$E_{dist}(Q;\Omega) = \frac{1}{N}\sum_{i=1}^{N} \operatorname{dist}(z_i,\ \mathcal{V}(Q)).$$

This error is a measure for how well the learned equations fit the given samples. For curves, a method for the error's calculation is laid out in Breiding and Timme [6]. It depends on solving the polynomial system

$$\begin{bmatrix} \det\begin{pmatrix}\mathbf{x}-z_i & \mathcal{J}(Q)(\mathbf{x})^T\\ & Q(\mathbf{x})^T\end{pmatrix} \end{bmatrix} = 0 \tag{9}$$

obtained by attaching the vector $\mathbf{x}-z_i$ to the jacobian $\mathcal{J}(Q)(\mathbf{x})^T$. Each solution of the system (9) is a critical point in the euclidian distance to z_i that lies on the variety $\mathcal{V}(Q)$. The solution x_i with the smallest distance to the point z_i is selected and we calculate the mean distance

$$E_{dist} = \frac{1}{N}\sum_{i=1}^{N} ||z_i - x_i||^2.$$

A solution to the polynomial system (9) can be obtained via `HomotopyContinuation.jl` (cf. Breiding and Timme [7]). Generalizing this approach to arbitrary varieties involves finding the regular points $x_i \in \mathcal{V}(Q)$ such that $z_i - x_i$ is perpendicular to $\mathcal{V}(Q)$'s tangent space $T_{x_i}\mathcal{V}(Q)$ in x_i. The smallest distance direction is given by an orthogonal vector. As already observed, the orthogonal complement of the tangent space in x_i is spanned by the row vectors of the jacobian $\mathcal{J}(Q)(x_i)$. Consequently, what is really done when solving the polynomial system (9) is to check, whether $\mathbf{x}-z_i$ is linearly dependent on the row vectors of $\mathcal{J}(Q)(\mathbf{x})$ for a regular point $\mathbf{x} \in \mathcal{V}(Q)$. In other words, we check whether $\mathbf{x}-z_i$ lies in the orthogonal complement of $T_{\mathbf{x}}\mathcal{V}(Q)$.

Assume that $\mathcal{J}(Q)(\mathbf{x})$ has rank m for a regular point $\mathbf{x} \in \mathcal{V}(Q)$, so $m = \operatorname{codim}(\mathcal{V}(Q))$. As discussed earlier, we want $\mathbf{x}-z_i$ to be perpendicular to $T_{\mathbf{x}}(\mathcal{V}(Q))$. Equivalently, all $(m+1)\times(m+1)$ minors of

$$\begin{bmatrix} \mathbf{x}-z_i \\ \mathcal{J}(Q)(\mathbf{x}) \end{bmatrix}$$

vanish (cf. Draisma et al. [14]). According to Lemma 2.1 from Draisma et al. [14], the variety consisting of all points $\mathbf{x}$, where both Q and all $(m+1)\times(m+1)$ minors of the aforementioned system vanish, is finite for general points z_i. The mean distance E_{dist} is a feasible error function, as it is invariant under scaling, does not depend on prior knowledge of the underlying vanishing ideal and is not biased towards another method. It uses the samples' euclidian distance to the learned variety. If a variety is a good approximation of the samples Ω, the mean distance is small.

4.3 Transferring GPCA to Arbitrary Varieties

Similar to Section 3.3, let the samples $\Omega = \{z_1, \ldots, z_N\}$ be corrupted with isotropic Gaussian noise. That is

$$z_i = \hat{a}_i + n_i \quad \text{for} \quad i = 1,\ \ldots,\ N,$$

with $\hat{a}_i$ a point on V and n_i some random noise. We still assume that the amount of equations m and the highest occuring degree n for a minimum generating system of $\mathcal{I}(V)$ are known. As the presence of noise makes finding vanishing polynomials impractical, we search for m polynomials of degree n, whose vanishing set approximates our samples Ω well.

It is impractical to use the n-th Veronese embedding in this case, because we are not generally

dealing with homogeneous polynomials. We can solve this issue by embedding the samples into $\mathbb{P}_k^D$, using one of the standard affine charts of projective space:

$$\begin{aligned} \iota : k^D &\hookrightarrow \mathbb{P}_k^D \\ (y_1, \ \ldots, \ y_D) &\mapsto [y_1 : \cdots : y_D : 1]. \end{aligned}$$

The coordinate ring of $\mathbb{P}_k^D$ is $k[x_1, \ldots, x_D, x_{D+1}]$. As a result, the standard Veronese embedding of projective space $\nu_n^{\mathbb{P}}(x_1, \ldots, x_{D+1})$ of degree n can be applied again. When choosing an affine patch by mapping $x_{D+1} \mapsto 1$, a vector consisting of all monomials up to degree n in $k[x_1, \ldots, x_D]$ is the outcome. We denote it by $\nu_{\leq n}(\mathbf{x})$. With the elimination monomial order by x_{D+1} and a degree-respecting order on the other variables x_i for the entries of the vector $\nu_n^{\mathbb{P}}$, it is possible to express $\nu_{\leq n}$ in terms of the Veronese embeddings of degree up to n:

$$\nu_{\leq n}(\mathbf{x})^T = \left(\nu_n(\mathbf{x})^T, \ \nu_{n-1}(\mathbf{x})^T, \ \ldots, \ \nu_1(\mathbf{x})^T, \ 1\right).$$

Remark 4.2. *Conveniently, both vectors $\nu_n^{\mathbb{P}}$ and $\nu_{\leq n}$ have the same length by construction. This is due to the fact that the dimensions of the vector space $k[x_1, \ldots, x_D]_{\leq n}$ agrees with the dimension of $k[x_1, \ldots, x_D, x_{D+1}]_n$.*

While enabling us to work in a projective setting, the embedding ι makes it possible to calculate polynomials for samples in affine space. Initially, assume $\Omega \subset k^D$. The polynomials that generate the vanishing ideal $\mathcal{I}(\Omega)$ are denoted by $Q(\mathbf{x}) = C^T \nu_{\leq n}(\mathbf{x})$. Analogous to Section 3.3, we can set up a constraint programming problem

$$\begin{aligned} Q^\star \quad &= \quad \arg\min_Q \frac{1}{N} \sum_{i=1}^N Q(z_i)^T \left(\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T\right)^\dagger Q(z_i) \\ &s.t. \quad \frac{1}{N} \sum_{i=1}^N \mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T = I_{m\times m} \end{aligned}$$

for $z_i \in \Omega$. Subsequently, we intend to employ the eigenvalue problem introduced in Proposition 3.8. With it, we want to find a good initialization for the above nonlinear optimization problem, as done in Section 3.3. However, there is an issue with this approach: Since $\nu_{\leq n}$ contains a constant term, the last column of $\nabla\nu_{\leq n}(\mathbf{x})$ is entirely zero. For this reason, the last row and column of $\nabla\nu_{\leq n}(z_i)\nabla\nu_{\leq n}(z_i)^T$ is zero for each $i \in \{1, \ldots, N\}$. As a result, $\Gamma = \sum_{i=1}^N \nabla\nu_{\leq n}(z_i)\nabla\nu_{\leq n}(z_i)^T$ is not invertible, revealing we cannot convert the generalized eigenvalue problem to a regular eigenvalue problem.

By embedding Ω into $\mathbb{P}_k^D$ via ι, this issue can be solved. Using the n-th Veronese embedding $\nu_n^{\mathbb{P}}$ of $\mathbb{P}_k^D$ instead of $\nu_{\leq n}$ on k^D, we are in the same case as in the generalized principal component analysis introduced in Section 3.3. Accordingly, for the new array of polynomials $P(\mathbf{x}) = C^T \nu_n^{\mathbb{P}}(\mathbf{x})$, we can set up the same constraint program as was done in the nonlinear optimization problem (2) with the samples $\iota(\Omega) = (\iota z_1, \ldots, \iota z_N)$. Namely,

$$P^\star \quad = \quad \arg\min_P \frac{1}{N} \sum_{i=1}^N P(\iota z_i)^T \left(\mathcal{J}(P)(\iota z_i)[\mathcal{J}(P)(\iota z_i)]^T\right)^\dagger P(\iota z_i) \tag{10}$$

$$\text{subject to } \frac{1}{N} \sum_{i=1}^N \mathcal{J}(P)(\iota z_i)[\mathcal{J}(P)(\iota z_i)]^T = I_{m\times m}. \tag{11}$$

For initializing this nonlinear optimization problem, we can finally employ the eigenvalue problem

$$\Gamma^{-1}\Sigma c_i^\star = \lambda_i c_i^\star \tag{12}$$

$$\text{with} \quad \Sigma = \frac{1}{N}\sum_{i=1}^{N} \nu_n^{\mathbb{P}}(\iota z_i)\nu_n^{\mathbb{P}}(\iota z_i)^T \quad \text{and} \quad \Gamma = \frac{1}{N}\sum_{i=1}^{N} \nabla\nu_n^{\mathbb{P}}(\iota z_i)\nabla\nu_n^{\mathbb{P}}(\iota z_i)^T$$

that is discussed in Section 3.3. The eigenvalue problem (12) is a sensible choice for an initialization, as proven in Proposition 3.8. The Veronese embedding $\nu_n^{\mathbb{P}}$ guarantees that Γ is invertible, if there are no polynomials of degree strictly smaller than n vanishing on V, analogous to Ma et al. [24].

Remark 4.3. *This small derivation from Section 4.1 enables us to simultaneously treat projective and affine varieties. While for affine varieties the above procedure is used, for projective varieties we can employ the nonlinear optimization problem* (10) *without embedding the samples in advance.*

Remark 4.4. *The transformation from projective to affine variety does not follow by mapping the last coordinate to 1. The programming language* `Julia` *implicitly chooses a monomial order in* $\nu_n^{\mathbb{P}}$ *that does not respect the explicit choices discussed in this section. When we want to compute the vanishing ideal of an affine variety, we need to be aware of this. Before calculating the final output* Q*, the entries in* C *are rearranged accordingly.*

All of the above notions let us to present an algorithm. It computes the equations generating a vanishing set from noisy data points drawn from an algebraic variety.

Algorithm 1: LearnVanishingIdeal(Ω,m,n)

Input: A poissbly noisy set of samples $\Omega \subset \mathbb{P}_k^D$, the codimension of the underlying variety m and the highest allowed degree n.
Output: Array of Polynomials $Q = (q_1, \ldots, q_m)$ of degrees at most n that fit Ω well.

```
initialization
startEigen = findStartValues(Ω,m,n)

lossEigen = sampsonDistance(Ω, m, n, C, startEigen)
outputEigen = gradientDescent(lossEigen, C, startEigen, m)

return outputEigen'*vectorOfMononials(x,n)
```

In the initialization, two polynomial variables `x` and `C` are instantiated and in the affine case, the samples are embedded into projective space via ι. Additionally, the method `vectorOfMononials` creates $\nu_n^{\mathbb{P}}$. Afterwards, an array of start values `startEigen` is calculated by solving the eigenvalue problem (12). The Sampson distance (cf. Section 3.3) is then used to generate the loss functions `lossEigen`.

There is a fundamental difficulty in finding a pseudoinverse for a matrix with polynomial coefficients in `C`, namely $\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T$. In a general matrix ring, inverse elements do not necessarily exist. Only after having evaluated $\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T$, the pseudoinverse is meaningful. This problem is addressed in Sampson [33] and resolved as follows: When calculating the loss function, the initial values of `C` are inserted into the product of jacobians $\mathcal{J}(Q)(z_i)\,[\mathcal{J}(Q)(z_i)]^T$ for each $z_i \in \Omega$. Consequently, this procedure produces a constant weighting in the loss function depending on the start values. For that reason, the loss function (10) is a polynomial of degree 2 in C's entries. Thereafter, an adaptive `gradientDescent` is used to find a minimum of the

Sampson distance. One gradient descent update is given by the equation

$$C_{t+1} = C_t - \gamma_t \nabla_C F(C_t),$$

with C_t as the current value of the coefficient matrix C and γ_t a scalar that is adaptively chosen with respect to the rate of change. F is the Sampson distance calculated before. The resulting configuration of `C` is then multiplied with $\nu_n^{\mathbb{P}}(\mathbf{x})$ and returned.

One may wonder, whether the problem of `FindEquations` not fitting noisy samples well displayed in Figure 5 is solved by this method. A quantitative approach is provided in a later section, but when even the simple problem of finding a planar elliptic curve cannot be solved by the presented algorithm, the approach is not promising. In Figure 6, a comparison between the method `FindEquations` from Section 4.1 and our method `LearnVanishingIdeal` is presented.

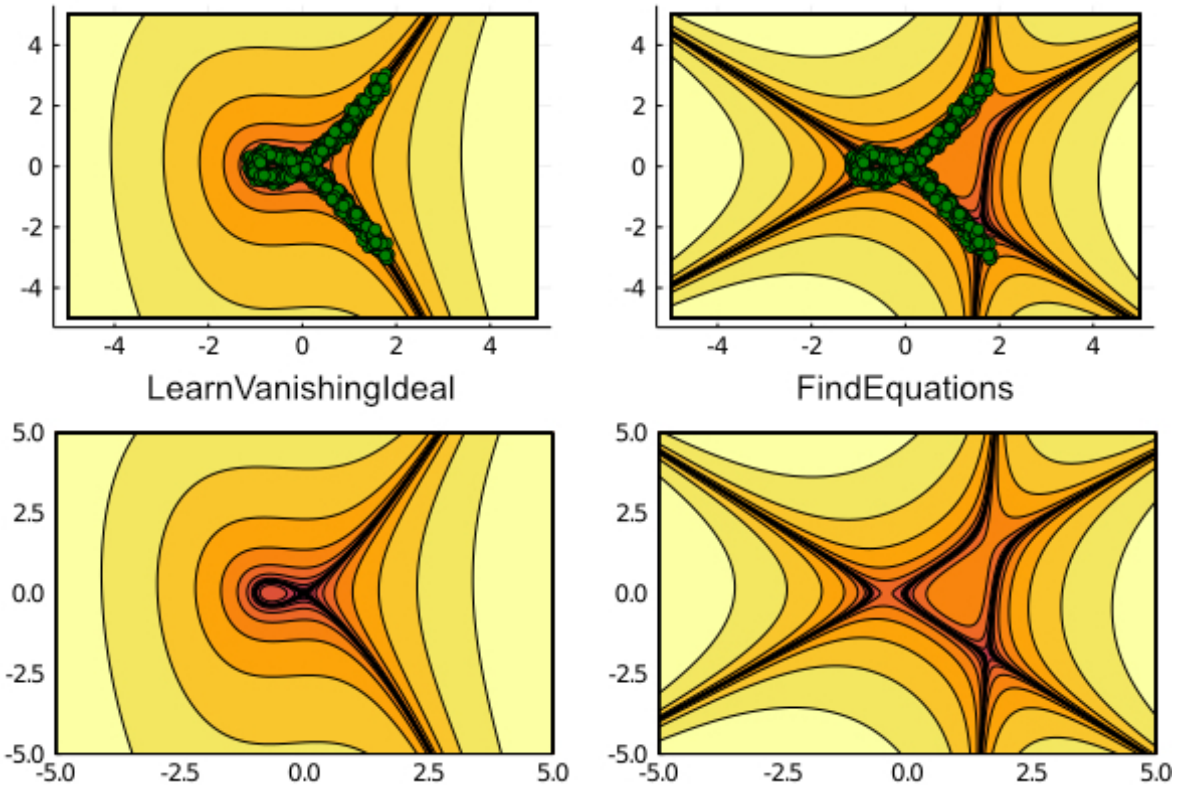

Figure 6: Comparative contour plots of the two methods' results.

In a similar vein as in Section 4.1, a sample set from the ideal $(y^2 - x^2 - x^3)$ is used with 7% Gaussian noise added. The polynomial on the left of Figure 6 is derived using the method `LearnVanishingIdeal` introduced in this chapter and the polynomial on the right is derived using the method `FindEquations`. The log squared error $\log(q(x,y)^2)$ of the learned polynomial q is plotted as contour lines. While the top image includes the scattered data points Ω, the bottom image simply contains the contour plot. The darker the region, the smaller is $\log(q(x,y)^2)$ and hence, the closer q is to 0. The logarithm helps to distinguish values close to 0.

We can deduce that the vanishing locus on the left fits the data points Ω better than the polynomial derived via the method `FindEquations`. Even the singularity of the original elliptic curve is recovered. As already mentioned, this behaviour is quantified with experiments in Section 4.6 and raises hope for useful results, even in the presence of noise.

As a final remark, the sample points in the images on the right on average lie in darker regions.

This is related to observations about the Sampson distance explained in Section 3.3: The mean squared error is smaller on the right variety. For this reason, it is apparently not a good measure for the approximation's quality, because the variety on the left fits the data points better. With the mean euclidian distance, a more suitable error function is described in Section 4.2.

4.4 Variations of the Algorithm

A question that immediately arises after having suggested an algorithm addresses its performance. To answer the question, it is sensible to choose more than the smallest m singular values from the eigenvalue approach (12). Possibly, the best initialization for the nonlinear optimization problem (10) is not among the vectors corresponding to the smallest singular values. Higher confidence about the choice of initialization can be obtained by using all vectors corresponding to singular values within a margin of the m-th smallest singular value. We use $\tau \cdot \lambda_m$ as a threshold, such that all singular values $\lambda_i \leq \tau \cdot \lambda_m$ are considered as start values for the iteration. Subsequently, we pick all m-element subsets of these singular vectors to create distinct m-element combinations that are all fed into the gradient descent algorithm. The optimized combination with the smallest error (cf. Section 4.2) is then returned.

Especially in noise-free cases, the method `FindEquations` performs solidly, in some cases even better than the proposed method `LearnVanishingIdeal`. We try to make use of this behavior by utilizing the multivariate Vandermonde matrix to find start values for the optimization problem (10). With the start value combinations from the previous paragraph, we find the configuration with the smallest error. In doing so, both, the start values and the outcome of the iteration are considered. According to Breiding et al. [5], this idea combats one of the issues from Section 3.1. Without a priori knowledge about the true number and maximal degree of the equations, we can use the ordered singular values to guess how many equations are necessary to fit the data. As laid out in Ma et al. [24], the knee point of the singular values' graph from the eigenvalue approach (12) is much sharper compared to the singular values' graph of the multivariate Vandermonde matrix. This suggests we should base the amount of singular values on the eigenvalue approach, rather than the Vandermonde approach.

Another improvement can be obtained by reusing the output of the gradient descent method and calculating the Sampson distance, with the prior output as new initialization. Since the output is now closer to the original vanishing locus, the new loss function is an even better approximation of the sample points' true distance from the variety. This loss function will then be used to perform another gradient descent step, leading to the final result.

These ideas are implemented in the advanced algorithm `LearnVanishingIdeal` on the next page. Here, the variable `iter` corresponds to the number of times the fine-tuning of the loss function is performed. We propose `iter` $= 2$, as our experiments already promise good results after improving the Sampson distance once. The attentive reader might have noticed the threshold variable τ. As mentioned earlier in this section, our intention is to consider more than m singular vectors as initialization for the iteration. Thus, we pick all start values with singular values within the threshold $[0, \tau \cdot \lambda_m]$ for the m-th smallest singular value λ_m from the eigenvalue problem (12). A suitable pick is $\tau = 2.0$, as this promises to maintain a good balance between performance and complexity in the algorithm. Moreover, we pick the same number of smallest singular values from the Vandermonde system (cf. Section 4.1). Iterating through all m-element subsets of the start values, a gradient descent technique is applied to the Sampson distance (10), initialized with the aforementioned combination of start values. The coefficient vectors `intermediateValues` with the smallest error is returned. Suitable choices for `error` are discussed in Section 4.2.

While significantly improving the algorithm's accuracy, the above variations share one drawback: They worsen the algorithm's already considerably long runtime. In the `Julia` programming

language, most of the involved complexity derives from polynomial manipulations, such as multiplication and differentiation. The involved matrix operations' runtime is negligible compared to the polynomial operation's complexity, taking into account that the matrices' dimensions are bounded above by N, m and the dimension of the search space $\binom{n+D}{n}$. All of these numbers are insignificant in terms of modern linear algebra when used alongside polynomial operations.

Algorithm 2: LearnVanishingIdeal(Ω,m,n,iter,τ)

Input: A possibly noisy set of samples $\Omega \subset \mathbb{P}_k^D$,
the underlying variety's codimension m,
the highest allowed degree n,
the number of iterations `iter` for each configuration of start values,
a threshold τ for choosing the amount of considered singular values.
Output: Array of polynomials $Q = (q_1, \ldots, q_m)$ of degrees at most n that fit Ω well.

```
initialization
startEigen, startVander = findStartValuesUpToThreshold(Ω,m,n,τ)
combinations = [makeCombinations(startEigen,m), makeCombinations(startVander,m)]

globalerror = ∞
for combination in combinations do
    for startValues in combination do
        if error(startValues) < globalError then
            outputValues = startValues

        intermediateValues = startValues
        for i in 1:iter do
            loss = sampsonDistance(Ω, m, n, C, intermediateValues)
            intermediateValues = gradientDescent(loss, C, intermediateValues, m)

            if error(intermediateValues) < globalError then
                outputValues = intermediateValues
        end
    end
end

return outputValues'*vectorOfMonomials
```

Our goal in the following is to implement methods to algebraically find both the Sampson distance and its derivative for a given value of the coefficient matrix C. This way, polynomial manipulations are completely avoided. While the Sampson distance itself is needed to calculate an approximation of the current loss in the gradient descent algorithm, its derivative is necessary to determine the direction of the following optimization step. The Sampson distance (10) itself only depends on the coefficient matrix C, as the sample points Ω are fixed. When previously injecting the current value of the coefficient matrix C, computing the value of the loss function is devoid of polynomial operations.

Calculating the derivative algebraically is harder, as a scalar-by-matrix derivative needs to be computed. To achieve that, let us first define what is meant when talking about deriving a scalar function by a matrix:

Definition 4.5. *Let $\Phi : k^{p\times q} \to k$ be a scalar function in a $p \times q$ matrix $\boldsymbol{X} = (x_{ij})_{i=1,j=1}^{p,q}$ of independent variables x_{ij}. The derivative of Φ with respect to $\boldsymbol{X}$ is given by*

$$\frac{\partial \Phi}{\partial \boldsymbol{X}} = \begin{pmatrix} \frac{\partial \Phi}{\partial x_{11}} & \frac{\partial \Phi}{\partial x_{12}} & \cdots & \frac{\partial \Phi}{\partial x_{1q}} \\ \frac{\partial \Phi}{\partial x_{21}} & \frac{\partial \Phi}{\partial x_{22}} & \cdots & \frac{\partial \Phi}{\partial x_{2q}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \Phi}{\partial x_{p1}} & \frac{\partial \Phi}{\partial x_{p2}} & \cdots & \frac{\partial \Phi}{\partial x_{pq}} \end{pmatrix}.$$

Unfortunately, exact equivalents of the familiar chain rule and the product rule from multivariate calculus do not persist when dealing with matrix-valued functions. Even so, the following proposition is essential in algebraically calculating the Sampson distance's derivative.

Proposition 4.6. *Let $\boldsymbol{X}$ be a $p \times q$ matrix of independent variables. Let $\boldsymbol{A} \in k^{r\times p}$, $\boldsymbol{B} \in k^{q\times q}$ and $\boldsymbol{C} \in k^{p\times r}$. Then the following equality holds:*

$$\frac{\partial \operatorname{tr}\left(\boldsymbol{A}^T \boldsymbol{X} \boldsymbol{B} \boldsymbol{X}^T \boldsymbol{C}\right)}{\partial \boldsymbol{X}} = \boldsymbol{C}\boldsymbol{A}^T \boldsymbol{X}\boldsymbol{B} + \boldsymbol{A}\boldsymbol{C}^T \boldsymbol{X}\boldsymbol{B}^T$$

Proof. The product rule does not apply to our scalar-by-matrix derivative. However, it does apply to differential forms d (cf. Minka [26, p. 2]) and it is possible to convert from the differential to the derivative form. According to Giles [20, p. 4], the equalities

$$d(f(\mathbf{X}) \cdot g(\mathbf{X})) = d(f(\mathbf{X})) \cdot g(\mathbf{X}) + f(\mathbf{X}) \cdot d(g(\mathbf{X})), \quad d(\mathbf{B}) = 0 \text{ for } B \text{ not depending on } \mathbf{X}$$
$$\text{and} \quad d(f(\mathbf{X}) + g(\mathbf{X})) = d(f(\mathbf{X})) + d(g(\mathbf{X}))$$

apply to scalar functions f and g. Moreover, a differential expression $dy = \operatorname{tr}(\mathbf{B}d(\mathbf{X}))$ can be transformed to

$$\frac{\partial y}{\partial \mathbf{X}} = \mathbf{B}^T$$

according to Minka [26, p. 4]. Any differential operator commutes with transpositions and a matrix' trace, the latter of which is invariant under matrix transposition, cyclic permutation and commutes with addition. All of these basic properties can be looked up in Petersen and Pedersen [3, pp. 6-8]. With all of this in mind, we can form the following chain of equalities:

$$\begin{aligned} d\left(\operatorname{tr}\left(\mathbf{A}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{C}\right)\right) &= \operatorname{tr}\left(d\left(\mathbf{A}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{C}\right)\right) = \operatorname{tr}\left(d\left(\mathbf{A}^T\mathbf{X}\mathbf{B}\right)\mathbf{X}^T\mathbf{C} + \mathbf{A}^T\mathbf{X}\mathbf{B}d\left(\mathbf{X}^T\mathbf{C}\right)\right) \\ &= \operatorname{tr}\left(d\left(\mathbf{A}^T\mathbf{X}\mathbf{B}\right)\mathbf{X}^T\mathbf{C}\right) + \operatorname{tr}\left(\mathbf{A}^T\mathbf{X}\mathbf{B}d\left(\mathbf{X}^T\mathbf{C}\right)\right) \\ &= \operatorname{tr}\left(\mathbf{C}\mathbf{A}^T d(\mathbf{X})\mathbf{B}\mathbf{X}^T\right) + \operatorname{tr}\left(\mathbf{C}\mathbf{A}^T\mathbf{X}\mathbf{B}d(\mathbf{X})^T\right) \\ &= \operatorname{tr}\left(\mathbf{B}\mathbf{X}^T\mathbf{C}\mathbf{A}^T d(\mathbf{X})\right) + \operatorname{tr}\left(\left(\mathbf{C}\mathbf{A}^T\mathbf{X}\mathbf{B}d(\mathbf{X})^T\right)^T\right) \\ &= \operatorname{tr}\left(\mathbf{B}\mathbf{X}^T\mathbf{C}\mathbf{A}^T d(\mathbf{X}) + \mathbf{B}^T\mathbf{X}^T\mathbf{A}\mathbf{C}^T d(\mathbf{X})\right) \\ &= \operatorname{tr}\left(\left(\mathbf{B}\mathbf{X}^T\mathbf{C}\mathbf{A}^T + \mathbf{B}^T\mathbf{X}^T\mathbf{A}\mathbf{C}^T\right)d(\mathbf{X})\right). \end{aligned}$$

Together with the conversion rule for differential forms, the equations demonstrate

$$\frac{\partial \operatorname{tr}\left(\mathbf{A}^T\mathbf{X}\mathbf{B}\mathbf{X}^T\mathbf{C}\right)}{\partial \mathbf{X}} = \mathbf{A}\mathbf{C}^T\mathbf{X}\mathbf{B}^T + \mathbf{C}\mathbf{A}^T\mathbf{X}\mathbf{B},$$

concluding the proof. □

Ultimately, we can express the Sampson distance's derivative in a closed form, as $\nu_n^{\mathbb{P}}(z_i) \in k^{\binom{n+D}{n}}$, $\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T \in k^{m\times m}$ and the coefficient matrix C is a $\binom{n+D}{n} \times m$ matrix of independent variables. Hence, Proposition 4.6 applies and the Sampson distance $\mathcal{L}_S(C; \Omega, n)$'s derivative has

the form

$$\frac{\partial \mathcal{L}_S(C;\Omega,n)}{\partial C} = \frac{2}{N}\sum_{i=1}^{N} \nu_n^{\mathbb{P}}(z_i)\left(\nu_n^{\mathbb{P}}(z_i)^T C\right)\left(\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T\right)^{\dagger}.$$

It needs to be stressed that the product of jacobians is symmetric by Proposition 3.7 and that $\nu_n^{\mathbb{P}}(z_i)\nu_n^{\mathbb{P}}(z_i)^T$ is a $\binom{n+D}{n} \times \binom{n+D}{n}$ matrix. We use the associativity of matrix multiplication to reduce the memory the algorithm occupies. If the jacobian $\mathcal{J}(Q)(z_i)$ has a row rank deficiency, the variety $\mathcal{V}(Q)$ has a singularity in z_i. Regardless, this phenomenon occurs with probability 0, if Q does not contain polynomials of degree at most 0. We can filter out singular points from Ω, while maintaining the model's expressiveness. This way, we can to assume that all sample points z_i are regular. This algorithm's final improvement enables it to run at reasonable speed.

4.5 Combatting the Current Overfitting

In the previous sections, we witnessed the development of the algorithm `LearnVanishingIdeal` for learning vanishing ideals from noisy samples. Nonetheless, it still has issues; even after the improvements discussed in Section 4.4. Among them is the struggle of overfitting, i.e. having more parameters than can be justified by the data. Currently, we are trying to learn m equations of the same degree n. Our algorithm exploits the available parameters and potentially learns a higher-degree variety than intended. Consequently, our algorithm presently learns the wrong polynomials when the generators of $\mathcal{I}(V)$ have various degrees.

With this in mind, how can we make sense of the assumption that there are no polynomials of degree strictly smaller than n vanishing on the samples in this general setting. After all, the vanishing ideal of a variety generally contains generators of various degrees.

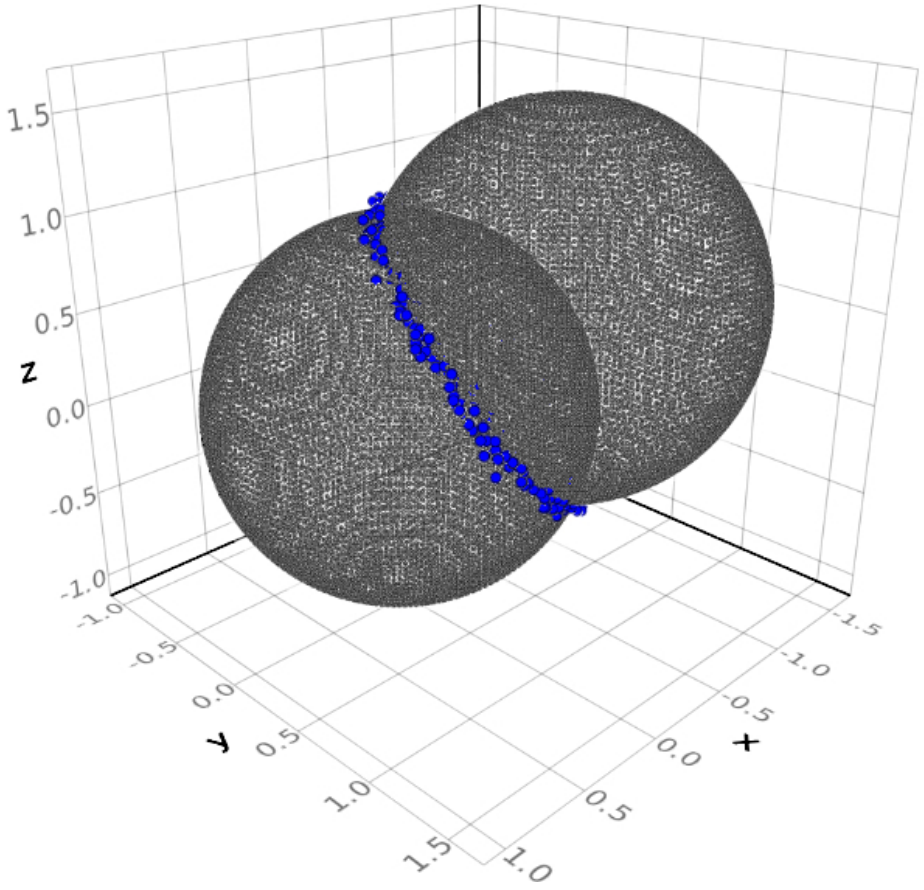

Figure 7: Sample points from the intersection of two unit spheres.

It seems reasonable to apply an approach similar to the recursive GPCA algorithm from Section 3.2, for example by finding the smallest degree n that fits the samples. However, a higher-degree variety approximates the data points better and consequently has a smaller error. In statistical literatur, this behavior is known as overfitting (cf. Dietterich [13]). When the degree is sufficiently high, it is possible that the learned equations fit our data points perfectly, even in the presence of noise. In particular, a line through the origin for each sample point (cf. Ma et al. [24]) approximates the samples Ω well in any reasonable metric, though in most cases this result is not desirable.

To understand why overfitting occurs, it should be stressed that there is an inherent ambiguity in our problem to find vanishing equations. Assume we are given sample points from the intersection of two unit spheres in $\mathbb{R}^3$ whose center points are 1 apart, as depicted in Figure 7. The result of the two spheres' intersection is a degree 2 curve that can be obtained from the intersection of a plane with a quadric.

Even though both, the intersection of two quadrics and a quadric with a plane, yield equivalent vanishing loci, the presence of noise complicates things. Finding a plane and a quadric leads to a degree 2 curve, which is the desired result. Despite that, using two quadrics our algorithm `LearnVanishingIdeal` returns a degree 4 curve. As discussed earlier, this space curve improves the sample points' approximation. However, this is not the vanishing locus we are looking for, as a variety of degree 4 can contain two irreducible components of individual degree 2. Our method `LearnVanishingIdeal` learns such a variety, as depicted in Figure 8.

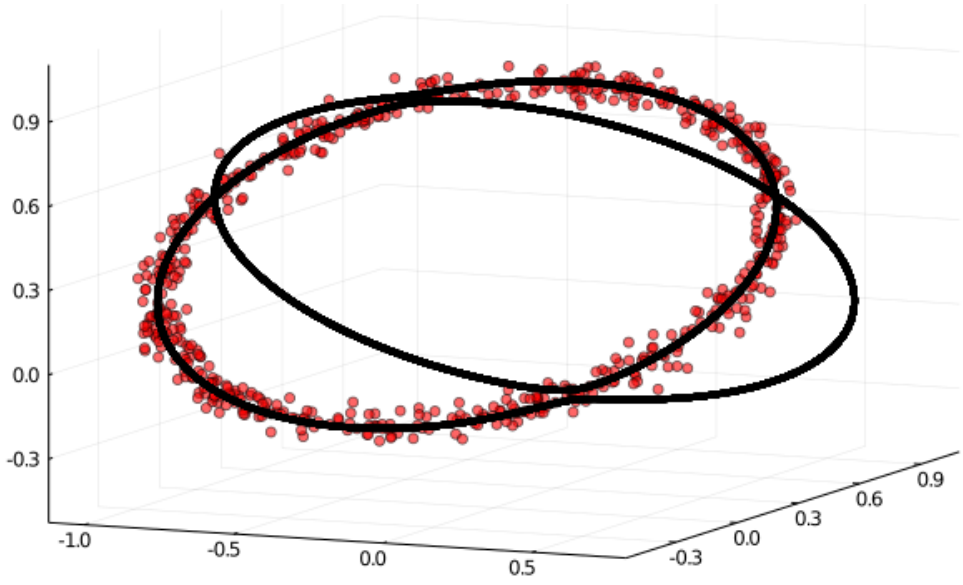

Figure 8: Vanishing set learned by `LearnVanishingIdeal` from the samples in Figure 7.

Since the current search space allows for two quadrics, our algorithm finds two such polynomials. To prevent that, we provide the algorithm with an increasing sequence $n_1 \leq \cdots \leq n_m$ of the vanishing ideal $\mathcal{I}(V)$'s generators' degrees. Correspondingly, we calculate the amount of times m_i each degree appears. After potentially embedding the data points Ω into projective space, for each unique degree n_i we solve the eigenvalue problem (12). This involves using the Veronese embedding $\nu_{n_i}(\mathbf{x})$ of degree n_i. The results of this approach are m_i coefficient vectors c_j of length $\binom{n_i+D-1}{n_i}$, presenting the polynomial vector Q in a different way:

$$Q(\mathbf{x}) = \left(c_1^T \nu_{n_1}^{\mathbb{P}}(\mathbf{x}), \ldots, c_m^T \nu_{n_m}^{\mathbb{P}}(\mathbf{x})\right).$$

Deriving this vector of polynomials Q, computes its jacobian:

$$\mathcal{J}(Q)(\mathbf{x}) = \begin{pmatrix} c_1^T \nabla \nu_{n_1}^{\mathbb{P}}(\mathbf{x}) \\ \vdots \\ c_m^T \nabla \nu_{n_m}^{\mathbb{P}}(\mathbf{x}) \end{pmatrix} \in k^{m \times D}.$$

Abbreviating the list of coefficient vectors by $\mathfrak{C} = (c_1, \ldots, c_m)$ and the corresponding list of degrees by $\underline{\mathbf{n}} = (n_1, \ldots, n_m)$ presents an alternative formulation of the Sampson distance:

$$\mathcal{L}_{aS}(\mathfrak{C}; \Omega, \underline{\mathbf{n}}) = \frac{1}{N} \sum_{i=1}^{N} Q(z_i)^T \left(\mathcal{J}(Q)(z_i)[\mathcal{J}(Q)(z_i)]^T \right)^{\dagger} Q(z_i).$$

Algebraically, the derivative of the adjusted Sampson distance $\mathcal{L}_{aS}$ is

$$\begin{aligned} \frac{\partial \mathcal{L}_{aS}(\mathfrak{C}; \Omega, \underline{\mathbf{n}})}{\partial c_i} &= \frac{2}{N} \sum_{j=1}^{N} Q(z_j)^T \cdot \left(\mathcal{J}(Q)(z_j)[\mathcal{J}(Q)(z_j)]^T \right)^{\dagger} \frac{\partial Q(z_j)}{\partial c_i} \\ &= \frac{2}{N} \sum_{j=1}^{N} Q(z_j)^T \cdot \left(\mathcal{J}(Q)(z_j)[\mathcal{J}(Q)(z_j)]^T \right)^{\dagger} \begin{pmatrix} \partial c_1^T \nu_{n_1}^{\mathbb{P}}(z_j) \,/\, \partial c_i \\ \vdots \\ \partial c_m^T \nabla \nu_{n_m}^{\mathbb{P}}(z_j) \,/\, \partial c_i \end{pmatrix} \\ &= \frac{2}{N} \sum_{j=1}^{N} \left(Q(z_j)^T \cdot \left(\mathcal{J}(Q)(z_j)[\mathcal{J}(Q)(z_j)]^T \right)^{\dagger} \right) \cdot \Xi_{n_i}(z_j)^T, \end{aligned}$$

with

$$\Xi_{n_i}(z_j) = \left(\mathbf{0},\ \ldots,\ \mathbf{0}\ ,\ \nu_{n_i}^{\mathbb{P}}(z_j),\ \mathbf{0},\ \ldots,\ \mathbf{0} \right) \ \in\ k^{\binom{n_i+D-1}{n_i} \times m}.$$

The equalities involve results from multivariate calculus, such as the product rule and the associativity of matrix multiplication. In such manner, the algorithm's memory usage is enhanced. In the implementation of `LearnVanishingIdeal`, we can use the derivatives of $\mathcal{L}_{aS}(\mathfrak{C}; \Omega, \underline{\mathbf{n}})$ to calculate a gradient descent step for the nonlinear optimization problem (10) via

$$\mathfrak{C}_{t+1} = \mathfrak{C}_t - \gamma_t \left(\frac{\partial \mathcal{L}_{aS}(\mathfrak{C}_t; \Omega, \underline{\mathbf{n}})}{\partial c_1}^T,\ \ldots,\ \frac{\partial \mathcal{L}_{aS}(\mathfrak{C}_t; \Omega, \underline{\mathbf{n}})}{\partial c_m}^T \right).$$

The parameter γ_t is adaptively chosen with respect to the loss function's stagnation. Bear in mind that $\mathfrak{C}$ is a list of vectors and should not be considered to be a matrix. Naturally, embedding both c_i and $\nu_{n_i}(z_j)$ into the top-dimensional search space where c_m lives, $\mathfrak{C}$ becomes a matrix. This choice is reasonable in the implementation, as it simplifies computations.

4.6 Comparison of the Presented Methods

According to Chapter 4, the newly proposed `LearnVanishingIdeal` performs at least as well as comparable methods for finding polynomial equations from point clouds. To quantify this assertion, several error functions are discussed in Section 4.2. Primarily, the mean euclidian distance, calculated by solving a polynomial system, is an objective measure for the approximation's quality. As secondary measures, the Sampson distance, the mean squared error and the algorithm's runtime are computed. Denote by `LVI_List` the method that uses a list of degrees to approximate generators of the vanishing ideal (cf. Section 4.5), while `LVI_Max` uses only the maximal occuring degree (cf. Section 4.4).

Example 4.7. Our first example comes from Section 4.3. It is displayed in Figure 5 how the method `FindEquations` reacts to noise. Assume we are given 1517 samples from the nodal curve V_1 with vanishing ideal $\langle y^2 - x^3 - x^2 \rangle$. Isotropic Gaussian noise in a range of 6% to 20% is added. In this case, the ideal can be generated with only one polynomial. Therefore, both `LVI` methods are equivalent. For a comparison between these algorithms and `FindEquations` from Breiding et al. [5], consider Figure 9. The methods `LVI_List` and `LVI_Max` are equivalent in this instance, because $\mathcal{I}(V_1)$ can be generated by one polynomial.

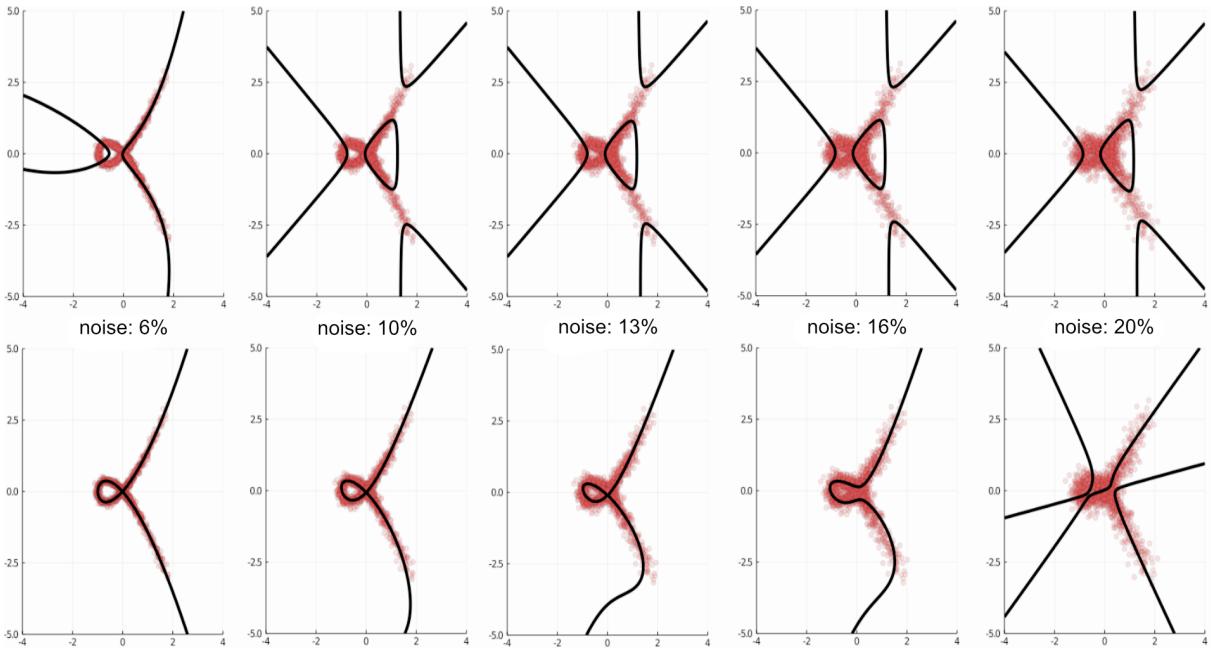

Figure 9: Comparison of `FindEquations` (top row) and `LearnVanishingIdeal` (bottom row).

We do not begin with 0% noise as in Section 4.1, because `FindEquations` only starts to break down at around 6% noise. As mentioned, we can expect our method to perform at least as well as `FindEquations` by construction. In the first three pictures of the bottom row it becomes visible that the curve is phenomenally close to the samples. Even the singularity of the nodal curve is recovered by the algorithm `LearnVanishingIdeal`. Especially when compared to the performance of `FindEquations`' curves in the top row, our method's advantages become clear.

Nevertheless, our method starts to disintegrate between 13% and 16% of noise. The learned curve is still adequately close to both the nodal curve and the samples. Outside the quadrangle $[-3, 3]^2$ the samples lie in, the curve is not recovered well. At 20% noise, `LearnVanishingIdeal` is no longer capable of retrieving the geometry of the nodal curve. Only its two rays are recovered. However, with this much noise it becomes increasingly difficult to recognize patterns in the samples, as the correlation of the data decreases. Table 1 presents a survey how the methods perform with 7% of noise. The respective numbers are rounded to three significant digits.

It can be observed that `FindEquations` outperforms our method `LearnVanishingIdeal` in two categories, namely the mean squared error and the consumed time. Two factors play a role in this: Firstly, our implementation of `FindEquations` is based on the singular value decomposition which is biased towards a least squares fomulation. Secondly, our method is inherently more complex than the method `FindEquations`. Using a regular computer, our method takes significantly longer to terminate than comparable methods. Nonetheless, the goal of this thesis is to find a robust algorithm. While desirable, a complexity-focused optimization of the presented method is reserved for future work.

	`FindEquations`	`LearnVanishingIdeal`
Mean Squared Error	0.0158	0.0235
Sampson Distance	0.0293	0.00432
Mean Distance	0.146	0.0542
Consumed Time	$0.039s$	$6.28s$

Table 1: Comparison of the algorithms using the vanishing ideal $\langle y^2 - x^3 - x^2 \rangle$.

Regardless, Table 1 also comprises desirable results. As expected, `LearnVanishingIdeal` performs better in the Sampson distance, since it optimizes this loss function. Most importantly, it performs significantly better in the mean distance from the variety.

Example 4.8. Increase the ambient space's dimension by 1 and treat a reducible and disconnected variety V_2, generated by the intersection of a plane with two separated spheres. The example illustrates that the irreducibility of the variety is insignificant for our algorithm. The corresponding vanishing ideal is given by

$$\begin{aligned} \mathcal{I}(V_2) &= \left(\langle y+z,\ (x-2)^2+y^2+z^2-1\rangle \cap \langle y+z,\ (x+2)^2+y^2+z^2-1\rangle\right. \\ &= \left\langle y+z,\ \left(x-2\right)^2+y^2+z^2-1\right) \cdot \left((x+2)^2+y^2+z^2-1\right)\rangle\,. \end{aligned}$$

We sample 560 points from V_2 and corrupt them with 3% Gaussian noise. The resulting data points can be seen in Figure 10.

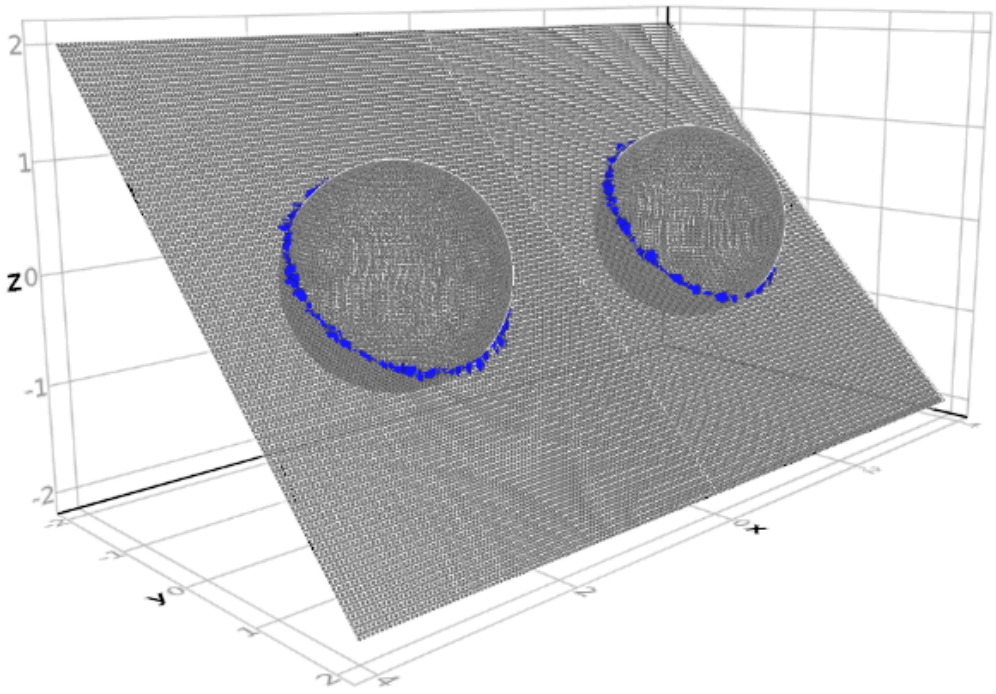

Figure 10: Samples from the intersection of two spheres and a plane.

The algorithms' performance with values rounded to 3 significant digits can be seen in Table 2. Once more, the `LVI` methods are significantly slower than the algorithm `FindEquations`. Naturally, this is an issue that needs to be dealt with. Yet, the primary goal is to reach maximum accuracy. We can deduce from Table 2 that the SVD-based method `FindEquations` performs

	`FindEquations`	`LVI_List`	`LVI_Max`
Mean Squared Error	$8.0 \cdot 10^{-10}$	$8.74 \cdot 10^{-4}$	$1.01 \cdot 10^{-7}$
Sampson Distance	2.44	0.00556	0.00512
Mean Distance	0.352	0.0391	0.0344
Consumed Time	$0.037s$	$49.0s$	$51.6s$

Table 2: Comparison of the algorithms performed on samples from V_2.

better in the mean squared error. However, our analysis in Section 3.3 shows that learning a variety with high curvature results in comparatively farther apart contour lines in the mean squared error's plot. As a consequence, strongly curved varieties have a smaller squared error on many points than varieties with less curvature. Therefore, in optimizing a least squares problem, the mean squared error is biased towards SVD and hence `FindEquations`.

More importantly, in both the Sampson distance and the mean distance our methods perform at least an order of magnitude better than `FindEquations`. The differences between the algorithms `LVI_List` and `LVI_Max` is negligible here, with the latter performing slightly better than `LVI_List`. An explanation for this behavior is quickly found. `LVI_Max` has more parameters available for optimization. Especially, when compared to `LVI_List` that looks for the true degrees of the vanishing ideal's generators, overfitting is an inevitable consequence. For this reason, let us consider Figure 11. In the top left corner, the sample points used in the algorithms are visualized. The other three images belong to the equations the corresponding algorithm learns.

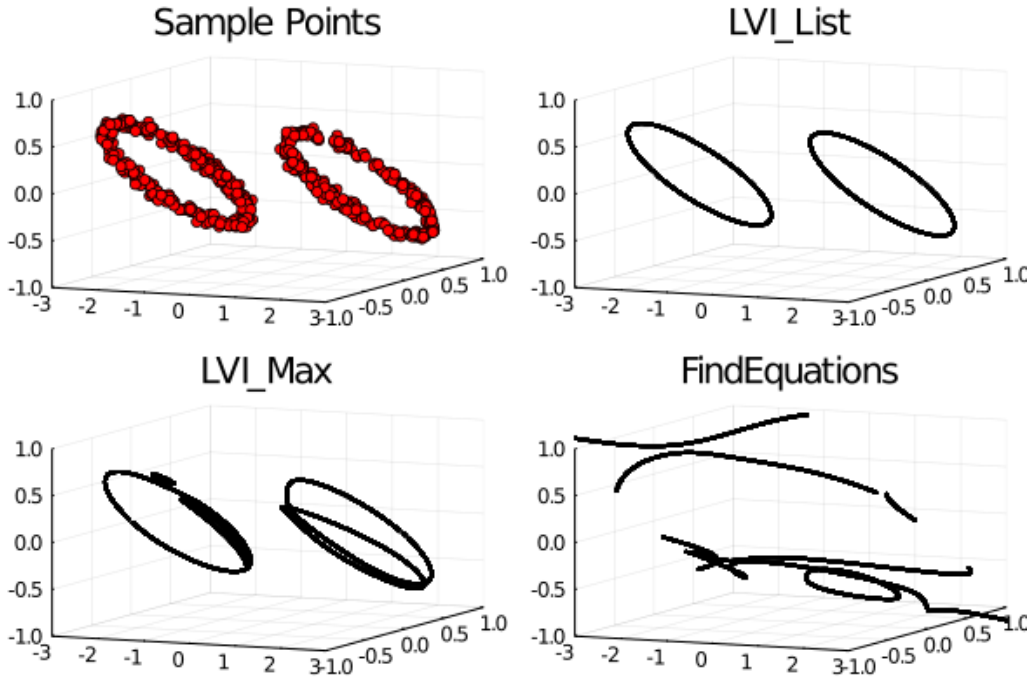

Figure 11: Comparative Plots of the different algorithms trying to learn $\mathcal{I}(V_2)$.

We can deduce that `LVI_Max` learns more irreducible components than expected, confirming previous observations. It is remarkable that `FindEquations` performs poorly in approximating the given data. Only due to its high curvature and amount of learned components is it able to perform well in the mean squared error. The results illustrate that `LVI_List` is the preferred method and should therefore be considered when taking the ideas from this thesis further.

Example 4.9. As a final example, consider the 2×3 matrices of rank at most 1, denoted by $V_{2\times 3}$. This variety is defined by the three 2×2 minors' singularity of a 2×3 matrix of independent variables. The variety $V_{2\times 3}$ has dimension 3 and degree 3, when embedded into $\mathbb{P}^5_{\mathbb{R}}$. Equivalently, it be realized as the Segre embedding $\sigma_{1,2}$ of $\mathbb{P}^1_{\mathbb{R}} \times \mathbb{P}^2_{\mathbb{R}}$ (cf. Gathmann [18, p. 60]), because of the algebraic relations that are satisfied in the image $[x_0y_0 : x_0y_1 : x_0y_2 : x_1y_0 : x_1y_1 : x_1y_2]$ of $\sigma_{1,2}$. We use 200 sample points from a data set that is provided in Breiding et al. [5] and corrupt them with 2% Gaussian noise. The result that `LearnVanishingIdeal` yields in $k[x_1, \ldots, x_6]$ is the ideal

$$\begin{aligned} I \;=\; \langle\; & -0.61x_1x_4 + 0.17x_1x_4 + 0.61x_2x_3 - 0.17x_2x_5 - 0.32x_3x_6 + 0.32x_4x_5, \\ & -0.35x_1x_4 - 0.46x_1x_6 + 0.35x_2x_3 + 0.46x_2x_5 + 0.41x_3x_6 - 0.41x_4x_5, \\ & 0.11x_1x_4 - 0.51x_1x_6 - 0.11x_2x_3 + 0.51x_2x_5 - 0.48x_3x_6 + 0.48x_4x_5 \;\rangle \end{aligned}$$

with its coefficients rounded to 2 digits. Conversely, `FindEquations` outputs

$$\begin{aligned} J \;=\; \langle\; & -0.47x_1x_4 + 0.29x_1x_6 + 0.47x_2x_3 - 0.28x_2x_5 - 0.45x_3x_6 + 0.44x_4x_5, \\ & 0.49x_1x_4 + 0.45x_1x_6 - 0.5x_2x_3 - 0.44x_2x_5 + 0.01x_2x_6 - 0.24x_3x_6 + 0.24x_4x_5, \\ & -0.18x_1x_4 + 0.47_1x_6 + 0.18x_2x_3 - 0.47x_2x_5 + 0.01x_2x_6 + 0.49x_3x_6 - 0.49x_4x_5 \;\rangle \end{aligned}$$

with the coefficients rounded to 2 digits again. Finally, the true vanishing ideal of $V_{2\times 3}$ is

$$\mathcal{I}(V_{2\times 3}) \;=\; \langle\; x_2x_3 - x_1x_4,\; x_2x_5 - x_1x_6,\; x_4x_5 - x_3x_6 \;\rangle.$$

It can be checked using a computer algebra system such as `Singular` (cf. Decker et al. [11]) that I and $\mathcal{I}(V_{2\times 3})$ agree. Even after omitting the monomial x_2x_6 from the generators, J is a different ideal. The dimension of J is 2, while its degree is 4. Accordingly, J geometrically does not represent $\mathcal{I}(V_{2\times 3})$, confirmed by the experimental results in Table 3.

	`FindEquations`	`LearnVanishingIdeal`
Mean Squared Error	0.00165	0.00260
Sampson Distance	0.0936	0.00108
Mean Distance	0.456	0.298
Consumed Time	$0.57s$	$11.8s$

Table 3: Experimental results on $V_{2\times 3}$.

Our methods `LVI_List` and `LVI_Max` are equivalent in this instance, as the three generators of $\mathcal{I}(V_{2\times 3})$ all have degree 2. The mean distance does not have a high validity in this setting, because it is calculated in $\mathbb{R}^6$. In $\mathbb{P}^5_{\mathbb{R}}$ however, there is an equivalence relation up to scaling, since the lines in $\mathbb{R}^6$ are equivalent in $\mathbb{P}^5_{\mathbb{R}}$. Consequently, the error is higher than usual for both compared algorithms. Breiding et al. [5] propose the Fubini distance to cope with this issue. For the sake of brevity, it is not introduced in this thesis.

The Sampson distance is a first-order approximation of the euclidian distance, as discussed in Chapter 3.3. The method `LearnVanishingIdeal` outperforms its counterpart in this category. Since the Sampson distance accounts for projective data by using the Veronese embedding ν_n as monomial vector, this error function is a good measure for the algorithms' quality in a projective setting.

4.7 Limitations of the Algorithm `LearnVanishingIdeal`

Example 4.10. Another interesting instance of the `LVI` algorithm arises when dealing with a variety, where Bézout's Theorem 2.8 does not hold. For this reason, consider

$$\begin{aligned} V_3 &= \mathcal{V}(\langle x^2+y^2-z^2-w^2\rangle) \cap \mathcal{V}(\langle x^2+y^2-z^2-(x+2z)\cdot w\rangle) \\ &= \mathcal{V}(\langle x^2+y^2-z^2-w^2,\ x^2+y^2-z^2-(x+2z)\cdot w\rangle) \subset \mathbb{P}^3_{\mathbb{R}}. \end{aligned}$$

In the affine chart $\{w=1\} \cong \mathbb{A}^3_{\mathbb{R}}$, this variety is an ellipse. As we have already encountered the intersection of two spheres in Section 4.5, this ellipse is not depicted here. In contrast, in the plane at infinity $\{w=0\} \cong \mathbb{P}^2_{\mathbb{R}}$, the two hypersurfaces whose intersection is V_3 share a common, positive-dimensional component, namely the cone

$$\mathcal{V}(\langle x^2+y^2-z^2\rangle) \subset \mathbb{P}^2_{\mathbb{R}}.$$

Since the two hypersurfaces share a positive-dimensional component, Bézout's theorem does not hold. Given 454 samples from this affine variety, corrupted with 6% Gaussian noise, the involved algorithms produce curves, whose affine chart $\{w=1\}$ is display in Figure 12.

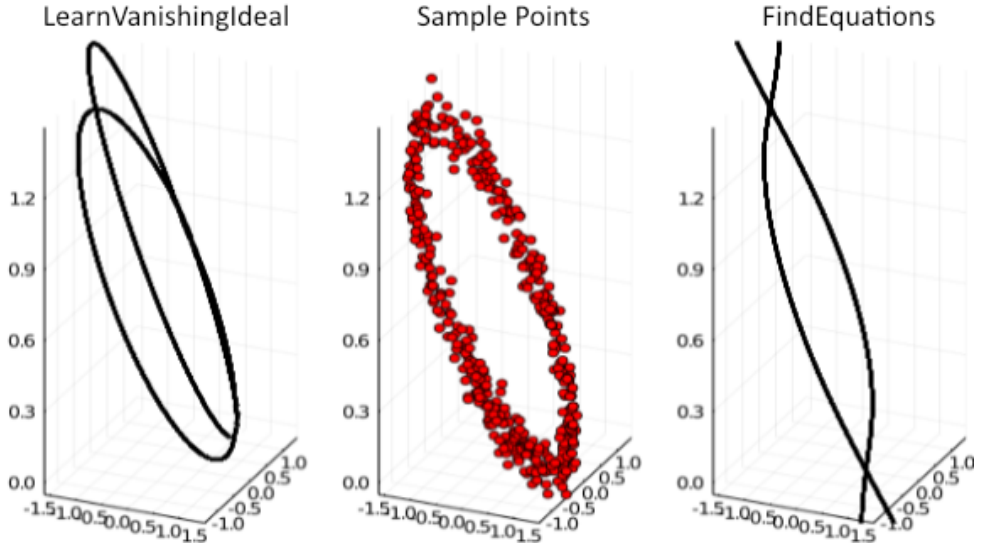

Figure 12: Comparative plots of the algorithms' different results for samples from V_3.

The algorithms `LVI_List` and `LVI_Max` are equivalent in this example, since we assume two generators of degree 2. Recall that in Chapter 4.5, intending to learn two degree 2 generators results in our algorithm finding an affine variety consisting of two irreducible components. The cone in the plane at infinity $\{w=0\}$ is not replicated, implying that the projective structure of the variety V_3 is not reproduced by either algorithm. As an alternative measure for quality, consider Table 4. The algorithms' errors and runtime are rounded to three significant digits.

	`FindEquations`	`LearnVanishingIdeal`
Mean Squared Error	0.00225	0.00425
Sampson Distance	0.199	0.00432
Mean Distance	0.487	0.0834
Consumed Time	$0.006s$	$13.5s$

Table 4: Quantitative results of the algorithms on samples from V_3.

Error-wise, our method consistently performs an order of magnitude better than `FindEquations`, which is in line with our previous observations. The time `LearnVanishingIdeal` takes is acceptable, although taking drastically longer than the method it is compared to. The bottom line of the example is that it does not suffice to sample from an affine patch, when we want to recover the underlying variety's projective geometry.

Example 4.11. As a final example, take 388 sample points from the transversal union of a parabolic curve and a sphere. Again, the Bézout bound does not hold. 5% Gaussian noise is induced. We denote this variety by V_4. It can be realized as the intersection

$$V_4 \;=\; \mathcal{V}(\langle (x^2+y^2+z^2-1)\cdot z\rangle) \;\cap\; \mathcal{V}(\langle (x^2+y^2+z^2-1)\cdot (y+x^2-2)\rangle).$$

Consequently, the corresponding vanishing ideal is given by

$$\mathcal{I}(V_4) \;=\; \langle (x^2+y^2+z^2-1)\cdot z,\; (x^2+y^2+z^2-1)\cdot (y-x^2+2)\rangle$$

and the vanishing locus can be seen in Figure 13, alongside the mentioned samples.

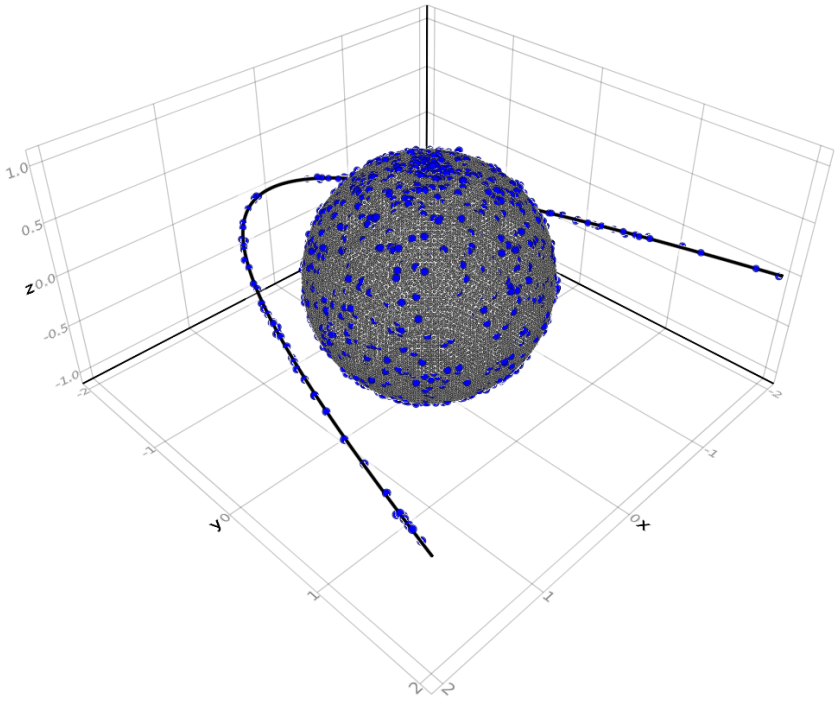

Figure 13: The variety V_4, whose degree is strictly below the Bézout bound.

The variety V_4 has dimension 2 and degree 4. However, the Bézout bound is 12 in this case. One may wonder, how this discrepancy affects our algorithm. To answer this question, consider Table 5.

	`FindEquations`	`LVI_List`	`LVI_Max`
Mean Squared Error	$1.48 \cdot 10^{-3}$	$8.46 \cdot 10^{-3}$	$4.48 \cdot 10^{-3}$
Sampson Distance	0.0304	0.0120	0.0180
Mean Distance	0.140	0.170	0.135
Consumed Time	$0.074s$	$49.0s$	$55.6s$

Table 5: Algorithmic comparison on samples from V_4.

Containing the errors arising from `LearnVanishingIdeal`'s application to the data points, Table 5 reveals that the algorithm's approximation is less than excellent. While the points are relatively close to the original variety, the mean distance of each method is comparatively large. Certainly, all errors are larger than expected when consulting Figure 13. To understand, how to interpret said behavior, consider Figure 14. In this picture, the image in the top left corner depicts the sample points. The method `LVI_List` from Section 4.5, our standard approach `LVI_Max` from Section 4.4 and `Findequations` from Section 4.1 are labelled accordingly.

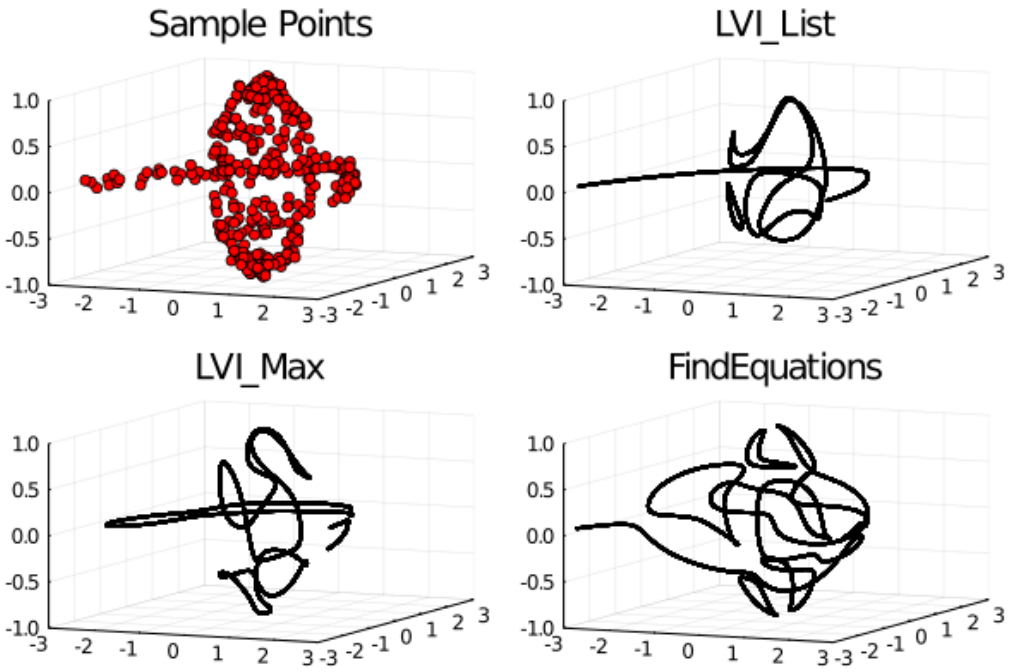

Figure 14: Visual comparison of the methods' results acting on samples from V_4.

It immediately becomes clear what went wrong. Our algorithm learns a curve, because it is given the freedom of 2 equations for a variety of codimension 1. Apparently, the method interprets the instruction to find 2 equations as finding a variety of codimension 2. Recall that our assumption throughout the thesis is that the underlying variety is a complete intersection. Here we see why it is necessary. All discussed algorithms behave poorly on the variety V_3. Nevertheless, Figure 14 indicates that `LVI_List` behaves best among them, which is coherent with our previous observations.

Unfortunately, the results in Table 5 do not reflect the above claim. The compared methods' performance does not differ significantly, resulting in a different optimal method for each compared category. The degree of the curves that `LVI_Max` and `FindEquations` learn is 16, while `LVI_List` learns a curve of degree 12. Having more parameters available to fit a curve to a variety of a higher dimension, puts the resulting errors in perspective.

Ma et al. [24] discusses several techniques to cope with the problem of different-dimensional irreducible components. One of them is the segmentation of data points into equidimensional subsets by recursive application of the generalized principal component analysis. In the equidimensional case, our method has proven to work well, making us believe that through future research the assumption that the underlying variety is a complete intersection can be omitted.

5 The Cayley-Bacharach Theorem

In this section we want to shine a light on a theoretical aspect of the algorithm we propose. Specifically, any cubic passing through 8 distinct points from the intersection of two cubics containing exactly 9 points, also passes through the remaining ninth point. This is a classic result, which is generally attributed to Cayley and Bacharach. However, they prove a generalization of this theorem. The more instrinsic formulation that is stated above is by Michel Chasles (cf. Plaumann [31, p. 14]).

The ultimate goal is to understand, if the algorithm respects this theorem. Inherently, when looking for a best-approximation that passes through 8 distinct points from the discrete intersection of two cubics inside the vector space of all cubics in $\mathbb{P}^2_{\mathbb{R}}$ or $\mathbb{P}^2_{\mathbb{C}}$, the curve our algorithm learns has to pass through the ninth. Otherwise, it contradicts Chasles' theorem. In the following, we pick $\mathbb{R}$ or $\mathbb{C}$ and denote by $\mathbb{P}^2$ the corresponding two-dimensional projective space.

First, we introduce terminology. A set of points $\{p_1, \dots, p_m\} \subset \mathbb{P}^2$ imposes ℓ conditions on polynomials of degree d, if the subspace of polynomials vanishing on $\{p_1, \dots, p_m\}$ has codimension ℓ. Incidentally, the vector space of degree d homogeneous polynomials in 3 variables has dimension $\binom{d+2}{d}$. Denote the number of conditions that are imposed by $\{p_1, \dots, p_m\}$ on the degree d polynomials by $h_{\{p_1,\dots,p_m\}}(d)$ and call it the Hilbert function. We say that $\{p_1, \dots, p_m\}$ fails to impose independent conditions on degree d polynomials, if $h_{\{p_1,\dots,p_m\}}(d) < m$. For the proof of Chasles' theorem, we need a technical lemma from Plaumann [31, p. 16] that is stated below.

Lemma 5.1. *Let $\Gamma = \{p_1, \dots, p_n\} \subset \mathbb{P}^2$ be a collection of $n \leq 2d+2$ distinct points. The points in Γ fail to impose independent conditions on curves of degree d if and only if either $d+2$ of the points in Γ are colinear or $n = 2d+2$ and additionally, Γ is contained in a conic.*

Proof. The proof of this lemma can be found in Plaumann [31, p. 16]. □

This enables us to state Chasles' theorem.

Theorem 5.2 (Chasles [9])**.** *Let X_1, X_2 be two cubic curves in $\mathbb{P}^2$ meeting in $\Gamma = \{p_1, \dots, p_9\}$, a set of 9 distinct points. Then any cubic passing through an arbitrary subset of eight points $\Gamma' = \{p_1, \dots, p_8\} \subset \Gamma$ also passes through p_9.*

Proof. A cubic curve denotes a variety of dimension 1 and degree 3. Notice that Bézout's theorem 2.8 guarantees that the maximal number of intersection points of two such cubic curves is 9. Assume Γ is a set of 9 points, defined by the intersection of X_1 and X_2. Let Γ' be any 8-element subset of Γ. We need to show that Γ and Γ' impose the same number of conditions on planar cubic curves. As a result, the same polynomials of degree 3 vanish on both sets. Intuitively, we expect that Γ imposes 9 conditions on the space of cubic curves. Nevertheless, we know two linearly independent polynomials of degree 3 vanishing on Γ: The defining equations f_1 and f_2 of X_1 and X_2. These polynomials are linearly independent, because if there was a linear relation $f_1 = \lambda f_2$ for $\lambda \in \overline{k}$, then

$$X_1 = \mathcal{V}(f_1) = \mathcal{V}(f_2) = X_2.$$

This is a contradiction to the assumption that the cubics X_1 and X_2 meet in exactly 9 points, because the underlying field is infinite and so are X_1 and X_2. Since there are two linear independent polynomials vanishing on Γ', $h_\Gamma(3) \leq 8$. To conclude the proof, we need to show that $h_{\Gamma'}(3) \geq 8$. Contrary to that, assume $h_{\Gamma'}(3) < 8$. By Lemma 5.1, this implies that either 5 of the points are colinear or Γ' is contained in a conic.

For the first part of this statement, note that by Bézout's theorem a line ℓ intersected with a cubic in relative general position contains at most 3 points. Therefore, the cubic must contain

the line ℓ, as they supposedly intersect in 5 points. Following this line of argumentation, both, X_1 and X_2 need to contain ℓ, as they contain Γ'. Consequently, $X_1 \cap X_2 \supset \ell$, contradicting the finiteness of $X_1 \cap X_2$.

In the case that Γ' is contained in a conic, Bézout's theorem guarantees that this conic C intersected with either cubic X_1 or X_2 has at most 6 points of intersection. At least, when X_1 and X_2 lie in relative general position. However, Γ' cotains 8 distinct points which is a contradiction to the relative generality of the curves. Hence, both cubics must contain a positive-dimensional component of the conic C. As the intersection $X_1 \cap X_2$ is finite by assumption, no component of C can be contained in $X_1 \cap X_2$. The only option left to consider is that C consists of two distinct lines: $C = \ell_1 \cup \ell_2$ with $\ell_1 \subset X_1$ and $\ell_2 \subset X_2$. By assumption, both $X_1 \cap X_2$ and $\ell_1 \cup \ell_2$ contain Γ'. This is impossible, as the intersection of two distinct lines contains at most 1 point. This demonstrates that $h_{\Gamma'} = 8$, concluding the proof. □

According to Eisenbud et al. [15, p. 307] a generalization of Chasles' theorem is available. The following statement is a version of the famous theorem by Cayley and Bacharach.

Theorem 5.3 (Cayley-Bacharach)**.** *Let $X_1,\ X_2 \subset \mathbb{P}^2$ be curves of degrees d_1 and d_2 respectively, meeting in a collection of $d_1 d_2$ distinct points $\Gamma = \{p_1, \ldots, p_{d_1 d_2}\}$. If $X \subset \mathbb{P}^2$ is any curve of degree $d_1 + d_2 - 3$ containing all but one point of Γ, then X contains all of Γ.*
More generally, suppose that Γ is the disjoint union of two subsets Γ' and Γ''. Then,

$$h_\Gamma(m) = h_{\Gamma'}(m) - h_{\Gamma''}(d_1 + d_2 - 3 - m) + |\Gamma''|$$

holds for any $m \leq d_1 + d_2 - 3$.

Proof. The theorem's proof can be found in Eisenbud et al. [15, pp. 307-308] □

To understand, how the algorithm `LearnVanishingIdeal` copes with Chasles' Theorem 5.2, we consider an example. We generate the set of 9 points Γ by the two equations $f_1 = x(x-z)(x+z)$ and $f_2 = y(y-z)(y+z)$ in the coordinate ring $\mathbb{C}[x,y,z]$ of $\mathbb{P}^2_{\mathbb{C}}$. The points of intersection are

$$\begin{aligned}\Gamma \ = \ \{&[1:0:1], [0:0:1], [-1:0:1], [1:1:1], [0:1:1], [-1:1:1],\\ &[1:-1:1], [0:-1:1], [-1:-1:1]\}\end{aligned}$$

Conveniently, Γ lies in the affine patch $\{z \neq 0\}$ of $\mathbb{P}^2_{\mathbb{C}}$ and all points are real. The situation is displayed in Figure 15. We choose $p_9 = [-1:-1:1]$ and define $\Gamma' = \Gamma \setminus \{p_9\}$. Chasles' theorem predicts that any cubic going through the 8 points in Γ' also passes through p_9. In terms of our algorithm, we want to insert Γ' into the Sampson distance and equate the outcome to 0. This way, the resulting polynomial is optimal. Subsequently, we derive a coefficient vector $w \in \mathbb{C}^{10}$ with $Q = w^T \nu_3^{\mathbb{P}}(x,y,z)$ vanishing on Γ'. In this case, the Sampson distance $\mathcal{L}_S(w;\Gamma')$ yields an equivalent result to the mean squared error $\mathcal{L}_{MS}(w;\Gamma')$, as it only introduces a weighting.

Proposition 5.4. *If a polynomial q's gradient is nonsingular on $\Omega = \{z_1, \ldots, z_N\}$, then q vanishes on Ω in the mean squared error if and only if it vanishes in the Sampson distance.*

Proof. Let q be an arbitrary polynomial and let $\Omega = \{z_1, \ldots, z_N\}$ be a set of points. Assume that the Sampson distance is well-defined, meaning that the gradient $\nabla q(z_i) \neq 0$ does not vanish for any $z_i \in \Omega$. Equivalently, $\mathcal{V}(q)$ does not have singularities that coincide with Ω. As a norm is positive-definite, a polynomial vanishes on Ω, if and only if each summand of the mean squared error $\frac{1}{N}\sum_{i=1}^N ||q(z_i)||^2$ is zero. This is still the case, when we multiply each summand with a non-zero constant. Since the gradient of q is not singular in Ω, each summand of the Sampson distance is also 0, implying that the entire sum is 0. The proof of the other implication is analogous, because $||\nabla q(z_i)||^2 > 0$ by assumption. □

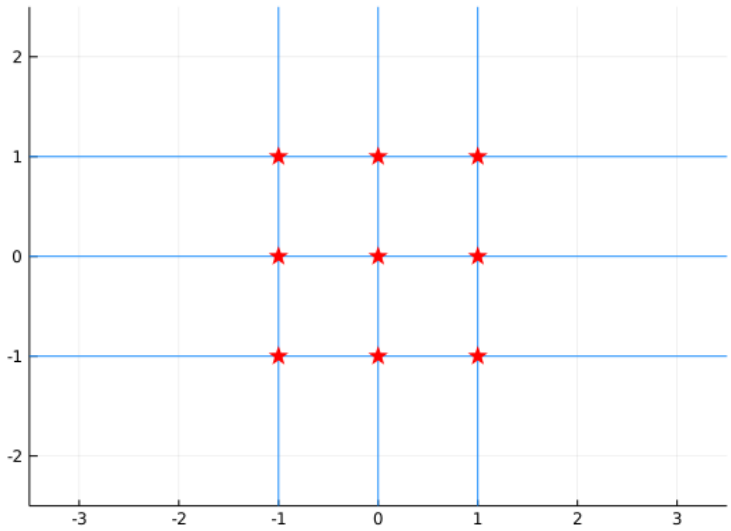

Figure 15: Intersection of cubics f_1 (vertical lines) and f_2 (horizontal lines).

Using Proposition 5.4, we can choose the error function among $\mathcal{L}_{MS}$ and $\mathcal{L}_{SD}$ with simpler output. Picking the mean squared error results in the loss function

$$\mathcal{L}_{MS}(w;\Gamma') = \frac{1}{8}\sum_{i=1}^{8} ||Q(p_i;w)||^2 = \frac{1}{8}\left(5w_1^2 + 2w_1w_2 - 2w_1w_3 + 6w_1w4 + 2w_1w_5 + \dots\right) = 0.$$

We want the polynomial our method produces to be the loss function's optimum. As a consequence, it is additionally required that $\nabla_w \mathcal{L}_{MS}(w;\Gamma') = \mathbf{0}$. As differentiation of a quadric polynomial produces linear forms, this equality leads to a linear system $Aw = 0$ for a coefficient matrix A. Thus, the rank of A is 8 with a basis of A's kernel given by the two coefficient vectors of f_1 and f_2. It follows that the only polynomials vanishing on Γ' are given by linear combinations of f_1 and f_2. Hence, for any linear combination g of f_1 and f_2 it holds that

$$g(p_9) = \lambda f_1(p_9) + \mu f_2(p_9) = \lambda \cdot 0 + \mu \cdot 0 = 0,$$

implying that any cubic passing through Γ' also passes through p_9. How `LearnVanishingIdeal` deals with Γ' as input is displayed in Figure 16 on the next page with the excluded point point p_9 displayed as a star. It is illustrated that our algorithm implicitly learns the conditions imposed by two intersecting cubics.

More generally, we want to prove that our algorithm has this output for an arrangement of 9 distinct points $\Gamma = \{p_1, \dots, p_9\}$ from the intersection of two arbitrary cubics in $\mathbb{P}^2$. In doing so, we rewrite the loss function $\mathcal{L}_{MS}(w, \Gamma')$ to an eigenvalue problem for any 8 element subset $\Gamma' = \{p_1, \dots, p_8\}$ of Γ. This is already executed in the proof of Proposition 3.8, creating the eigenvalue problem

$$A(\Gamma') \cdot w = 0 \qquad \text{for} \qquad A(\Gamma') = \frac{1}{8}\sum_{i=1}^{8} \nu_3^{\mathbb{P}}(p_i)\nu_3^{\mathbb{P}}(p_i)^T \in \mathbb{C}^{10\times 10}. \tag{13}$$

Notice that this formulation only captures the loss function's $\mathcal{L}_{MS}$ global minima. The following proposition resolves this discrepancy.

Proposition 5.5. *$w \in \mathbb{C}^{10}$ solves the eigenvalue problem (13) if and only if $Q(\boldsymbol{x}; w) = w^T \cdot \nu_3^{\mathbb{P}}(\boldsymbol{x})$ minimizes the loss function $\mathcal{L}_{MS}(w;\Gamma') = \frac{1}{8}\sum_{i=1}^{8} ||Q(p_i;w)||^2$.*

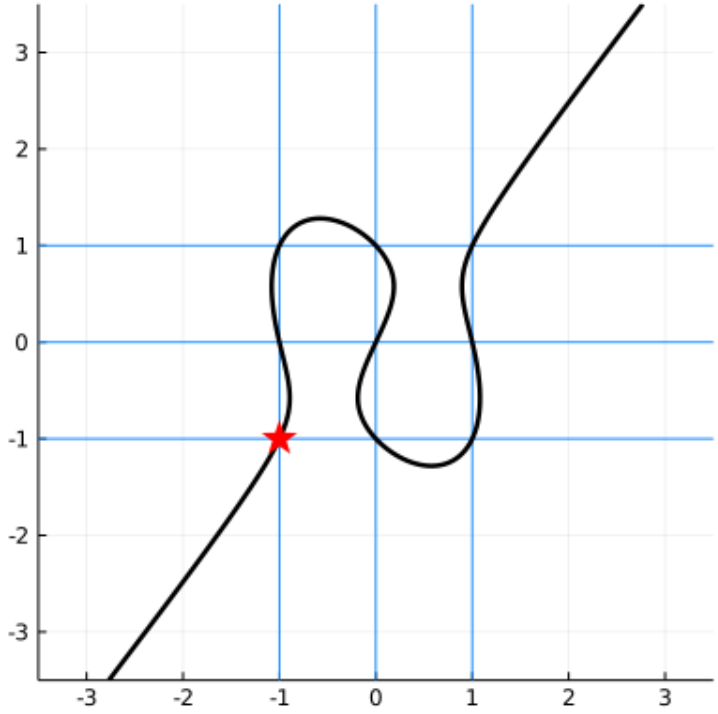

Figure 16: The cubic `LearnVanishingIdeal` learned with respect to Γ' passing through Γ.

Proof. Take any $c \in \ker(A(\Gamma'))$. Then,

$$0 = c^T \cdot A(\Gamma') \cdot c = \frac{1}{8} \sum_{i=1}^{8} c^T \cdot \nu_3^{\mathbb{P}}(p_i) \nu_3^{\mathbb{P}}(p_i)^T \cdot c = \mathcal{L}_{MS}(c; \Gamma') \tag{14}$$

for $Q(\mathbf{x}) = c^T \nu_3^{\mathbb{P}}(\mathbf{x}) = \nu_3^{\mathbb{P}}(\mathbf{x})^T c$. As $||Q(p_i)|| \geq 0$ for all $p_i \in \mathbb{P}^2$, c minimizes $\mathcal{L}_{MS}(w; \Gamma')$.

For the other implication, assume that $Q(\mathbf{x}) = \nu_3^{\mathbb{P}}(\mathbf{x})^T c$ minimizes $\mathcal{L}_{MS}(w; \Gamma')$. The equations (14) demonstrate that $\mathcal{L}_{MS}(w; \Gamma')$ is a quadratic optimization problem. Moreover, the matrix $A(\Gamma') = \frac{1}{8} \sum_{i=1}^{8} \nu_3^{\mathbb{P}}(p_i) \nu_3^{\mathbb{P}}(p_i)^T$ is symmetric positive semidefinite. Burke's Theorem 3.2 [17, pp. 28-29] implies that every local minimum of $\mathcal{L}_{MS}(w; \Gamma)$ is global. Hence, it suffices to show that $\mathcal{L}_{MS}(c; \Gamma) = 0$ is possible. Theorem 3.2 in Burke [17, pp. 28-29] proves this by demonstrating that each solution to $A(\Gamma')w = 0$ is a local minimum of $\mathcal{L}_{MS}(w; \Gamma')$, concluding the proof. □

The formulation of the eigenvalue problem (13) and Proposition 5.5 allow us to assure that our algorithm abides by the Cayley-Bacharach theorem.

Theorem 5.6. *Assume that two cubic plane curves X_1 and X_2 meet in 9 distinct points $\Gamma = \{p_1, \ldots, p_9\}$ and let $\Gamma' = \{p_1, \ldots, p_8\} \subset \Gamma$. Then, the coefficient vector w appears as a solution of (13) if and only if w yields a cubic $q(x, y, z) = w^T \nu_3^{\mathbb{P}}(x, y, z)$ that vanishes on Γ.*

Proof. Let $w \in \mathbb{C}^{10}$ correspond to a polynomial $q(x, y, z) = w^T \nu_3^{\mathbb{P}}(x, y, z) = \nu_3^{\mathbb{P}}(x, y, z)^T w$ that vanishes on Γ. As $\Gamma' \subset \Gamma$ and $q|_{\Gamma} = 0$, w satisfies the eigenvalue problem (13) by construction.

For the other implication, let w solve the eigenvalue problem (13). By Proposition 5.5,

$$\mathcal{L}_{MS}(\Gamma'; w) = \frac{1}{8} \sum_{i=1}^{8} ||q(p_i; w)||^2 = 0$$

for the polynomial $q(\mathbf{x}; w) = w^T \nu_3^{\mathbb{P}}(\mathbf{x})$. As any norm is nonnegative, each summand of $\mathcal{L}_{MS}(\Gamma'; w)$ must vanish. Consequently, q vanishes on Γ' by the norm's definiteness. Chasles Theorem 5.2 then implies that q vanishes on Γ. □

Assume that our algorithm `LearnVanishingIdeal` is presented with an 8-element subset Γ' of $\Gamma = \{p_1, \ldots, p_9\} \subset \mathbb{P}^2$ as training set. Again, Γ is given by the intersection of two plane cubics. If any point $p \in \Gamma$ corresponded to a singularity of either curve, the intersection's multiplicity in p would be at least 2 by Corollary 2.24 in Gathmann [19, p. 17]. Two cubics intersect in at most 9 or infinitely many points by Bézout's theorem, even when counting multiplicities (cf. Gathmann [19, p. 31]). Hence, the cardinality of Γ cannot be 9, as one point is counted at least twice. This contradicts our assumption, so Γ does not contain a singularity of either cubic. As a result, the combination of Proposition 5.4, Proposition 5.5 and Theorem 5.6 guarantees that our algorithm correctly identifies the polynomials that vanish on Γ. Consequently, our algorithm implicitly learns the rule that is prescribed by the Cayley-Bacharach theorem.

6 Conclusion and Future Work

6.1 Conclusion

Numerical algebraic geometers try to infer generators of a variety's vanishing ideal from a point cloud. In doing so, they want to preserve the topological and geometric structure of the original vanishing locus. While there are methods available to find generators in the noise-free case, the introduction of noise to the data points complicates this undertaking.

As a solution, this thesis proposes the `LearnVanishingIdeal` algorithm for learning generators of a complete intersection V's vanishing ideal from potentially noisy data points Ω. After introducing the notion of affine and projective varieties and corresponding properties such as a variety's dimension and degree, we can present the generalized principal component analysis as our method's foundation. Knowing the participating linear spaces' number and dimension, this algorithm aims at segmenting sample points from subspace arrangements. A recursive formulation makes it possible to relax the assumption that the involved dimensions are known. In the presence of noise, statistical techniques depending on minimizing the distance from the sample points to the learned variety provides meaningful results. A different approach to learning polynomials vanishing on point clouds is given by the algorithm `FindEquations`. The evaluation of a finite set of monomials $\mathcal{M}$ in Ω creates the multivariate Vandermonde matrix. If $\mathcal{M}$ is chosen well enough, its kernel generates the vector space $\mathcal{I}(V) \cap \mathcal{M}$. Nevertheless, this approach is prone to perturbations. As a result, reasonable adjustments are necessary. By establishing several error functions, the approximation's quality can be quantified. The application of statistical techniques previously applied in the generalized principal component analysis leads to the implementation of `LearnVanishingIdeal`. In the process, knowledge about the variety's dimension and the maximal occuring degree among a minimum generating set is assumed. Immediately, variations are proposed that enhance the algorithm's accuracy and complexity. Nonetheless, overfitting remains an issue. By choosing an approach that considers the individual degrees of a minimum generating set of V's vanishing ideal, overfitting is reduced. Both, the method that takes only the maximal occuring degree and the method that takes all degrees as input are evaluated in several examples. They exhibit good results, especially when compared to `FindEquations`. Further experiments indicate it is essential that the underlying variety is a complete intersection. In other circumstances, the method `LearnVanishingIdeal` does not work reliably. As a final remark, the algorithm satisfies the conditions imposed by the Cayley-Bacharach theorem on the intersection of two planar cubics.

The algorithm proposed in this thesis generalizes the methods for subspace arrangements by Ma et al. to algebraic varieties. A clever idea by Sampson transforms the nonlinear optimization problem to a quadratic optimization problem with quadratic constraints. By embedding the sample points into projective space, previous issues are resolved. Assuming knowledge about the generators' degrees makes it possible to reduce overfitting. With a few variations, this algorithm works dependably for complete intersections. Finally, the algebraic calculation of derivatives enhance the method's runtime.

6.2 Future Work

Under the thesis' assumptions, the algorithm `LearnVanishingIdeal` produces convincing results. Unfortunately, real-world data at times does not provide knowledge of the generating equations' degrees and the variety's dimension, suggesting that our assumptions are overly optimistic. Consequently, another algorithm is needed for the respective data. For this reason, a method for calculating the underlying variety's dimension from point samples has been proposed (cf. Breiding et al. and Camastra [5, 8]). Consequently, we assume that the variety's dimension is known. As previously mentioned, the variety our samples Ω come from is a complete intersection, so we

need to learn codimension-many equations, i.e. $m = \operatorname{codim}(V) = D - \dim(V)$.

The only task we are left with is finding the right degrees for our algorithm. We can start with the variety's degree $n = 1$ and go through all possible combinations according to Bézout's theorem. Immediately, this poses the question of when to stop looking. The error tends to decrease with increasing degree, as the search space contains more parameters. In order to determine when we are satisfied with the result, many textbooks propose using a stopping criterion. One such criterion arises as the prescription of a maximum degree. However, doing so might lead to underfitting the data, if the algorithm is not provided with an educated guess for the underlying variety's degree. Alternatively, another stopping criterion is progressing until the error does not considerably decrease anymore. Even so, this stopping criterion is insufficient for varieties. To understand why, consider Figure 17 of data points from a parabola of degree 4 in $\mathbb{R}^2$.

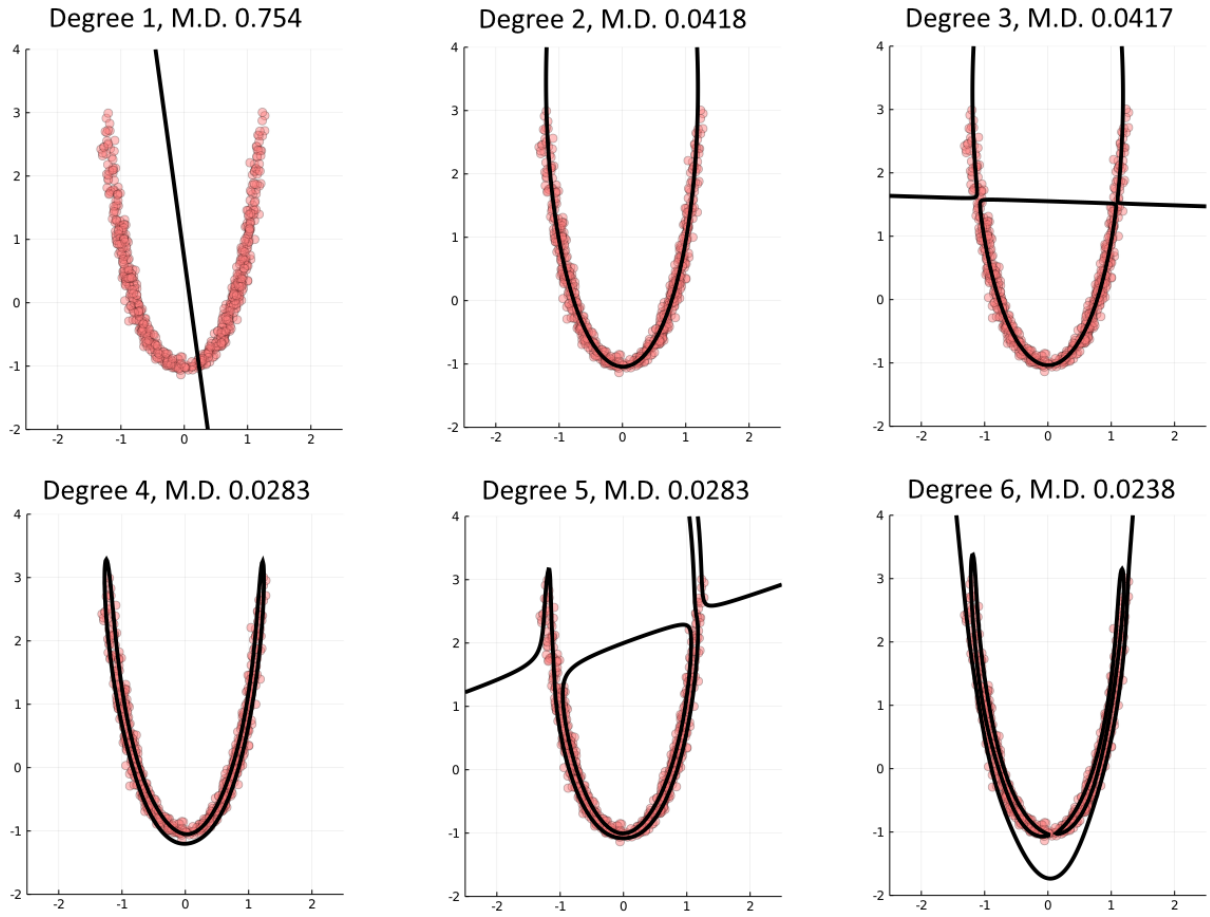

Figure 17: Application of `LearnVanishingIdeal` to samples from the parabola $y - x^4 - x^2 + 1$.

We sampled 690 points from the variety with vanishing ideal $\langle y - x^4 - x^2 + 1 \rangle$ and 5% Gaussian noise. Afterwards, the method `LearnVanishingIdeal` is applied to the data points, using a degree from 1 through 6. The resulting vanishing set is then plotted and labelled with the corresponding degree and mean distance rounded to three significant digits. Recall that the mean distance is defined in Section 4.2, referring to the arithmetic mean of the distances from the sample points to the learned variety. The error only significantly changes when the curve's degree is even. We suspect this is due to 2 dividing 4. However, further investigation is necessary here. Another important observation to be made in Figure 17 is that it does not suffice to sample points in the bounded quadrangle $[-3, 3]^2$. In degree 4, `LearnVanishingIdeal` fits a bounded variety in $\mathbb{R}^2$ to the samples, although the original variety is unbounded. At the moment, it

is sufficient to realize that all of these observations make it impracticable to find a stopping criterion with these approaches. Taking stagnation of the mean distance as reference, a degree 2 curve is our best guess, underestimating the degree. Conversely, if we take a more sensitive threshold for the stopping criterion, the algorithm may not terminate within the first 6 degrees, corresponding to an overestimation of the degree. This fundamental tension deserves additional attention in future work.

All the arguments above demonstrate that a sensible stopping criterion encompasses a measure for the search space's complexity. This involves a déja-vu with a problem that is supposedly already solved in Section 4.5: overfitting. It occurs in the form of learning a higher-degree variety than necessary when looking for a best-approximating degree. In statistical learning theory, there are two famous approaches for model selection: information criteria (cf. James et al. [23, p. 210-213]) and cross-validation (cf. James et al. [23, pp. 176-178]). On the one hand, information criteria try to choose the best model by adding a term to the loss function that accounts for the complexity of the search space. As a larger parameter space yields an approximation with smaller error, the result is high variance. At least, if it contains the previous search spaces. A solution can be found by penalizing an increasing coefficient space's dimension. For instance, this can be achieved by adding

$$\mathcal{L}_{aS}(\mathfrak{C}; \Omega, \underline{\mathbf{n}}) + \frac{\sigma^2}{N} \cdot \sum_{i=1}^{m} \binom{n_i + D}{D}$$

to the adjusted Sampson distance $\mathcal{L}_{aS}$ from Section 4.5 with the isotropic Gaussian noise model's variance σ^2 and a configuration of degrees $\underline{\mathbf{n}} = (n_1, \ldots, n_m)$. The added term is analogous to the Akaike information criterion in Ma et al. [24], a famous model selection criterion. The sum stretches over the dimensions of the equations' search spaces. As there are additional constraints for the equations, the added term does not correctly represent the joint search space's dimension. Nonetheless, it is an adequate approximation. If σ is known or we have an adequate approximation of it, this provides a reasonable stopping criterion: Assume we have found a variety with degree d that approximates the data better than all preceding models with respect to the above loss function. Let p be the smallest prime dividing d. If we do not find a model that fits the data better until degree $p + d$, the process stops. Otherwise, the new best-approximating model is used as baseline. For this to properly work, sufficiently many data points are necessary.

Model selection via cross-validation on the other hand can be realized by splitting the data into three sets: a training set, a validation set and a test set. Our model learns equations from the training data for a given degree configuration of the underlying variety's generating equations. It is then checked on the validation set, which configuration has the smallest error with respect to the previously regularized loss function. Finally, the performance of the best fitting models is evaluated on the test set. While overfitting models tend to have a small error on the training data, they usually perform poorly on the test data. Cross-validation exploits this behavior by only using the error on the test set for choosing the best-approximating model. Potential improvements of this method are leave-one-out and k-fold cross-validation (cf. James et al. [23, pp. 178-183]). Analogous to information criteria, cross-validation chooses the best among a range of models. It can be used as supplement to information criteria, for example by choosing the best model for a fixed degree d of the learned variety $\mathcal{V}(Q)$.

A powerful way to advance research in this regard is to use topological data analysis and similar techniques to infer information about the degree of a variety. In doing so, a meaningful stopping criterion for the model selection approach from above would be provided. Deriving a variey's degree by considering point samples drawn from it is a task yet to be accomplished.

Eventually, implementing the proposed modifications to account for the relaxed assumptions is

going to drastically increase the complexity of our algorithm. Thus, ways to optimize its implementation need to be considered. The `HomotopyContinuation.jl` package uses expensive polynomial manipulations, so relying on local methods or even the Sampson distance instead might help. In addition, parallel computing, treating the threshold τ (cf. Section 4.4) as a hyperparameter and using an optimized solver for the nonlinear optimization problem (10) instead of vanilla gradient descent have the potential to significantly improve the algorithm's runtime. All of these points require experimental verification.

The sparsity of the involved polynomial equations is an important factor in the development of an algorithm for learning polynomial equations. However, it has not been studied in this thesis. Naturally, we expect the generators of an ideal to be sparse, implying that further work is required here. An idea is to feed the algorithm's output into a computer algebra system such as `Oscar.jl` (cf. Decker et al. [11]) to further analyze the polynomials. As a result, we find a more convenient representation of the data and compute a Gröbner basis of the learned ideal. Another way to solve this issue is to use a different constraint in the nonlinear optimization program that does not enforce orthonormality of the coefficient vectors with respect to the inner product $\langle \text{-}, \text{-} \rangle_\Gamma$ (cf. Section 4.4).

As discussed in Section 4.6, we have not been able to relax the assumption that the underlying variety is a complete intersection yet. Conceivably, a recursive approach as presented in Section 3.2 combined with model selection criteria can potentially help in omitting this assumption. In Ma et al. [35], a similar approach has proven to be promising.

7 References

[1] John A. Abbott, Anna M. Bigatti, Martin Kreuzer and Lorenzo Robbiano: *Computing Ideals of Points.* Journal of Symbolic Computation **30** (2000), pp. 341-356.

[2] Ake Björck and Victor Pereyra: *Solution of Vandermonde systems of equations.* Mathematics of Computation **24** (1970), pp. 893-903.

[3] Kaare Brandt Petersen and Michael Syskind Pedersen: *The Matrix Cookbook.* `math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf` (2012), Accessed: October 17, 2020.

[4] Paul Breiding and Orlando Marigliano: *Random points on an algebraic manifold.* SIAM Review **2** (2020), pp. 683–704.

[5] Paul Breiding, Sara Kalisnik Verovsek, Bernd Sturmfels and Madeleine Weinstein: *Learning Algebraic Varieties from Samples.* Revista Matemática Complutense **31** (2018), pp. 545–593.

[6] Paul Breiding and Sascha Timme: *The point on a variety that minimizes the distance to a given point.* `JuliaHomotopyContinuation.org/examples/critical-points`. Accessed: July 13, 2020.

[7] Paul Breiding and Sascha Timme: *HomotopyContinuation.jl: A package for homotopy continuation in Julia.* International Congress on Mathematical Software **10931** (2018), pp. 458-465.

[8] Francesco Camastra: *Data Dimensionality Estimation Methods: A survey.* Pattern recognition **36** (2003), pp. 2945-2954.

[9] Michel Chasles: *Construction de la courbe du troisième ordre déterminée par neuf points.* Comptes Rendus des Séances de l'Académie des Sciences **36** (1853), pp. 943–952.

[10] David A. Cox, John Little and Donal O'Shea: *Ideals, Varieties and Algorithms.* Springer Science & Business Media (2013).

[11] Wolfram Decker, Claus Fieker, Max Horn and Michael Joswig: *Oscar.jl.* `oscar.computeralgebra.de/` (2018). Accessed: November 26, 2020.

[12] Harm Derksen: *Hilbert series of subspace arrangements.* Journal of pure and applied algebra, **209** (2007), pp. 91-98.

[13] Tom G. Dietterich: *Overfitting and Undercomputing in Machine Learning.* ACM Computing Surveys **27** (1995), pp. 326-327.

[14] Jan Draisma, Emil Horobet, Giorgio Ottaviani, Bernd Sturmfels and Rekha R. Thomas: *The Euclidean Distance Degree of an Algebraic Variety.* Foundations of Computational Mathematics **16** (2016), pp. 99–149.

[15] David Eisenbud, Mark Green and Joe Harris, *Cayley-Bacharach Theorems and Conjectures.* Bulletin of the American Mathematical Society **33.3** (1996), pp. 295-324

[16] Andrew Fitzgibbon, Maurizio Pilu and Robert Fisher: *Direct least squares fitting of ellipses.* Proceedings of 13th International Conference on Pattern Recognition **1** (1996), pp. 253-257.

[17] James V. Burke: *Nonlinear Optimization.* `sites.math.washington.edu/~burke/crs/408/notes/Math408_W2020/math408text.pdf` (2020). Accessed: December 29, 2020.

[18] Andreas Gathmann: *Algebraic Geometry.* `mathematik.uni-kl.de/~gathmann/class/alggeom-2014/alggeom-2014.pdf` (2014). Accessed: November 19, 2020.

[19] Andreas Gathmann: *Plane Algebraic Curves*. mathematik.uni-kl.de/~gathmann/class/curves-2018/curves-2018.pdf (2018). Accessed: December 30, 2020.

[20] Mike B. Giles: *An extended collection of matrix derivative results for forward and reverse mode algorithmic differentiation*. Advances in Automatic Differentiation, Lecture Notes in Computational Science and Engineering **64** (2008), pp. 35-44.

[21] Robin Hartshorne: *Algebraic Geometry*. Springer Science & Business Media **52** (2013).

[22] Daniel Heldt, Martin Kreuzer, Sebastian Pokutta and Hennie Poulisse: *Approximate Computation of zero-dimensional polynomial ideals*. Journal of Symbolic Computation **44** (2009), pp. 1566-1591.

[23] Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani: *An Introduction to Statistical Learning*. Springer Texts in Statistics **103** (2013).

[24] Yi Ma, Allen Yang, Harm Derksen and Robert Fossum: *Estimation of Subspace Arrangements with Applications in Modeling and Segmenting Mixed Data*. SIAM Review **50** (2008), pp. 413–458.

[25] Mateusz Michałek and Bernd Sturmfels: *Invitation to Nonlinear Algebra*. American Mathematical Society (2021).

[26] Thomas P. Minka: *Old and New Matrix Algebra Useful for Statistics*. tminka.github.io/papers/matrix/minka-matrix.pdf (2000), Accessed: October 17, 2020.

[27] H. Michael Möller and Bruno Buchberger: *The construction of multivariate polynomials with preassigned zeros*. Lecture Notes in Computer Science **144** (1982), pp. 24-31.

[28] Peter J. Olver: *On Multivariate Interpolation*. Studies in Applied Mathematics **116** (2006), pp. 201-240.

[29] Victor Y. Pan: *How Bad Are Vandermonde Matrices?* SIAM Journal on Matrix Analysis and Applications **37** (2016), pp. 676–694.

[30] Roger Penrose: *A generalized inverse for matrices*. Mathematical Proceedings of the Cambridge Philosophical Society **51** (1955), pp. 406-413.

[31] Daniel Plaumann: *Classical Algebraic Geometry*. Lecture Notes, Universität Konstanz (2015), chapter 8.

[32] Chris Preston: *Ein Skript für Lineare Algebra I und II*. https://www.math.uni-bielefeld.de/~preston/teaching/linalg/files/linalg.pdf (2004). Accesses: November 20,2020.

[33] Paul Sampson: *Fitting conic section to "very scattered" data: An iterative refinement of the Bookstein algorithm*. Computer Vision, Graphics and Image Processing, **18** (1982), pp. 97–108.

[34] Igor R. Shafarevich: *Basic Algebraic Geometry 1. Varieties in Projective Space*. Springer-Verlag (1994).

[35] Rene Vidal, Yi Ma and Shankar Sastry: *Generalized Principal Component Analysis (GPCA)*. IEEE Transactions on Pattern Analysis and Machine Intelligence **27** (2005), pp. 1945-1959.