# Approximate Vanishing Ideal via Data Knotting

**Hiroshi Kera, Yoshihiko Hasegawa**

Department of Information and Communication Engineering,
Graduate School of Information Science and Technology,
The University of Tokyo, Tokyo, Japan
kera@biom.t.u-tokyo.ac.jp

## Abstract

The vanishing ideal is a set of polynomials that takes zero value on the given data points. Originally proposed in computer algebra, the vanishing ideal has been recently exploited for extracting the nonlinear structures of data in many applications. To avoid overfitting to noisy data, the polynomials are often designed to approximately rather than exactly equal zero on the designated data. Although such approximations empirically demonstrate high performance, the sound algebraic structure of the vanishing ideal is lost. The present paper proposes a vanishing ideal that is tolerant to noisy data and also pursued to have a better algebraic structure. As a new problem, we simultaneously find a set of polynomials and data points for which the polynomials approximately vanish on the input data points, and almost exactly vanish on the discovered data points. In experimental classification tests, our method discovered much fewer and lower-degree polynomials than an existing state-of-the-art method. Consequently, our method accelerated the runtime of the classification tasks without degrading the classification accuracy.

## 1 Introduction

Bridging computer algebra and various applications such as machine learning, computer vision, and systems biology has been attracting interest over the past decade (Torrente 2008; Laubenbacher and Sturmfels 2009; Li et al. 2011; Livni et al. 2013; Vera-Licona et al. 2014; Gao and Wang 2016). Borrowed from computer algebra, the vanishing ideal concept has recently provided new perspectives and has proven effective in various fields. Especially, vanishing ideal based approaches can discover the nonlinear structure of given data (Laubenbacher and Stigler 2004; Livni et al. 2013; Hou, Nie, and Tao 2016; Kera and Hasegawa 2016; Kera and Iba 2016). The vanishing ideal is defined as a set of polynomials that always take zero value, *i.e.*, vanishes, on a set of given data points:

$$\mathcal{I}(X) = \{g \in \mathcal{P} \mid g(\mathbf{x}) = 0, \forall \mathbf{x} \in X\}, \qquad (1)$$

where $X$ is a set of $d$-dimensional data points, and $\mathcal{P}$ is a set of $d$-variate polynomials. A vanishing ideal can be spanned by a finite set of vanishing polynomials (Cox, Little, and O'shea 1992). This basis of the vanishing ideal can
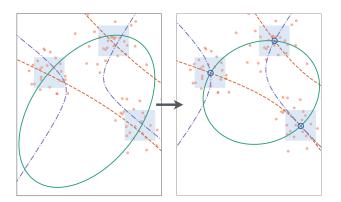
Figure 1: (Left panel) Polynomials that approximately vanish on the data points (red dots) in VCA. (Right panel) Vanishing polynomials and data knots (blue circles) obtained by the proposed method, which almost exactly pass the data knots.

be viewed as a system to be satisfied by the input data points; as such, it holds the nonlinear structure of the data. Exploiting these properties, Livni et al. (2013) proposed Vanishing Component Analysis (VCA), which extracts the compact nonlinear features of the data to be classified, and enables training by a simple linear classifier. The vanishing ideal has also identified the underlying dynamics in limited observations (Torrente 2008; Robbiano and Abbott 2010; Kera and Hasegawa 2016). In these applications, monomials consisting of models are inferred from the vanishing ideal of the observations.

As the available data in many applications are exposed to noise, it has been common to compute polynomials that approximately rather than exactly vanish on the data to avoid the overfitting problem (Heldt et al. 2009; Fassino 2010; Livni et al. 2013; Limbeck 2013). However, this approximation destroys the sound algebraic structure of the vanishing ideal. For instance, as shown in the left panel of Fig. 1, a set of approximately vanishing polynomials is no longer an algebraic system because it possesses no common roots. In other words, there is a tradeoff between preserving the algebraic soundness of the vanishing ideal and avoiding overfitting to noise. Such a tradeoff has not been explicitly consid-

ered in existing work.

In the present paper, we newly deal with this tradeoff by addressing a vanishing ideal that is well tolerant to noisy data while pursuing to preserve a sound algebraic structure. Specifically, we introduce a new task that jointly discovers a set of polynomials and *summarized* data points (called data knots) from the input data. As shown in the right panel of Fig. 1, the polynomials in this task avoid overfitting because they approximately vanish on the original data, and preserve the better algebraic structure of a vanishing ideal than those by VCA because they intersect (and vanish) almost exactly at the data knots.

To our knowledge, we present the first computation of a vanishing ideal that handles both the given fixed points and the jointly updated points. As the noise level increases, the vanishing ideal of the given fixed data performs less accurately because it requires a coarser approximation. If the approximation remains fine, the computed vanishing polynomials (especially the higher-degree polynomials) will overfit to noise. To circumvent this problem, the proposed method conjointly computes the vanishing polynomials and the data knots. Assuming that lower-degree polynomials are less overfitted to noise and better preserve the data structure, our method generates the vanishing polynomials from lower degree to higher degree while updating the data knots at each degree. Consequently, overfitting by the higher-degree polynomials is avoided. The data knots are updated by a nonlinear regularization, which can be regarded as a generalization of the Mahalanobis distance, which is commonly used in the metric learning (Bellet, Habrard, and Sebban 2013). In a theoretical analysis, we also guarantee that the proposed algorithm terminates and that in the extreme case, the generated polynomials exactly (rather than almost) vanish on the data knots.

Experiments confirmed that our methods generate fewer and lower-degree polynomials than an existing vanishing ideal-based approach. In different classification tasks, our methods obtained a much more compact nonlinear feature than VCA, reducing the test runtime. To verify that the obtained data knots well represent the original data, we trained a $k$-nearest neighbor classifier. Although the data knots are far fewer than the original points, the classification accuracy of the nearest neighbor classifier was comparable to that of the baseline methods in most cases.

## 2 Related Work

Although our work addresses a new problem, several works are closely related to the present study.

Fassino (2010) proposed a Numerical Buchberger–Möller (NBM) algorithm that computes approximately vanishing polynomials for the vanishing ideal of input data $X$. NBM requires that each polynomial exactly vanishes on a set of nearby data points $X^\varepsilon$ (called an admissible perturbation) from $X$ up to a hyperparameter $\varepsilon$. However, because the admissible perturbations can differ among the approximately vanishing polynomials, NBM does not generally output vanishing polynomials that vanish on the same points. In addition, NBM does not specify the admissible perturbations, but

only checks the sufficient condition of the existence of such points.

Abbott, Fassino, and Torrente (2007) addressed a new task of *thinning out* the data points as a preprocessing for computing vanishing ideals afterward. Similar to clustering, their approach computes the empirical centroids of the input data. As thinning out and the vanishing-ideal computation are performed independently, the empirical centroids do not necessarily result in compact, lower-degree vanishing polynomials. In contrast, our method conjointly performs the thinning out and vanishing-ideal computation in a single framework. Note that both methods by Abbott, Fassino, and Torrente and by us aim to reduce data points by summarizing nearby points, which are different from the clustering tasks that aim to group even distant points according to their categories.

## 3 Preliminaries

**Definition 1.** (Vanishing Ideal) Given a set of $d$-dimensional data points $X$, the vanishing ideal of $X$ is a set of $d$-variate polynomials that take zero values, (*i.e.*, vanish) at all points in $X$. Formally, the vanishing ideal is defined as Eq. (1).

**Definition 2.** (Evaluation vector, Evaluation matrix) Given a set of data points $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ with $N$ samples, the evaluation vector of polynomial $f$ is defined as

$$f(X) = (f(\mathbf{x}_1) \quad f(\mathbf{x}_2) \quad \cdots \quad f(\mathbf{x}_N))^\top \in \mathbb{R}^N.$$

Given a set of polynomials $F = \{f_1, f_2, ..., f_s\}$, its evaluation matrix is $F(X) = (f_1(X)\ f_2(X)\ \cdots\ f_s(X)) \in \mathbb{R}^{N \times s}$.

**Definition 3.** ($\varepsilon$-vanishing polynomial) Given a set of data points $X$, a polynomial $g$ is called an $\varepsilon$-vanishing polynomial on $X$ when the norm of its evaluation vector on $X$ is less than or equal to $\varepsilon$, *i.e.*, $\|g(X)\| \leq \varepsilon$, where $\|\cdot\|$ denotes the Euclidean norm of a vector. Otherwise, $g$ is called an $\varepsilon$-nonvanishing polynomial.

In the vanishing ideal concept, polynomials are identified with their evaluation on the points; that is, a polynomial $f$ is mapped to an $N$-dimensional vector $f(X)$. Two polynomials $f$ and $\tilde{f}$ are considered to be equivalent if they have the same evaluation vector, *i.e.*, $f(X) = \tilde{f}(X)$. When $f(X) = \mathbf{0}$, $f$ is a vanishing polynomial on $X$.

Another important relation between polynomials and their evaluations is that the sum of the evaluation of polynomials equals to the evaluation of the sum of the polynomials. More specifically, the following relation holds for a set of polynomials $G = \{g_1, ..., g_s\}$ and coefficient vectors $\mathbf{c} \in \mathbb{R}^s$:

$$G(X)\mathbf{c} = (G\mathbf{c})(X),$$

where $G\mathbf{c} = \sum_{i=1}^{s} c_i g_i$ defines the inner product between a set $G$ and a coefficient vector $\mathbf{c}$, with $c_i$ being the $i$-th entry of $\mathbf{c}$. Similarly, $G\mathbf{C} = \{G\mathbf{c}_1, ..., G\mathbf{c}_k\}$ defines the multiplication of $G$ by a matrix $\mathbf{C} = (\mathbf{c}_1, \cdots, \mathbf{c}_k)$. These special inner products will be used hereafter.

## 4 Proposed Method

Given a set of data points $X_0$, we seek a set of polynomials $G$ and data knots $Z$ such that the polynomials in $G$ approximately vanish on $X_0$ and almost exactly vanish on $Z$.

**Algorithm 1** FindBasis

**Require:** $\tilde{C}_t, F^{t-1}, Z, X_0, \varepsilon, \delta$
**Ensure:** $G_t, F_t$
1: # Compute residual space
2: $C_t = \tilde{C}_t - F^{t-1}F^{t-1}(Z)^\dagger \tilde{C}_t(Z)$
3:
4: # SVD for evaluation matrix on $X_0$
5: $C_t(X_0) = \mathbf{U}_0 \mathbf{D}_0 \mathbf{V}_0^\top = \mathbf{U}_0 \mathbf{D}_0 [\tilde{\mathbf{V}}_0 \ \mathbf{V}_0^\varepsilon]^\top$
6:
7: # SVD for evaluation matrix on $Z$
8: $C_t(Z)\mathbf{V}_0^\varepsilon = \mathbf{U}\mathbf{D}\mathbf{V}^\top = \mathbf{U}\mathbf{D}[\tilde{\mathbf{V}} \ \mathbf{V}^\delta]^\top$
9: $C_t(Z)\tilde{\mathbf{V}}_0 = \hat{\mathbf{U}}\mathbf{E}\mathbf{W}^\top = \hat{\mathbf{U}}\tilde{\mathbf{E}}[\tilde{\mathbf{W}} \ \mathbf{W}^\delta]^\top$
10:
11: $G_t = C_t \mathbf{V}_0^\varepsilon \mathbf{V}^\delta$
12: $F_t = C_t \tilde{\mathbf{V}}_0 \tilde{\mathbf{W}} \cup C_t \mathbf{V}_0^\varepsilon \tilde{\mathbf{V}}$
13: # Divide each polynomial in $F_t$ by the norm of its evaluation vectors on $Z$
14: **return** $G_t, F_t$

Specifically,

$$\forall g \in G, \|g(X_0)\| \le \varepsilon, \|g(Z)\| \le \delta, \qquad (2)$$

where $\varepsilon, \delta$ are hyperparameters of the error tolerance that requires $g$ to be $\varepsilon$-vanishing on $X_0$ and $\delta$-vanishing on $Z$. In our case, $\delta$ is set much smaller than $\varepsilon$.

Like many other methods that construct a vanishing ideal basis, our polynomial construction starts from a degree-0 polynomial (*i.e.*, constant), and increments the polynomial degree in successive data fittings. For each degree $t$, the proposed method iterates the following two steps:

1. Compute a set of approximately vanishing polynomials $G_t$ and a set of nonvanishing polynomials $F_t$.

2. Update the data knots such that the approximately vanishing polynomials in $G^t = \cup_{i=1}^t G_i$ more closely vanish.

These steps constitute our technical contributions to the field. In the former step, we newly introduce a vanishing polynomial construction that handles two set of points $X_0, Z$ with different approximation scales $\varepsilon, \delta$. In the latter step, we introduce a nonlinear regularization term that preserves the nonlinear structure of data while updating $Z$.

In the following subsections, we first describe the polynomial construction part (Step 1; Section 4.1) and the data-knotting part (Step 2; Section 4.2). To improve the quality of the vanishing polynomials and data knots, we propose an iterative update framework in Section 4.3. Finally, we present our whole algorithm in Section 4.4.

## 4.1 Vanishing polynomial construction

Given a set of data points $X_0$ and tentative data knots $Z$, we here construct two polynomial sets $G_t$ and $F_t$ of degree $t \ge 1$. $G_t$ is the set of degree-$t$ polynomials that are $\varepsilon$-vanishing on $X_0$ and $\delta$-vanishing on $Z$. $F_t$ is a set of polynomials that are not $\delta$-vanishing on $Z$. Note that at this moment, we have $G^{t-1} = \cup_{i=0}^{t-1} G_i$ and $F^{t-1} = \cup_{i=0}^{t-1} F_i$.

To obtain $G_t$ and $F_t$, we first generate candidate degree-$t$ polynomials $C_t$ based on the VCA framework. Multiplying

all possible combinations of polynomials between $F_1$ and $F_{t-1}$, we construct $\tilde{C}_t = \{fg \mid f \in F_1, g \in F_{t-1}\}$. Then, $C_t$ is generated as the residual polynomials of $\tilde{C}_t$ with respect to $F^{t-1}$ and their evaluation vectors.

$$C_t = \tilde{C}_t - F^{t-1}\left(F^{t-1}(Z)^\dagger \tilde{C}_t(Z)\right),$$

where $A^\dagger$ denotes the pseudo-inverse matrix of $A$. Note that the second term is the inner product between $F^{t-1}$ and $F^{t-1}(Z)^\dagger \tilde{C}_t(Z)$, and not the evaluation of $F^{t-1}$ on $F^{t-1}(Z)^\dagger \tilde{C}_t(Z)$. This step calculates the residual column space of $\tilde{C}_t(Z)$ that is orthogonal to that of $F^{t-1}(Z)$.

$$C_t(Z) = \tilde{C}_t(Z) - F^{t-1}(Z)\left(F^{t-1}(Z)^\dagger \tilde{C}_t(Z)\right).$$

In short, when evaluated on the data knots $Z$, the column spaces of the residual polynomials $C_t$ and the above polynomials are orthogonal.

After generating $C_t$, degree-$t$ nonvanishing polynomials $F_t$ and vanishing polynomials $G_t$ are constructed by applying singular value decomposition (SVD) to $C_t(X_0)$:

$$C_t(X_0) = \mathbf{U}_0 \mathbf{D}_0 \mathbf{V}_0^\top,$$

where $\mathbf{U}_0 \in \mathbb{R}^{N \times N}$ and $\mathbf{V}_0 \in \mathbb{R}^{|C| \times |C|}$ are orthogonal matrices, and $D \in \mathbb{R}^{N \times |C|}$ contains singular values only along its diagonal. Multiplying both sides by $\mathbf{V}_0$ and focusing on the $i$-th column, we obtain

$$C_t(X_0)\mathbf{v}_i = \sigma_i \mathbf{u}_i,$$

where $\mathbf{v}_i$ and $\mathbf{u}_i$ are the $i$-th columns of $\mathbf{V}_0$ and $\mathbf{U}_0$ respectively, and $\sigma_i$ is the $i$-th diagonal entry of $\mathbf{D}_0$. Moreover, when $\sigma_i \le \varepsilon$, we have

$$\|g_i(X_0)\| = \|(C_t\mathbf{v}_i)(X_0)\| = \|C_t(X_0)\mathbf{v}_i\| = \|\sigma_i\mathbf{u}_i\| = \sigma_i,$$

meaning that polynomial $g_i := C_t\mathbf{v}_i$ is an $\varepsilon$-vanishing polynomial on $X_0$. We can regard $\mathbf{v}_i$ as a coefficient vector of the linear combination of the polynomials in $C_t$. Let us denote $\mathbf{V}_0 = (\tilde{\mathbf{V}}_0 \ \mathbf{V}_0^\varepsilon)$, where $\tilde{\mathbf{V}}_0$ and $\mathbf{V}_0^\varepsilon$ are matrices corresponding to the singular values exceeding $\varepsilon$ and not exceeding $\varepsilon$, respectively. For any unit vector $\mathbf{v} = \mathbf{V}_0^\varepsilon \mathbf{p} \in \text{span}(\mathbf{V}_0^\varepsilon)$ expressed in terms of a unit vector $\mathbf{p}$, a polynomial $g = C_t\mathbf{v}$ satisfies

$$\|g(X_0)\| = \|C_t(X_0)\mathbf{v}\| = \|G_t(X_0)\mathbf{p}\| \le \sigma_{\varepsilon_{\max}}\|\mathbf{p}\| = \sigma_{\varepsilon_{\max}},$$

meaning that $g = C_t\mathbf{v}$ is an $\varepsilon$-vanishing polynomial, where $\sigma_{\varepsilon_{\max}}$ is the largest singular value not exceeding $\varepsilon$.

Next, we construct $\delta$-vanishing polynomials in the above polynomial space. This constraint problem is formulated as follows:

$$\min_{(\hat{\mathbf{V}}^\delta)^\top \hat{\mathbf{V}}^\delta = \mathbf{I}} \|C_t(Z)\hat{\mathbf{V}}^\delta\|_{\mathrm{F}}, \quad \text{s.t. } \text{span}(\hat{\mathbf{V}}^\delta) \subset \text{span}(\mathbf{V}_0^\varepsilon),$$

where $\| \cdot \|_{\mathrm{F}}$ denotes the Frobenius norm of a matrix. From the discussion above, it can be reformulated as,

$$\min_{\mathbf{P}^\top \mathbf{P} = \mathbf{I}} \|C_t(Z)\mathbf{V}_0^\varepsilon \mathbf{P}\|_{\mathrm{F}},$$

since the column space of $\mathbf{V}_0^\varepsilon$ spans the coefficient vectors of $\varepsilon$-vanishing polynomials on $X_0$. This problem can be simply solved by applying SVD to $C_t(Z)\mathbf{V}_0^\varepsilon$ and selecting the right singular vectors corresponding to the singular values not exceeding $\delta$. Supposing that SVD yields $C_t(Z)\mathbf{V}_0^\varepsilon = \mathbf{UDV}^\top$, we denote $\mathbf{V} = (\tilde{\mathbf{V}} \; \mathbf{V}^\delta)$, where $\tilde{\mathbf{V}}$ and $\mathbf{V}^\delta$ are matrices corresponding to the singular values exceeding $\delta$ and not exceeding $\delta$, respectively. The polynomials $G_t$ that are $\varepsilon$-vanishing on $X$ and $\delta$-vanishing on $Z$ are then obtained as follows:

$$G_t = C_t \mathbf{V}_0^\varepsilon \mathbf{V}^\delta.$$

$F_t$ is constructed similarly, but consists of two parts.

$$F_t = C_t \tilde{\mathbf{V}}_0 \tilde{\mathbf{W}} \cup C_t \mathbf{V}_0^\varepsilon \tilde{\mathbf{V}},$$

where $\tilde{\mathbf{W}}$ is the counterpart of $\tilde{\mathbf{V}}$ for $C_t(Z)\tilde{\mathbf{V}}_0$. The left-part polynomials are $\varepsilon$-nonvanishing on $X_0$ and $\delta$-nonvanishing on $Z$, whereas the right-part polynomials are $\varepsilon$-vanishing on $X_0$ but $\delta$-nonvanishing on $Z$. Following the VCA framework, the polynomials in $F_t$ are rescaled by their evaluation vectors on $Z$ to maintain numerical stability. The main generating procedures of $G_t$ and $F_t$ are summarized in Alg. 1. For simplicity, we omit the step that constructs $\tilde{C}_t$.

## 4.2 Data knotting

Given a set of points $X$ and a set of vanishing polynomials $G^t$ up to degree $t$, we seek new data points $Z$ for which the polynomials in $G^t$ more exactly vanish on $Z$ while preserving the nonlinear structure of $X$. In the present paper, we refer to these $Z$ as data knots and the searching process as data knotting (by analogy to ropes). As described later, we iteratively update the data knots, so we designate the current and updated data knots as $X$ and $Z$, respectively.

First, we intuitively illustrate the concept in a simple case. Let $\mathbf{X}$ to be a matrix whose $i$-th row corresponds to the $i$-th point of $X$. Applying SVD, we have

$$\mathbf{X} = \mathbf{UDV}^\top = \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top + \mathbf{U}_\varepsilon \mathbf{D}_\varepsilon \mathbf{V}_\varepsilon^\top, \qquad (3)$$

where $\mathbf{U}, \mathbf{V}$ are orthonormal matrices, and $\mathbf{D}$ has the singular values only along its diagonal. The former and latter terms define the principal and minor variances in the data, respectively, regarding singular values exceeding $\varepsilon$ and the rest. Note that when $\mathbf{X}$ is mean-centralized, $C_1(X) = \mathbf{X}$. We seek new data points for which the minor variance ideally vanishes to a zero matrix while the principal variance is preserved. In this linear case, the new points are simply defined as $\mathbf{Z} = \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top$, where $\mathbf{Z}$ for $Z$ is defined identically to $\mathbf{X}$.

However, discovering $Z$ in nonlinear cases is not straightforward. In the degree-$t$ case, Eq. (3) becomes

$$C_t(X) = \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top + \mathbf{U}_\varepsilon \mathbf{D}_\varepsilon \mathbf{V}_\varepsilon^\top. \qquad (4)$$

As in the linear case, we seek $Z$ such that $C_t(Z) = \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top$. To this end, we address the following minimization problem:

$$\min_Z \left\| C_t(Z) - \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top \right\|_{\mathrm{F}}. \qquad (5)$$

Multiplying Eq. (4) by $\mathbf{V}$ and $\mathbf{V}_\varepsilon$ respectively, we obtain

$$F_t(X) = F_t(Z),$$
$$G_t(Z) = \mathbf{O}.$$

In the derivation, we used $\tilde{\mathbf{V}}^\top \tilde{\mathbf{V}} = \mathbf{I}$, $\mathbf{V}_\varepsilon^\top \mathbf{V}_\varepsilon = \mathbf{I}$, and $\tilde{\mathbf{V}}^\top \mathbf{V}_\varepsilon = \mathbf{O}$, where $\mathbf{I}$ is an identity matrix and $\mathbf{O}$ is a zero matrix. From these relations, Eq. (5) is reformulated as,

$$\min_Z \|G_t(Z)\|_{\mathrm{F}} + \lambda_t \|F_t(Z) - F_t(X)\|_{\mathrm{F}}, \qquad (6)$$

where $\lambda_t$ is a hyperparameter. It can be easily shown,

$$C_t(Z) = \tilde{\mathbf{U}}\tilde{\mathbf{D}}\tilde{\mathbf{V}}^\top \iff F_t(X) = F_t(Z), G_t(Z) = \mathbf{O}.$$

See the supplementary material for the proof. Note that the optimization problem of Eq. (6) can be factorized into a subproblem on each data point.

$$\min_{\mathbf{z}_i} \|G_t(\mathbf{z}_i)\|_{\mathrm{F}} + \lambda_t \|F_t(\mathbf{z}_i) - F_t(\mathbf{x}_i)\|_{\mathrm{F}}, \qquad (7)$$

where $\mathbf{z}_i, \mathbf{x}_i$ are the $i$-th data point of $Z, X$, respectively. Considering the polynomials up to degree $t$, Eq. (7) becomes

$$\min_{\mathbf{z}_i} \sum_{k=1}^t \|G_k(\mathbf{z}_i)\|_{\mathrm{F}} + \sum_{k=1}^t \lambda_k \|F_k(\mathbf{z}_i) - F_k(\mathbf{x}_i)\|_{\mathrm{F}}. \qquad (8)$$

The first term encourage the polynomials in $G^t$ to vanish on $\mathbf{z}_i$, and the second term constrains the $\mathbf{z}_i$ to nearby $\mathbf{x}_i$ with respect to $F^t$. This formulation provides an interesting insight:

**Remark 1.** The second term of Eq. (8) is a regularization term that is equivalent to a nonlinear generalization of the Mahalanobis distance.

The Mahalanobis distance between two data points $\mathbf{x}$ and $\mathbf{y}$ of a mean-centralized matrix $\mathbf{X}$ is a generalization of the Euclidean distance, where each variable is normalized by its standard deviation, $i.e.$, $d(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^\top \Sigma^\dagger (\mathbf{x} - \mathbf{y})}$, where $\Sigma = \mathbf{X}^\top \mathbf{X}$ is an empirical covariance matrix. The remark above holds because our $F_t$ describes the nonlinear principal variance of the current data knots $X$. For simplicity, we consider only the linear case $t = 1$. In this case, $C_1(X)$ is a mean-centralized $X$ because it is the residual with respect to $F_0 = 1/\sqrt{|X|}$. Adopting the notations in lines 8 and 9 of Alg. 1, we let $\tilde{\mathbf{E}}$ and $\tilde{\mathbf{D}}$ be submatrices of $\mathbf{E}$ and $\mathbf{D}$ corresponding to singular values larger than $\delta$. The distance between two points $\mathbf{x}, \mathbf{y}$ of the mean-centralized $X$ is then calculated as

$$\|F_1(\mathbf{x}) - F_1(\mathbf{y})\|^2$$
$$= \left\| \mathbf{x}^\top \left( \tilde{\mathbf{V}}_0 \tilde{\mathbf{W}} \tilde{\mathbf{E}}^\dagger \quad \mathbf{V}_0^\varepsilon \tilde{\mathbf{V}} \tilde{\mathbf{D}}^\dagger \right) - \mathbf{y}^\top \left( \tilde{\mathbf{V}}_0 \tilde{\mathbf{W}} \tilde{\mathbf{E}}^\dagger \quad \mathbf{V}_0^\varepsilon \tilde{\mathbf{V}} \tilde{\mathbf{D}}^\dagger \right) \right\|^2,$$
$$= (\mathbf{x} - \mathbf{y})^\top \Sigma^{-1} (\mathbf{x} - \mathbf{y}),$$

where

$$\Sigma^{-1} = \tilde{\mathbf{V}}_0 \tilde{\mathbf{W}} (\tilde{\mathbf{E}}^\top \tilde{\mathbf{E}})^{-1} \tilde{\mathbf{W}}^\top \tilde{\mathbf{V}}_0^\top + \mathbf{V}_0^\varepsilon \tilde{\mathbf{V}} (\tilde{\mathbf{D}}^\top \tilde{\mathbf{D}})^{-1} \tilde{\mathbf{V}}^\top \mathbf{V}_0^{\varepsilon \top}.$$

A straightforward calculation shows that $\Sigma$ is the principal variance of the empirical covariance matrix $C_1(X)^\top C_1(X)$.

Similarly, in nonlinear case $t > 1$, $\|F_t(\mathbf{x}) - F_t(\mathbf{y})\|$ is a Mahalanobis distance with respect to principal variance of $C_t(X)^\top C_t(X)$. Therefore, the regularization term in Eq. (8) can be regarded as a generalized Mahalanobis distance in nonlinear cases (details are provided in the supplementary material).

To optimize Eq. (8), quasi-Newton method with numerical gradient calculation was adopted in the present paper. We restricted the $F_k$ to lower-degree polynomials by assuming that lower-degree structures holds sufficiently good structure of data. Specifically, we took into account the regularization terms up to the degree where the first vanishing polynomial is found.

### 4.3 Exact Vanish Pursuit

Our original goal to discover the $\delta$-vanishing polynomials on the data knots cannot be achieved by simply applying the data knotting to a fixed set of polynomials. In some cases (e.g., when three polynomials never intersect at any one instant), there may be no data knots on which the given polynomials sufficiently vanish. To resolve this problem, we introduce an iterative framework that alternatively updates data knots and polynomials (Alg. 2).

Let us construct the degree-$t$ polynomials $G_t$ and $F_t$. At this moment, we have polynomial sets with degree less than $t$, $G^{t-1}$ and $F^{t-1}$, and tentative data knots $Z$. Introducing $\eta > \delta$, we repeat the following steps.

1. Fixing $Z$, update $G_t$ and $F_t$ by Alg. 1. In this step, the polynomials in $G_t$ are $\varepsilon$-vanishing on $X_0$ and $\eta$-vanishing on $Z$.

2. Fixing $F^t, G^t$, update the data knots $Z$ by solving Eq. (8).

3. Decrease $\eta$.

This iteration terminates before $\eta$ reaches $\delta$ when $G^t$ becomes a set of $\delta$-vanishing polynomials, $G_t$ becomes an empty set. The parameter $\eta$ approaches to $\delta$ over the iterations, and when $\eta = \delta$ then all the polynomials $G_t$ are $\delta$-vanishing on $Z$ and the algorithm terminates. Note that in this case the polynomials in $G^{t-1}$ may be no longer $\delta$-vanishing on $Z$ because $Z$ has been updated. The next subsection introduces the reset framework, which resolves this situation. The way of reducing $\eta$ can affect the algorithm result. In the present study, we decrease $\eta$ in a pragmatic way; we introduce a cooling parameter $\gamma < 1$. Generally, $\eta$ was updated by $\gamma\eta$, but when the largest norm of the evaluation vector of $g \in G$, then $\eta$ was updated by that norm, i.e., $\eta = \min(\gamma\eta, \max_{g \in G} \|g(Z)\|)$. The proper decrease of $\eta$ is left for future work.

The iterative framework introduced in this section is summarized in Alg. 2. In this subroutine, the orders of the data knotting and polynomial construction are reversed for easy implementation in the latter sections.

### 4.4 Algorithm

This section describes the overall algorithm of the proposed method (Alg. 3). The input are data points $X_0$, the error tolerances $\varepsilon, \delta$, and the regularization weight $\lambda$. The algorithm outputs a set of polynomials $G$ and data knots $Z$ for which

---

**Algorithm 2** ExactVanishPursuit

**Require:** $G^t, F^t, \tilde{C}_t, Z, X_0, \varepsilon, \eta, \delta, \lambda$
**Ensure:** $G_t, F_t$
1: **while** $\eta > \delta$ and $G_t$ is not empty **do**
2:     Update $Z$ by solving Eq. (8);
3:     **if** $\forall g \in G^t, \|g(Z)\| \le \delta$ **then**
4:         break
5:     **end if**
6:     Decrease $\eta$;
7:     $G_t, F_t = \text{FindBasis}(\tilde{C}_t, F^{t-1}, Z, X_0, \varepsilon, \eta)$
8: **end while**
9: **return** $G_t, F_t$

---

polynomials in $G$ are $\varepsilon$-vanishing on $X_0$ and $\delta$-vanishing on $Z$.

As it proceeds, the algorithm increments the degree of the polynomials. The degree-0 polynomial sets are initialized to $G_0 = \{\}, F_0 = \{f(\cdot) = 1/\sqrt{|X|}\}$, and the initial data knots $Z$ are set to $X_0$. We also introduce an error tolerance parameter $\eta$, which is set to $\eta = \varepsilon$. Although we aim to discover the $\delta$-vanishing polynomials on $Z$, we first consider the $\eta$-vanishing polynomials on $Z$, which is updated rather than fixed. $\eta$ is gradually decreased throughout the iterations, and eventually reaches $\delta$, thereby obtaining $\delta$-vanishing polynomials.

At degree-$t$, the algorithm proceeds through the following four steps: (1) generate $G_t$ and $F_t$ by Alg. 1, where $G_t$ is a set of polynomials that are $\varepsilon$-vanishing on $X_0$ and $\eta$-vanishing on $Z$; (2) update $G_t$, $F_t$, and $Z$ by Alg. 2 such that the polynomials in $G_t$ become $\delta$-vanishing on $Z$; (3) generate degree-$(t + 1)$ candidate polynomials for the next iteration; (4) check the termination conditions (*reset* or advance to the next degree). Reset restores all variables except $Z$ and $\eta$ to the $t = 1$ stage. A reset is performed if there is a $\delta$-nonvanishing polynomial in $G$ and the algorithm cannot proceed to the next degree, i.e., when $C_t$ is empty. The reset system feedbacks the results of higher-degree polynomials to lower-degree ones via the data knots $Z$. To our knowledge, the reset system is unique to our method; all of the existing methods appear to greedily construct the polynomials from lower to higher degrees.

Termination is guaranteed when $\eta$ reaches $\delta$. To prove it, first note that when $\eta = \delta$, no data knotting occurs so $Z$ is fixed. To describe arbitrary polynomials on $Z$, we need collect $|Z|$ linearly independent polynomials in $F$, because the polynomials are associated with $\mathbb{R}^{|Z|}$-dimensional vectors. In Alg. 1, the column space of the evaluations of candidate polynomials $C_t$ is orthogonal to that of $F$ on $Z$. By its construction, the column space of $F_t(Z)$ approximately spans the column space of $C_t(Z)$. When $|F_t| = 0$, the algorithm terminates; otherwise, the rank of $F$ is strictly increased by appending $F_t$. Therefore, the rank of $F$ reaches $|Z|$ after a finite number of steps, and the algorithm terminates. As $\eta = \delta$, all polynomials in $G$ are $\delta$-vanishing polynomials on $Z$.

Note that the output $G$ is not necessarily a basis of the vanishing ideal for $Z$ because polynomials that are $\delta$-vanishing

**Algorithm 3** Main

**Require:** $X_0, \varepsilon, \delta, \lambda$
**Ensure:** $G, Z$
 1: # Initializetoin
 2: $G = \{\}, F = F_0 = \{f(\cdot) = 1/\sqrt{|X|}\}$
 3: $\tilde{C}_1 = Z = X_0$
 4: $\eta = \varepsilon, t = 1$
 5: **loop**
 6:    # Compute bases of degree-$t$ polynomials
 7:    $G_t, F_t = \text{FindBasis}(\tilde{C}_t, F, Z, X_0, \varepsilon, \eta)$
 8:    $G_t, F_t = \text{ExactVanishPursuit}(G \cup G_t, F \cup F_t,$
 9:        $\tilde{C}_t, Z, X_0, \varepsilon, \eta, \lambda)$
10:    $G = G \cup G_t, F = F \cup F_t$
11:    $C_t = \{f_t f_1; f_t \in F_t, f_1 \in F_1\}$
12:
13:    **if** $C_t$ is empty **then**
14:       **if** $\forall g \in G, \|g(Z)\| \le \delta$ **then**
15:          **return** $Z, G$
16:       **else**
17:          # Reset to degree 1
18:          $t = 1$
19:          $G = \{\}, F = \{f(\cdot) = 1/\sqrt{Z}\}$
20:          $C_1 = Z$
21:          Decrease $\eta$;
22:       **end if**
23:    **else**
24:       $t = t + 1$
25:    **end if**
26: **end loop**

on $Z$ but $\varepsilon$-nonvanishing on $X_0$ are excluded from $G$. This result is reasonable because the polynomials in $G$ do not well approximate the original data. In some cases, however, we require a basis of the vanishing ideal. Such a basis can be generated by applying existing basis generation methods such as VCA to small data knots $Z$, which is much less computationally costly than applying to $X_0$.

# 5 Results

In this sections, we demonstrate that our method discovers a compact set of low-degree polynomials and a few data knots that well represent the original points. The proposed method exhibits both noise tolerance and good preservation of the algebraic structure. We first illustrate the vanishing polynomials and data knots obtained on simple datasets as qualitative evaluation. In the next classification task, we show that the polynomials output by out method avoid overfitting and hold the useful nonlinear feature of data as observed in VCA. Finally, we evaluate the representativeness of the data knots by training $k$-nearest neighbor classifiers in the classification tasks. Note that classification tasks are adopted to measure how well the proposed method can hold nonlinear structure of data. The proposed method is not specially tailored for classifications, and it can also contribute to other tasks where vanishing ideal based approach has been introduced.

## 5.1 Illustration with simple data

We applied our method and VCA with the same error tolerance to simple data perturbed by noise: three blobs with different variances (60 points, 30% noise on one blob; 5% noise on the remaining blobs), a single circle (30 samples, 5% noise), and a pair of concentric circles (50 samples, 2% noise). Here $n\%$ noise denotes zero-mean Gaussian noise with $n$-standard deviation. The blobs were generated by adding noise at three distinct points. As shown in the Fig. (2), each set of polynomials obtained by our method exactly vanishes on the data knots, and approximately vanishes on the input data points, while those by VCA only coarsely intersect with each other. In Figs. 2(a) and 2(b), one blob has much larger variance than the others. Whereas each blob with small variance is represented by a single data knot, the very noisy blob is represented by two knots; consequently, polynomials obtained by our method almost exactly vanish on the knots and approximately vanish over the whole blob. In contrast, while the polynomials obtained by VCA are similar to those by ours, they intersect with each other much more coarsely. In Figs. 2(c), 2(d), 2(e), and 2(f), both our method and VCA discovered the lowest algebraic structures (a circle and a pair of concentric circles). In Fig. 2(c), our method only outputs a circle since other polynomials that approximately vanish on the original data do not sufficiently vanish on the data knots. In Fig. 2(e), some polynomials discovered by our method are different from those by VCA for better preserving the algebraic structure. Note that the same error tolerance is used for both our method and VCA. Thus, the polynomials obtained by our method still approximately vanish on the original points as those by VCA.

## 5.2 Compact lower-degree feature extraction

The vanishing polynomials obtained by our method compactly hold the original data structure. To show this, we compare our method with VCA in four classification tasks. Bothe methods were tested by Python implementation on a workstation with four processors and 8GB memory. Both the proposed method and VCA adopt the feature-extraction method of Livni et al. (2013):

$$\mathcal{F}(\mathbf{x}) = \Big(\cdots, \underbrace{\left|g_i^{(1)}(\boldsymbol{x})\right|, \cdots, \left|g_i^{(|G_i|)}(\boldsymbol{x})\right|}_{G_i(\boldsymbol{x})^\top}, \cdots\Big)^\top,$$

where $G_i = \{g_i^{(1)}, ..., g_i^{(|G_i|)}\}$ denote the computed vanishing polynomials of the $i$-th class. By its construction, the feature $\mathcal{F}(\mathbf{x})$ of sample $\mathbf{x}$ in the $i$-th class should be a vector whose entries are approximately zero in the $G_i$ part and non-zero elsewhere. The classifier for the proposed method and VCA was a linear Support Vector Machine (Cortes and Vapnik 1995). The datasets were downloaded from UCI machine learning repository (Lichman 2013). The hyperparameters were determined by 3-fold cross validation and the results were averaged over ten independent runs. In each run, the datasets were randomly split into training (60%) and test (40%) datasets.

The classification results are summarized in Table 1. The proposed method achieved comparable classification accu-
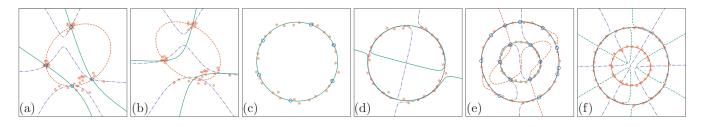
Figure 2: A set of vanishing polynomials and data knots (blue circles) output by the proposed method (a,c,e) and VCA (b,d,f) for simple data exposed to noise (red dots). The input data are (a,b) three blobs with different variance (c,d) a single circle, (e,f) two concentric circles. To enhance visibility, not all of the polynomials are shown.

Table 1: Classification results

|  | Accuracy [%] | | | | Test runtime [sec] | | #features | | #degree | |
|---|---|---|---|---|---|---|---|---|---|---|
|  | Proposed | VCA | Proposed-hd | VCA-hd | Proposed | VCA | Proposed | VCA | Proposed | VCA |
| Iris | 0.96 | 0.95 | 0.75 | 0.59 | 3.6e-4 | 6.6e-4 | 14.9 | 62.8 | 1.5 | 2.0 |
| Wine | 0.98 | 0.98 | 0.94 | 0.67 | 7.6e-4 | 2.1e-3 | 100.5 | 592.9 | 1.7 | 2.3 |
| Vehicle | 0.80 | 0.80 | 0.53 | 0.60 | 1.6e-3 | 6.7e-2 | 121.5 | 4147.9 | 1.6 | 2.5 |
| Vowel | 0.92 | 0.93 | 0.80 | 0.35 | 1.5e-3 | 2.3e-3 | 191.0 | 267.6 | 1.5 | 1.8 |

Table 2: $k$-nearest neighbor classification

|  | Data knots | $k$-means centroids | Original points | Knotting ratio |
|---|---|---|---|---|
| Iris | 0.95 | 0.95 | 0.94 | 0.08 |
| Wine | 0.95 | 0.97 | 0.95 | 0.20 |
| Vehicle | 0.60 | 0.61 | 0.68 | 0.05 |
| Vowel | 0.76 | 0.79 | 0.96 | 0.53 |

racy to VCA, with much more compact features. The dimensions of the feature vectors obtained by our method were only 3–70% those of the VCA feature vectors. The degree of the discovered polynomials was also lower in the proposed method. Consequently, the test runtime was lower in our method than in VCA. This result suggests that the proposed method well preserves the data structure even after data knotting. As we insisted in the introduction, higher-degree polynomials can be sensitive to noisy data under traditional vanishing polynomial construction with fixed data points. To confirm this, we also evaluated the methods by restricting the polynomials for feature extraction to half of them in the higher-degree part (Proposed-hd and VCA-hd). As can be seen from Table 1, the higher-degree polynomials by our method lead to much higher classification accuracy than those by VCA, which suggests that our method provides noise-tolerant polynomials even in relatively higher degree while VCA does not. An exception is the result for the Vehicle dataset. In this case, our method provided greatly compact features (only 2% dimension of those by VCA). Thus the 50 % restriction at Proposed- may remove important features that contribute much to the accuracy.

## 5.3 Evaluating data knots

In the proposed framework, data knotting greatly reduces the number of original noisy data points, enabling lower-

degree vanishing polynomials. Here we evaluate the representativeness of the data knots with the classification accuracy of the $k$-nearest neighbor classifier trained with data knots. As baselines, we also trained classifiers with $k$-means centroids and the original points as baselines. The number of centroids for $k$-means clustering is set to the same number of data knots for each class. As the number of data knots and $k$-means centroids are much smaller than the number of original points, we set $k = 1$ in the classifiers. The other training and testing settings were those of the previous section.

The results are summarized in Table 2. The number ratio between the data knots and original points, called the knotting ratio, confirms the that the original data points were condensed into far fewer points after the knotting. Training the classifier on few data knots achieves the comparable accuracy of the classification with the $k$-means centroids, supporting our argument that the data knots well represent the original data. Note that the data knots are designed to provide lower-degree vanishing polynomials while $k$-means clustering is for simply summarizing nearby points. Compared to the classification results with original data points, those with data knots were comparable for the datasets with fewer classes (Iris and Wine; 3 classes). However, the accuracy degrades for the datasets with more classes (four classes in Vehicle and eleven classes in Vowel). In these datasets, there can be more overlapped points across different classes, resulting in accuracy degradation in ours and $k$-means. A possible solution specially tailored for classification is to introduce class-discriminative information to data knots. This modification is an interesting future extension.

## 6   Conclusion and Future work

The present paper focused on the tradeoff between noise-tolerance and better preserving an algebraic structure at the vanishing ideal construction, which has not been explicitly

considered before. We addressed a new problem to discover a set of vanishing polynomials that approximately vanish on the noisy input data and almost exactly vanish on the jointly discovered representative data points (called data knots). In the proposed framework, we introduced a vanishing polynomial construction method that takes into account two different point sets with different noise-tolerance scales. We also linked the newly introduced nonlinear regularization term to the Mahalanobis distance, which is commonly used in metric learning. In experiments, the proposed method discovered much more compact and lower-degree algebraic systems than the existing method.

Computing the vanishing polynomials that exactly vanish on the data knots remains an open problem. Exactly vanishing polynomials are desired because they can be manipulated by algebraic operations and combined with other algebraic tools. For practical reasons (numerical implementation and optimization), our method returns exactly vanishing polynomials in an extreme case only ($\delta = 0$), which often show poor performance in reality due to the numerical instability. Another future work is to increase computational efficiency. The proposed method is rather slower than VCA in training runtime (see the supplemental material) due to the optimization step for data knotting. Also, our method has three main hyperparameters to be tuned ($\varepsilon$, $\delta$, and $\lambda$), which requires additional cost for the cross validation step. Empirically, the most important hyperparameter is $\varepsilon$ that defines the error tolerance for the original data. Determining $\varepsilon$ is similar to determining the number of principal components in Principal Component Analysis (PCA), while $\varepsilon$ affects not only selecting linear polynomials but also selecting nonlinear polynomials. How to choose $\varepsilon$ is still an open problem for many vanishing ideal based approaches including ours.

## Acknowledgement

## References

Abbott, J.; Fassino, C.; and Torrente, M.-L. 2007. Thinning out redundant empirical data. *Mathematics in Computer Science* 1(2):375–392.

Bellet, A.; Habrard, A.; and Sebban, M. 2013. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*.

Cortes, C., and Vapnik, V. 1995. Support-vector networks. *Machine learning* 20(3):273–297.

Cox, D.; Little, J.; and O'shea, D. 1992. *Ideals, varieties, and algorithms*, volume 3. Springer.

Fassino, C. 2010. Almost vanishing polynomials for sets of limited precision points. *Journal of Symbolic Computation* 45(1):19–37.

Gao, G., and Wang, C. 2016. Nonlinear discriminant analysis based on vanishing component analysis. *Neurocomputing* 218(C):172–184.

Heldt, D.; Kreuzer, M.; Pokutta, S.; and Poulisse, H. 2009. Approximate computation of zero-dimensional polynomial ideals. *Journal of Symbolic Computation* 44(11):1566–1591.

Hou, C.; Nie, F.; and Tao, D. 2016. Discriminative vanishing component analysis. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 1666–1672. AAAI Press.

Kera, H., and Hasegawa, Y. 2016. Noise-tolerant algebraic method for reconstruction of nonlinear dynamical systems. *Nonlinear Dynamics* 85(1):675–692.

Kera, H., and Iba, H. 2016. Vanishing ideal genetic programming. In *Proceedings of the 2016 IEEE Congress on Evolutionary Computation*, 5018–5025. IEEE.

Laubenbacher, R., and Stigler, B. 2004. A computational algebra approach to the reverse engineering of gene regulatory networks. *Journal of Theoretical Biology* 229(4):523–537.

Laubenbacher, R., and Sturmfels, B. 2009. Computer algebra in systems biology. *American Mathematical Monthly* 116(10):882–891.

Li, F.; Li, Z.; Saunders, D.; and Yu, J. 2011. A theory of coprime blurred pairs. In *Proceedings of the 2011 International Conference on Computer Vision*, 217–224. Washington, DC, USA: IEEE.

Lichman, M. 2013. UCI machine learning repository.

Limbeck, J. 2013. *Computation of approximate border bases and applications*. Ph.D. Dissertation, Passau, Universität Passau.

Livni, R.; Lehavi, D.; Schein, S.; Nachliely, H.; Shalev-Shwartz, S.; and Globerson, A. 2013. Vanishing component analysis. In *Proceedings of The Thirteenth International Conference on Machine Learning*, 597–605.

Robbiano, L., and Abbott, J. 2010. *Approximate Commutative Algebra*, volume 1. Springer.

Torrente, M.-L. 2008. *Application of algebra in the oil industry*. Ph.D. Dissertation, Scuola Normale Superiore, Pisa.

Vera-Licona, P.; Jarrah, A.; Garcia-Puente, L. D.; McGee, J.; and Laubenbacher, R. 2014. An algebra-based method for inferring gene regulatory networks. *BMC systems biology* 8(1):37–52.