Electronic version (post-print) of an article published as
International Journal of Pattern Recognition and Artificial Intelligence
Vol. 27, No. 2 (2013) 1350004 (24 pages)

DOI: 10.1142/S0218001413500043

Modeling by fitting a union of polynomial functions to data in an errors-in-variables context

LEVENTE HUNYADI and ISTVÁN VAJK

Budapest University of Technology and Economics
Department of Automation and Applied Informatics
1117 Budapest, Magyar tudósok krt. 2. (building Q)

We present a model construction method based on a local fitting of polynomial functions to noisy data and building the entire model as a union of regions explained by such polynomial functions. Local fitting is shown to reduce to solving a polynomial eigenvalue problem where the matrix coefficients are data covariance and approximated noise covariance matrices that capture distortion effects by noise. By defining the asymmetric distance between two points as the projection of one onto the function fitted to the neighborhood of the other, we use a best weighted cut method to find a proper partitioning of the entire set of data into feasible regions. Finally, the partitions are refined using a modified version of a k-planes algorithm.

Keywords: pattern recognition, polynomial fitting, covariance matching, spectral clustering

1. Introduction

Many scientific fields, including computer vision, pattern recognition, data mining or system identification, involve problems where data is to be modeled by multiple subspaces. Even though the amount or dimensionality of the data may be large, the subspaces typically have few dimensions and admit a simple structure. Several algorithms exist that can tackle the so-called (multi-)subspace clustering or hybrid linear modeling problem, where the objective is to build a model where each partition of the data is captured by a linear relationship and the entire model is a composition of these partitions. Not every data set, however, admits a decomposition into linear relationships, and applying linear methods to essentially nonlinear relationships possibly by means of some smoothing approach loses the simplicity and explanation power of these methods. A natural generalization of subspace clustering is (multi-)manifold clustering, where each curved manifold is captured by some nonlinear relationship.

We discuss an approach to the clustering problem where each partition is characterized not by a line or plane but a polynomial, or more restrictively, a quadratic curve or surface, formulated in their implicit form $f(\mathbf{x}, \boldsymbol{\theta}) = 0$, or in more general,

not by a subspace but by a curved manifold. This involves a nonlinear estimation problem for each cluster. A simple but computationally efficient approach to deal with this estimation problem is to transform it into a linear setting. In two dimensions, for instance, the three components x, y and the constant term, is transformed into 6-component version with x^2, xy, y^2, x, y and the constant term, allowing us to express quadratic curves with a linear relationship $\boldsymbol{\theta}^\top \mathbf{f}(\mathbf{x}) = 0$ where $\boldsymbol{\theta}$ encapsulates the coefficients of x^2, xy, y^2, x, y and the constant term, respectively. Unfortunately, such transformation distorts any noise that may be present in x or y in a nonlinear manner, which must be duly accounted for, or simple algebraic estimation methods might produce suboptimal results.

Once an effective estimation method is in our hands that can fit simple polynomial functions to data without incurring too much computational overhead, we may obtain parameter estimates for any locality where points are related by the same function. Based on the assumption that a neighborhood that is sufficiently large to give reliable estimates but sufficiently small that all points in the neighborhood are similar in terms of how they are related, we define an asymmetric distance measure between two points as the distance from one point to its projection onto the manifold estimated from the neighborhood of the other point. The asymmetric distance measure produces an asymmetric distance matrix of the pairwise asymmetric distance between any two points.

Spectral clustering methods can find decomposition of data using spectral properties of the distance matrix. Given the matrix is not symmetric as typically expected, we use a modified spectral method that can exploit the asymmetry in the problem. The method splits the entire data set into k clusters, where each cluster is (ideally) characterized by the same implicit polynomial relationship. The output may then be refined with iterative approaches in the spirit of k-planes (k-flats), where estimation and assignment steps alternate until an acceptable decomposition of the data set is found.

The proposed method is robust, capable of handling intersections and data near boundaries, and adapts to more substantial levels of Gaussian noise.

The outline of the rest of the paper is as follows. Section 2 surveys related work, both fitting nonlinear functions to (a single cluster of) data as well as multi-subspace clustering of data, two fields that our work can be seen as a combination of. Section 3 discusses our contributions, elaborating on each step of our algorithm. Section 4 demonstrates how our algorithm works with some simple but illuminating examples and Section 5 concludes the paper.

2. Related work

Estimating parameters of a linear system where all data points are related by the same function but polluted by Gaussian noise with a known structure is the well-understood and widely-used method of total least squares fitting [14], solved as either an eigenvalue problem or a computationally more robust singular value

problem, which yields maximum likelihood estimates. Here, we seek to maximize the probability

$$p(\mathbf{x}_i \mid \boldsymbol{\theta},\, \mathbf{x}_{0,i}) = \frac{1}{\sqrt{(2\pi)^{\dim \mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}}} \det \mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}}}} \exp\left(-\frac{1}{2}\left(\mathbf{x}_i - \mathbf{x}_{0,i}\right)^\top \mathbf{C}^{-1}_{\boldsymbol{\sigma}^2_{\mathbf{x}}}\left(\mathbf{x}_i - \mathbf{x}_{0,i}\right)\right)$$

over the parameter vector $\boldsymbol{\theta}$ and all (unknown) data points $\mathbf{x}_{0,i}$, with (observed) data points $\mathbf{x}_i$ and noise covariance matrix $\mathbf{C}_{\boldsymbol{\sigma}^2_{\mathbf{x}}}$ (available up to scale) at our disposal. While tackled relatively easily for the linear case, the nonlinear case poses difficulty with the exploding number of unknowns. Generalizations of this single-cluster linear problem to the nonlinear case exist, both iterative and non-iterative algorithms, which may no longer achieve optimality but nevertheless can deliver accurate and consistent estimates.

Among fitting algorithms, methods based on maximum likelihood (ML) are regarded as the most accurate, and they attain the theoretical accuracy bound called the Kanatani–Cramer–Rao lower bound up to high-order error terms [18]. The key to approximated maximum likelihood estimation (AML) [7], one of the various computational schemes that have been proposed, is to formulate the probability function in an alternative way and substitute functions of unobservable noise-free variables with their approximations obtained from available noisy components. While the reformulated objective function depends on the unknown quantities $\mathbf{x}_{0,i}$, once substituted with $\mathbf{x}_i$, we get an iterative scheme that can produce parameter estimates in a few iterations.

Non-iterative schemes, even if not strictly optimal [18], can still deliver accurate estimates, and avoid any possibility of divergence inherent to iterative algorithms, especially in the presence of high levels of noise. The nonlinear extension to the Koopmans method [29] or the consistent algebraic least squares estimator [20,22], in contrast, are more reminiscent of the total least squares scheme, and try to match the data covariance matrix with the theoretical noise covariance matrix. Like with AML, the noise covariance matrix depends on $\mathbf{x}_{0,i}$, which are unknown, but can be approximated with $\mathbf{x}_i$, which are available. The nonlinear Koopmans method, for instance, reduces to a polynomial eigenvalue problem once the noise covariance matrix has been approximated, whereas a regular eigenvalue problem lies at the heart of total least squares.

On the other hand, many approaches exist that approximate a data set with a spline-like approach or some other weighting scheme where the model arises as a set of basis functions and associated weights. The approach may be geometrically motivated whereby approximation is interpreted as a continuous evolution process that drives an initial surface towards the target specified by the data points [2], or be a moving least squares variant where each data point is associated with a support region and has a local surface estimate, and these local estimates are blended together with a weighting matrix [13]. Other methods use a decomposition scheme to split the entire domain to suitably small domains that can be fit with a simple function [25] or merged to form larger clusters based on the inter-cluster

distance and intra-cluster distance criteria [32]. These methods, however, typically do not aid in understanding the data by finding a natural decomposition that reflects the internal model from which data originate.

In contrast, partitioning the entire data set into clusters where points within the cluster are related in the same way whereas different clusters are captured by different relationships can identify internal structure. Beyond the simple case when the cluster is represented by a single point, such as k-means, work on partitioning data has been focusing on subspace clustering or hybrid linear modeling algorithms where clusters are assumed to be related in a linear manner. These approaches include k-flats (KF) and its variants [3,1], generalized principal component analysis (GPCA) [31], local subspace affinity (LSA), random sample consensus (RANSAC) for hybrid linear modeling, agglomerative lossy compression (ALC) [21], spectral curvature clustering (SCC) [5], sparse subspace clustering (SSC) [10] and (spectral) local best-fit flats (S/LBF) [33]. An overview of such algorithms is given in [30].

Extensions of hybrid linear modeling algorithms exist that are not limited to clusters with intra-cluster data related in a linear manner. Nonlinear manifold clustering methods include locally linear manifold clustering (LLMC) [12], manifold clustering with node-weighted multidimensional scaling [27], kernel spectral curvature clustering (KSCC) [4] and mixture of probabilistic principal surfaces (MiPPS) [19]. Many of these are based on minimizing a cost function involving an affine combination of all points with different weights, or expectation maximization with general basis or kernel functions, shifting focus towards discovering the clusters rather than understanding the structure of the clusters themselves. A common characteristic is that the cluster definition is implicit (captured by some arrangement of data, e.g. closeness to a cluster center) rather than explicit (present in the data itself, e.g. explained by the equation of an ellipse).

Our interest has been motivated by an effective method [33], which uses spectral clustering for partitioning data into multiple linear subspaces. Extending its results to the nonlinear case, our goal has been to alloy estimation methods for the single-cluster scenario with clustering schemes used in the multi-cluster but linear setting. Previous work of ours [15,16,17] has focused on discovering polynomial patterns in data via a generalization of the k-planes approach, which could easily get stuck in local minima as it depended on the choice of an arbitrary initial state chosen without much prior information. This work provides a more robust approach with the initial state chosen using spectral clustering, and is a multi-cluster curved manifold clustering method of its own right.

3. Modeling with polynomial fitting

The main idea of our contribution is replacing linear subspace fitting with fitting curved manifolds, described by polynomial functions. Conceptually, our method is a spectral clustering algorithm with a specialized distance matrix measuring the distance of a point and its projection onto a curved manifold, which has been

(1) for each data point $\mathbf{x}_i$
  (a) start with an initial neighborhood $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$
  (b) estimate the parameters $\boldsymbol{\theta}_0$ that best capture points in $\mathcal{N}(\mathbf{x}_i)$
  (c) enlarge $\mathcal{N}(\mathbf{x}_i)$ by adding nearest-neighbor points
  (d) compute new estimates $\boldsymbol{\theta}_k$ and compare them to the estimates $\boldsymbol{\theta}_{k-1}$ obtained in the previous iteration
  (e) repeat until the neighborhood $\mathcal{N}(\mathbf{x}_i)$ cannot be enlarged without worsening the accuracy of $\boldsymbol{\theta}_k$
  (f) let $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_i)}$ be the best $\boldsymbol{\theta}_k$ that belongs to the optimum $\mathcal{N}(\mathbf{x}_i)$ around $\mathbf{x}_i$
(2) for each pair of data points $\mathbf{x}_i$ and $\mathbf{x}_j$, obtain their asymmetric distances
  (a) project $\mathbf{x}_i$ to the manifold parametrized by $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}$ and calculate a_{ij}
  (b) project $\mathbf{x}_j$ to the manifold parametrized by $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_i)}$ and calculate a_{ji}
(3) perform spectral clustering
  (a) build a matrix of asymmetric distances
  (b) assign data points to groups based on the distance matrix eigenvectors

Fig. 1: The pseudo-code of the proposed algorithm.

estimated from a local neighborhood. As such, the inputs of the algorithm are as follows:

$\mathbf{X}$	data set, n rows, N columns
n	number of dimensions (2 for plane, 3 for space)
N	number of data points
$\boldsymbol{\Sigma}$	the $n \times n$ noise covariance matrix
k	the desired number of clusters

Each column of the data set matrix $\mathbf{X}$ is a data point $\mathbf{x}_i^\top = \begin{bmatrix} x_i & y_i \end{bmatrix}$ for 2 dimensions or $\mathbf{x}_i^\top = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}$ for 3 dimensions, and the covariance matrix $\boldsymbol{\Sigma} = \mathbb{E}\left(\tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^\top\right)$ where the data point $\mathbf{x}_i$ is split into a noise-free part and a noise part as $\mathbf{x}_i = \mathbf{x}_{i,0} + \tilde{\mathbf{x}}_i$ where neither can be observed directly. As output, the algorithm produces a mapping (a membership set) $\mathcal{I}$ between data points, and one of the k clusters. Data points $\mathbf{x}_i$ in the same cluster are captured by the same polynomial function. Thus, the method constructs a model as a union of polynomial functions fitted to data. The pseudo-code of the algorithm is shown in Figure 1.

The algorithm can be broken down into the following sub-problems:

(1) Fitting polynomial surfaces to data. This is accomplished with the nonlinear Koopmans (NK) method [29], which can be seen as an alternative formulation of consistent algebraic least squares [20,22]. The method solves a polynomial eigenvalue problem, the matrix coefficients being the data covariance matrix and noise cancellation terms. The data covariance matrix is computed from the

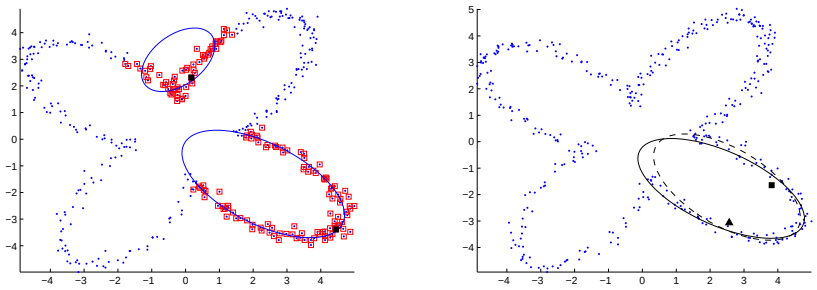
data $\mathbf{x}_i$ subject to the non-linear transformation $\mathbf{f}(\mathbf{x}_i)$, and the noise cancellation terms mitigate the effect of this transformation on noise. The approach is not optimal in a maximum likelihood sense but nevertheless consistent: more data produces better estimates.

(2) Identifying local neighborhoods. Any estimation method requires sufficient data to produce reliable parameter estimates, especially in the presence of noise. In most cases, the local neighborhood of a data point consists of other points that are related in a similar way, or in other words, captured by the same polynomial function. This is implicit in many multi-manifold clustering algorithms but is made explicit in our approach: points in a local neighborhood determine a fitting, and the fitting is deemed to apply in that neighborhood. Choosing the right neighborhood can be accomplished with an inflationary algorithm involving k_{nn} nearest neighbors, where k_{nn} is an increasingly larger number until a suitable error measure begins to increase rather than decline.

(3) Projecting a point onto a curved manifold. While projecting to a subspace can be easily accomplished, projecting to a curved manifold captured by a polynomial function is computationally more intensive. Efficient methods that compute the so-called *foot point* of a data point that does not lie on the curved manifold are essential to the algorithm. Special cases for projecting to ellipses and ellipsoids, and quadratic curves and surfaces will be discussed. Projection onto a manifold lets us determine the distance between a point and its foot point where the parameters of the curved manifold on which the foot point lies have been estimated from a local neighborhood.

(4) Spectral clustering using an asymmetric distance matrix. A clustering method by weighted cuts in directed graphs will be employed to find a partitioning of the entire data set into disjoint clusters, exploiting the asymmetry in the distance matrix. The method is based on a generalization of the original spectral approach involving symmetric matrices. The asymmetric distance matrix stems from curved manifolds estimated around the local neighborhood of points, and projecting all (typically other) points onto this manifold, and measuring the distance between point and its foot point. Spectral clustering reveals points that naturally group together, i.e. groups of points that are captured by the same polynomial relationship, which is what we ultimately seek.

Figure 2 illustrates some of the sub-problems that comprise the steps of the algorithm. Sub-figure 2a shows how the estimation algorithm fits an ellipse to a noisy set of data and how the local neighborhood is enlarged until the greatest possible number of data points is used to estimate the parameters of an ellipse; observe how the neighborhood is no longer extended once the fitting error begins to increase. The filled square marker indicates the point from which neighborhood detection starts, empty square markers around dots indicate neighborhood membership of data points. Sub-figure 2b shows how the asymmetric distance, which is the basis for spectral clustering, acts as a measure of point proximity, i.e. whether the points



(a) local neighborhood detection (b) proximity based on data projection

Fig. 2: Sub-problems that comprise our proposed algorithm.

belong to the same group. The filled triangle and the filled square marker indicate points that are close even though their Euclidean distance is relatively large. The continuous line shows the ellipse estimated from the points that form the neighborhood of the data point with the filled triangle marker, whereas the dashed line depicts the ellipse that belongs to the neighborhood of the point with the filled rectangle marker. Both the triangle marker point projected to the dashed line ellipse and the square marker point to the the continuous line ellipse yield foot points that are relatively near to their originals, producing a small asymmetric distance measure.

3.1. *Parameter estimation*

For the purposes of parameter estimation, we assume that the data has zero mean and has been normalized to its root mean square (RMS), which is meant to reduce numerical errors. (The estimation algorithm, as presented here, is not invariant to translation and scaling.) For each dimension x of the data set $\mathbf{X}$, this means

$$\begin{aligned} m_x &= \frac{1}{N}\sum_{i=1}^{N} x_i \\ x_i &\leftarrow \frac{x_i - m_x}{s} \\ \sigma_x &\leftarrow \frac{\sigma_x}{s} \end{aligned}$$

and likewise for all other dimensions y, z, etc. where

$$s = \sqrt{\frac{1}{2n}\sum_{i=1}^{n}\left\{(x_i - m_x)^2 + (y_i - m_y)^2 + \ldots\right\}}$$

Reducing data spread also reduces the additive noise on components of $\mathbf{x}_i$ and the noise covariance matrix $\mathbf{\Sigma}$ is updated accordingly. Thereafter, we assume data is

polluted by Gaussian noise with covariance $\mathbf{\Sigma}$.

The objective of the parameter estimation step is to find parameters $\mathbf{g}$ such that

$$f(\mathbf{x}_{i,0}, \mathbf{g}) = 0 \tag{1}$$

where $\mathbf{x}_{i,0}$ is a vector of noise-free components that can be observed only via their noisy counterparts $\mathbf{x}_i = \mathbf{x}_{i,0} + \tilde{\mathbf{x}}_i$ with $i = 1, \ldots, N$ with N being the number of observations, $\tilde{\mathbf{x}}_i$ is a noise contribution with $\tilde{\mathbf{x}}_i \sim N(\mathbf{0}, \mathbf{\Sigma})$ where $\mathbf{\Sigma} = \operatorname{diag}\left(\boldsymbol{\sigma}_{\mathbf{x}}^2\right)$ and $\boldsymbol{\sigma}_{\mathbf{x}}^2$ is a vector of variances for each component of an $\tilde{\mathbf{x}}_i$. $\boldsymbol{\sigma}_{\mathbf{x}}^2$ is assumed to be known up to scale (multiplication by a constant). The noise covariance matrix $\mathbf{\Sigma}$ is (in general) of full rank: there is no distinguished variable that we can observe noise-free, usually called an errors-in-variables approach in statistics. Furthermore, we rewrite (1) in a form

$$\mathbf{f}^\top(\mathbf{x}_{0,i})\boldsymbol{\theta} = 0 \tag{2}$$

where the function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ is a (so-called) carrier function that maps the data vector $\mathbf{x}_{0,i} \in \mathbb{R}^n$ into m-dimensional space. In other words, once we substitute $\mathbf{z}_{0,i} = \mathbf{f}(\mathbf{x}_{0,i})$, the relationship simplifies into a (pseudo-)linear relationship

$$\mathbf{z}_{0,i}^\top \boldsymbol{\theta} = 0$$

that splits the expression into data and parameters. In particular, quadratic curves can be captured with

$$\mathbf{z}_i^\top = \begin{bmatrix} x_i^2 & x_i y_i & y_i^2 & x_i & y_i & 1 \end{bmatrix}$$

and quadratic surfaces can be captured with

$$\mathbf{z}_i^\top = \begin{bmatrix} x_i^2 & y_i^2 & z_i^2 & y_i z_i & x_i z_i & x_i y_i & x_i & y_i & z_i & 1 \end{bmatrix}$$

where x_i, y_i and z_i are data point coordinates in two or three dimensions.

The estimation method resembles total least squares [14] in that it matches the data covariance matrix with the conceptual noise covariance matrix. The underlying assumption is that were it not for the noise present in observations $\mathbf{x}_i$, the data covariance matrix would be a singular matrix due to (2). Thus, finding the eigenvector of the data covariance matrix with respect to a noise covariance matrix with the smallest-magnitude eigenvalue, parameter estimates can be found [29,20,22].

Arranging $\mathbf{z}_i = \mathbf{f}(\mathbf{x}_i)$ in a (mean-free) matrix $\mathbf{Z}$ where each row is a data point, and setting $\mathbf{D} = \frac{1}{N}\mathbf{Z}^\top\mathbf{Z}$, we have a data covariance matrix $\mathbf{D}$ of noise-contaminated observations. Next, we find the smallest-magnitude (positive real) eigenvalue μ in the polynomial eigenvalue problem (PEP)

$$\mathbf{\Psi}(\mu)\boldsymbol{\theta} = (\mathbf{D} - \mathbf{C}(\mu))\,\boldsymbol{\theta} = \mathbf{0}$$

in which $\mathbf{C}(\mu)$ is a polynomial in the scalar μ, meant to cancel noise effects, with matrix polynomial coefficients to be derived below. As the matrix $\mathbf{D}$ is of full rank,

the null space of $\mathbf{D} - \mathbf{C}(\mu)$ yields the parameter vector $\boldsymbol{\theta}$ (up to scale, which we normalize to $\|\boldsymbol{\theta}\| = 1$). $\mu \geqq 0$ expresses a physical constraint that noise magnitude cannot be negative.

The key point is to express the covariance matrix $\mathbf{C}$. The original noise covariance matrix $\boldsymbol{\Sigma}$ is no longer valid in the transformed space, hence data points and the carrier function must be used to estimate $\mathbf{C}$. If $\mathbf{f}$ is a polynomial function, $\mathbf{C} = \mathbf{C}(\mathbf{x}, \boldsymbol{\sigma}_{\mathbf{x}}^2)$ can be approximated in a series of simple steps, and written as a matrix polynomial in μ if $\boldsymbol{\sigma}_{\mathbf{x}}^2$ is expressed as $\boldsymbol{\sigma}_{\mathbf{x}}^2 = \mu \bar{\boldsymbol{\sigma}}_{\mathbf{x}}^2$ (with μ being "magnitude" and $\bar{\boldsymbol{\sigma}}_{\mathbf{x}}^2$ "direction") where $\left\|\bar{\boldsymbol{\sigma}}_{\mathbf{x}}^2\right\| = 1$. Let w_i be an arbitrary component in $\mathbf{z}_i^\top$ and its noise contribution $\tilde{w}_i$ be normally distributed with variance σ_w^2. The relationships to use are as follows:

- $\mathbb{E}\left(w_i^p\right) = \mathbb{E}\left\{(w_{0,i} + \tilde{w}_i)^p\right\}$: an observed quantity is split into an unobserved quantity and noise.
- $\mathbb{E}\left(\tilde{w}_i^{2p}\right) = \sigma_w^{2p}(2p-1)!!$: even central moments of the normally distributed variable $\tilde{w}$ (where !! stands for double factorial).
- $\mathbb{E}\left(\tilde{w}_i^{2p-1}\right) = 0$: odd central moments of the normally distributed variable $\tilde{w}$.
- $\mathbb{E}\left(w_i\right) \approx \bar{w} = \frac{1}{N}\sum_{i=1}^{N} w_i$: expected value is approximated with mean.

For example, $\mathbf{C}$ for estimating parameters of a quadratic curve will take the form $\mathbf{C}(\mu) = \mu^2\mathbf{C}_2 + \mu\mathbf{C}_1$ where $\mathbf{C}_2$ is a matrix whose entries depend on σ_x^2 and σ_y^2, while those of $\mathbf{C}_1$ depend on σ_x^2 and σ_y^2 as well as $\mathbb{E}\left(x^2\right)$, $\mathbb{E}\left(y^2\right)$, $\mathbb{E}\left(xy\right)$, $\mathbb{E}\left(x\right)$ and $\mathbb{E}\left(y\right)$, all of which are approximated from finite samples.

Let us, for instance, consider expressing the top left entry of the matrix polynomial $\mathbf{C}(\mu) = \mathbb{E}\left(\tilde{\mathbf{z}}_i\tilde{\mathbf{z}}_i^\top\right)$ with data linearized as

$$\mathbf{z}^\top = \begin{bmatrix} x^2 & xy & y^2 & x & y & 1 \end{bmatrix}$$

where we have dropped the index i in x_i and y_i for brevity. Here, we seek $\mathbb{E}\left\{\left(x^2\right)^2\right\}$, for which

$$\begin{aligned}\mathbb{E}\left\{\left(x^2\right)^2\right\} &= \mathbb{E}\left(x^4\right) = \mathbb{E}\left(x_0 + \tilde{x}\right)^4 \\ &= \mathbb{E}\left(x_0^4 + 4x_0^3\tilde{x} + 6x_0^2\tilde{x}^2 + 4x_0\tilde{x}^3 + \tilde{x}^4\right)\end{aligned}$$

where $\mathbb{E}\left(4x_0^3\tilde{x}\right) = 4x_0^3\mathbb{E}\left(\tilde{x}\right) = 0$ and $\mathbb{E}\left(4x_0\tilde{x}^3\right) = 4x_0\mathbb{E}\left(\tilde{x}^3\right) = 0$ (expected value of odd central moment equals zero), yielding

$$\mathbb{E}\left(x^4\right) = \mathbb{E}\left(x_0^4 + 6x_0^2\tilde{x}^2 + \tilde{x}^4\right)$$

where $\mathbb{E}\left(\tilde{x}^2\right) = \sigma_x^2$ and $\mathbb{E}\left(\tilde{x}^4\right) = 3\sigma_x^4$ (expected value of even central moment) such that

$$\mathbb{E}\left(x^4\right) = x_0^4 + 6x_0^2\sigma_x^2 + 3\sigma_x^4.$$

This gives

$$\begin{aligned}\mathbb{E}\left(x^4\right) - x_0^4 &= 6x_0^2\sigma_x^2 + 3\sigma_x^4 \\ &= 6\sigma_x^2\left(\mathbb{E}\left(x^2\right) - \sigma_x^2\right) + 3\sigma_x^4 \\ &= 6\sigma_x^2\mathbb{E}\left(x^2\right) - 3\sigma_x^4.\end{aligned}$$

where $\mathbb{E}\left(x^2\right) = \mathbb{E}\left(x_0^2 + \tilde{x}^2\right) = x_0^2 + \mathbb{E}\left(\tilde{x}^2\right) = x_0^2 + \sigma_x^2$ and thus $x_0^2 = \mathbb{E}\left(x^2\right) - \sigma_x^2$. Notice how the expression finally reduces to components that depend on either the original noise covariance matrix $\mathbf{\Sigma}$ such as σ_x^2 and σ_x^4, and components that can be approximated from observed samples $\mathbb{E}\left(x^2\right) \approx \frac{1}{N}\sum_i x_i^2$.

Iteratively applying the substitution technique above, we arrive at a special case of a PEP, called a quadratic eigenvalue problem (QEP)

$$\mathbf{\Psi}(\mu)\boldsymbol{\theta} = \left(\mathbf{D} - \mu\mathbf{C}_1 - \mu^2\mathbf{C}_2\right)\boldsymbol{\theta}$$

for estimating quadratic curves and surfaces, which involves a $\mathbf{C}(\mu)$ with at most a quadratic dependence on μ.

One way to solve the QEP

$$\mathbf{\Psi}(\mu)\boldsymbol{\theta} = \left(\mathbf{D} - \mathbf{C}(\mu)\right)\boldsymbol{\theta} = \mathbf{0}$$

is to apply linearization, thereby eliminating the polynomial dependence on μ at the expense of increasing the size of coefficient matrices, which is analogous to companion matrices constructed from polynomials where the eigenvector of the companion matrix yields the roots of the polynomial. In particular, transformations that preserve symmetry are especially favored for their numerical stability.

A well-known result [28] for linearizing the QEP

$$\mu^2\mathbf{R}_2 + \mu\mathbf{R}_1 + \mathbf{R}_0$$

is with the first companion form

$$\mathbf{\Xi}(\lambda) = \mathbf{\Xi}_1 - \lambda\mathbf{\Xi}_2 = \begin{bmatrix} \mathbf{0} & \mathbf{W} \\ -\mathbf{R}_0 & -\mathbf{R}_1 \end{bmatrix} - \lambda\begin{bmatrix} \mathbf{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_2 \end{bmatrix}$$

where the choice for the (arbitrary full-rank) $\mathbf{W} = -\mathbf{R}_0$ yields a generalized eigenvalue problem with symmetric matrices where the eigenvector $\mathbf{w}$ has a special structure

$$\mathbf{w} = \begin{bmatrix} \mathbf{v} \\ \lambda\mathbf{v} \end{bmatrix}.$$

In our case of quadratic curves and surfaces, the substitutions are

$$\begin{aligned}\mathbf{R}_0 &= \mathbf{D} \\ \mathbf{R}_1 &= \mathbf{C}_1 \\ \mathbf{R}_2 &= \mathbf{C}_2\end{aligned}$$

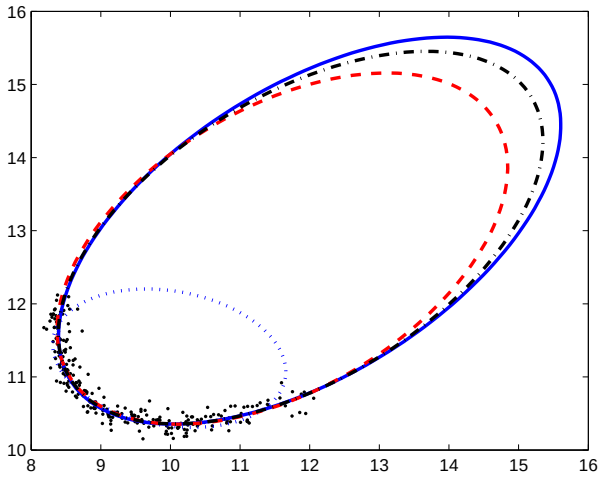


Fig. 3: Comparing the accuracy of direct ellipse fit (dotted line), hyperaccurate ellipse fit (dashed line) and the estimation method used in this paper (dash dot line). Both hyperaccurate ellipse fit and the method used in this paper are equally close to the original curve (continuous line) from which data has been sampled whereas direct ellipse fit exhibits a significant low-eccentricity bias. Data points (shown as dots) are contaminated with noise.

Solving the resulting generalized eigenvalue problem is the linearized equivalent of solving the original polynomial eigenvalue problem. As the linearized problem has eigenvectors $\mathbf{w}$ of dimension mp rather than m, the true polynomial eigenvector that belongs to the eigenvalue μ becomes the column $\mathbf{v}$ of $\mathrm{vec}\mathbf{V} = \mathbf{w}$ of the linearized eigenvector $\boldsymbol{\Xi}_1\mathbf{w} = \lambda\boldsymbol{\Xi}_2\mathbf{w}$ that gives the smallest normalized residual, i.e.

$$\mathbf{v} = \arg\min_{\mathbf{v}} \frac{\sum_k |[\boldsymbol{\Psi}(\mu)\mathbf{v}]_k|}{\sum_k |[\mathbf{v}]_k|}$$

where $[\mathbf{v}]_k$ is the kth component of the vector $\mathbf{v}$.

Comparison with other estimation methods Given the simplicity of traditional least squares estimators and the accuracy of maximum likelihood-based estimators, the near-optimality of iterative approaches and the inherent bias present in non-iterative approaches, we give some arguments for our choice of the fitting algorithm.

First, the algorithm we use is non-iterative, and such fitting methods fail to achieve the Kanatani–Cramer–Rao lower bound, the theoretical bound for accuracy [18]. However, the estimation method we employ is used in an unsupervised classification setting where we must simultaneously optimize two different but in-

terdependent aspects: identification of groups and estimation accuracy within the group. Provable statistical bounds for optimal algorithms are applicable insofar that we assume perfect clustering, and as such, accuracy loses significance when data in a group may not in reality belong to the same group. Furthermore, iterations required for non-linear optimization may often make the algorithm not converge in the presence of large noise (or specifically, a large degree of misclassification). The robustness and reduction in complexity we gain is sizable and comes at a price of only slightly compromised accuracy.

Second, non-iterative algorithms exist that outperform the algorithm used in this paper in one respect or other, e.g. hyperaccurate quadratic curve fitting [18] completely cancels second-order error. The estimation algorithm we use, however, yields similar results but comes at a modest computational cost, i.e. amounts to solving a generalized eigenvalue problem, whereas e.g. hyperaccurate fitting requires the computation of several matrix pseudo-inverses in addition. Both algorithms far outperform traditional least squares, which has a substantial low-eccentricity bias.

Figure 3 compares three estimation methods: a least-squares-based approach called direct ellipse fit [11] (shown with dotted line), hyperaccurate ellipse fit [18] (shown with dashed line) and the estimation based on a noise compensation approach we use in this paper (shown with dash dot line). In Figure 3, 250 original data points sampled evenly along an ellipse (continuous line) are contaminated with Gaussian noise of $\sigma_{\mathbf{x}} = 0.1$ (plotted as black dots). The original ellipse that has generated the data points has center $(12, 13)$, semi-axes lengths of 4 and 2, and angle of tilt $\frac{\pi}{6}$. It can be seen how the estimator we use can achieve an accuracy strikingly similar to that of hyperaccurate ellipse fit, while substantially outperforming direct ellipse fit.

3.2. *Finding the optimal local neighborhood*

Parameter estimation can produce a curve or surface that captures all related points in a curved manifold if the proper set of input data is identified. Assuming that points are surrounded by other related data points, a finite neighborhood of any (or most) data points can produce a good estimate valid around that data point. Should the neighborhood grow too large, unrelated data points may fall into the neighborhood and the parameter estimates will be seriously distorted. The key of finding the optimal neighborhood is adding as many nearest neighbors to a data points as possible without negatively impacting estimation quality.

Let us introduce the following notation:

$\mathbf{x}_i$	the point whose neighborhood to determine
$\mathcal{N}(\mathbf{x}_i)$	the neighborhood set of $\mathbf{x}_i$ (in Euclidean sense)
$\mathcal{N}_k(\mathbf{x}_i)$	the neighborhood set of $\mathbf{x}_i$ comprising of k nearest neighbors
$\mathbf{X}_{n(\mathbf{x}_i)}$	a matrix of points in the neighborhood of $\mathbf{x}_i$

The input to the algorithm to find the optimal local neighborhood is as follows:

k initial number of k-nearest neighbors
s k-nearest neighbor increment

As output, the algorithm yields $\mathcal{N}(\mathbf{x}_i)$, with the largest number of points that still guarantees that the points are captured by the same polynomial relationship.

The optimal neighborhood for each data point is thus determined by finding the neighborhood that minimizes the error measure

$$\frac{\mu}{|\mathcal{N}_k(\mathbf{x}_i)|} \tag{3}$$

where μ solves the polynomial eigenvalue problem

$$\left(\mathbf{D} - \mu\mathbf{C}_1 - \mu^2\mathbf{C}_2\right)\boldsymbol{\theta}$$

which is the objective function of the estimator we use with

$$\mathbf{D} = \frac{1}{|\mathcal{N}(\mathbf{x}_i)|}\mathbf{X}_{\mathcal{N}(\mathbf{x}_i)}^{\top}\mathbf{X}_{\mathcal{N}(\mathbf{x}_i)}.$$

The numerator of (3) measures the goodness-of-fit assuming a neighborhood membership $\mathcal{N}_k(\mathbf{x}_i)$ and the denominator of (3) penalizes smaller sets and encourages the neighborhood to expand until it engulfs outliers or unrelated data points.

Minimizing (3) is most easily accomplished with an inflation algorithm: starting with an initial number of k nearest neighbors, s new nearest neighbors are added each turn and the polynomial eigenvalue problem is solved for μ. The size of the neighborhood will be determined by the optimal μ as compared to the number of points in the neighborhood.

As an outcome of neighborhood discovery, points that are "inside" a curve or surface section are assigned parameter estimates drawn from a large neighborhood, whereas points near the edges or intersections (where many neighboring points originate from another group) have less reliable parameter estimates but also a far smaller neighborhood. Given that estimates belonging to points with larger neighborhoods are more consistent with estimates derived from points in the vicinity (e.g. as is the case when points are on the same stretch of homogeneous surface), these neighborhoods will dominate other neighborhoods in the spectral clustering phase of the algorithm.

3.3. *Projection*

A key point in our clustering algorithm is defining the distance between any two points as the distance of one point $\mathbf{x}_i$ from its *foot point* $\mathbf{x}_{f,i}$ such that

$$\mathbf{x}_{f,i} = \arg\min_{\mathbf{x}} \|\mathbf{x} - \mathbf{x}_i\|^2 .$$

A foot point is thus the projection of a point to a surface. In previous sections, we have seen how to estimate parameters of a surface from a set of points and how to find the optimal number of neighboring points to get surface parameter estimates. However, finding the projection of a point to the surface has remained an open

issue. Depending on the surface, various projection schemes can be devised; we discuss the special cases of projecting to a linear or a quadratic surface.

In order to simplify discussion, we assume that noise is equally distributed across all dimensions of the data points $\mathbf{x}_i$ and thus the Euclidean distance is a valid distance measure. Otherwise, we may apply a scaling transformation involving the noise covariance matrix $\mathbf{\Sigma}$ such that points $\mathbf{x}_i \leftarrow \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{x}_i$ satisfy this assumption.

3.3.1. *Projection for linear data functions*

When linear estimation is used, the foot point is calculated as a simple point–line or point–plane distance. Let $\mathbf{n}$ be the normalized line or plane normal vector and c the constant in the Hessian normal form equation of the plane

$$L(\mathbf{x}_f) = \mathbf{n} \cdot \mathbf{x}_f + c = 0$$

such that $\boldsymbol{\theta}^\top = \begin{bmatrix} \mathbf{n}^\top & c \end{bmatrix}$ up to scaling by a constant. In such a case, the (signed) distance between the point and the line or plane is given by the projection

$$d = \mathbf{n} \cdot \mathbf{x}_i + c.$$

3.3.2. *Projection for quadratic data functions*

When quadratic estimation is used, obtaining the foot point is not as straightforward [8]. A quadratic curve in two dimensions or a quadric surface in three dimensions can be recast in the form

$$Q(\mathbf{x}_f) = \mathbf{x}_f^\top \mathbf{A}\mathbf{x}_f + \mathbf{b}^\top \mathbf{x}_f + c = 0 \tag{4}$$

where $\mathbf{A}$ is a symmetric matrix such that

$$\boldsymbol{\theta}^\top = \begin{bmatrix} \operatorname{symvec}(\mathbf{A})^\top & \mathbf{b}^\top & c \end{bmatrix}$$

where $\operatorname{symvec}(\mathbf{A})$ stacks the upper triangular part of the matrix $\mathbf{A}$ including the main diagonal into a column vector. Geometrically, the closest point $\mathbf{x}_f$ on the surface to $\mathbf{x}$ must satisfy the condition that $\mathbf{x} - \mathbf{x}_f$ is normal to the surface. Since the surface gradient $\nabla Q(\mathbf{x}_f)$ is normal to the surface, the algebraic condition for the closest point is

$$\mathbf{x} - \mathbf{x}_f = t\nabla Q(\mathbf{x}_f) = t(2\mathbf{A}\mathbf{x}_f + \mathbf{b})$$

for some scalar t. Therefore, $\mathbf{x}_f = (\mathbf{I} + 2t\mathbf{A})^{-1}(\mathbf{x} - t\mathbf{b})$ where $\mathbf{I}$ is the identity matrix.

Instead of immediately replacing $\mathbf{x}$ in the quadratic equation, factor $\mathbf{A}$ using an eigendecomposition to obtain $\mathbf{A} = \mathbf{R}\mathbf{D}\mathbf{R}^\top$ where $\mathbf{R}$ is an orthonormal matrix whose columns and $\mathbf{D}$ is a diagonal matrix whose diagonal entries are eigenvectors

and eigenvalues of $\mathbf{A}$, respectively. Then

$$\begin{aligned}\mathbf{x}_f &= (\mathbf{I} + 2t\mathbf{A})^{-1}(\mathbf{x} - t\mathbf{b}) \\ &= (\mathbf{R}\mathbf{R}^\top + 2t\mathbf{R}\mathbf{D}\mathbf{R}^\top)^{-1}(\mathbf{x} - t\mathbf{b}) \\ &= \left\{\mathbf{R}(\mathbf{I} + 2t\mathbf{D})\mathbf{R}^\top\right\}^{-1}(\mathbf{x} - t\mathbf{b}) \\ &= \mathbf{R}(\mathbf{I} + 2t\mathbf{D})^{-1}(\alpha - t\beta)\end{aligned}$$

where

$$\alpha = \mathbf{R}^\top\mathbf{x} \qquad \beta = \mathbf{R}^\top\mathbf{b}.$$

Re-substituting into the quadratic equation (4),

$$\begin{aligned}0 = (\alpha - t\beta)^\top(\mathbf{I} + 2t\mathbf{D})^{-1}D(\mathbf{I} + 2t\mathbf{D})^{-1}(\alpha - t\beta) \\ + \beta(\mathbf{I} + 2t\mathbf{D})^{-1}(\alpha - t\beta) + c\end{aligned}$$

which is an at most fourth-degree polynomial for two dimensions, and an at most sixth-degree polynomial for three dimensions in terms of the scalar variable t. Once the roots t_k are found, they can be substituted into

$$\mathbf{x}_{f,k} = (\mathbf{I} + 2t_k\mathbf{A})^{-1}(\mathbf{x} - t_k\mathbf{b})$$

where the $\mathbf{x}_{f,k}$ that produces the smallest distance yields the foot point we seek.

3.3.3. *Projection to an ellipse or ellipsoid*

Finding the foot points can be simplified further if the type of the quadratic curve or surface can be identified or constrained. As an important special case, consider assignment to an ellipse (or ellipsoid); algorithms for other quadratic curves can be derived in a similar manner [6]. Without loss of generality, we can assume the ellipse (or ellipsoid) is axis-aligned and centered at the origin. If not, a transformation matrix $\mathbf{M}$ that axis-aligns and centers the ellipse can be applied to the data points, and the inverse transformation matrix $\mathbf{M}^{-1}$ to the computed foot points. We discuss projection to ellipses but the approach generalizes to ellipsoids with the addition of the extra dimension. Thus,

$$Q(\mathbf{x}_f) = \frac{x_f^2}{a^2} + \frac{y_f^2}{b^2} - 1 = 0. \tag{5}$$

For the distance between a data point $\begin{bmatrix} x & y \end{bmatrix}$ and a foot point $\begin{bmatrix} x_f & y_f \end{bmatrix}$ to be minimum, the distance vector must be normal to the ellipse, which means that the ellipse gradient vector

$$\frac{1}{2}\partial_{\mathbf{x}}Q(\mathbf{x}_f) = \begin{bmatrix} \frac{x_f}{a^2} & \frac{y_f}{b^2} \end{bmatrix}$$

and the distance vector should be equal up to magnitude. This implies (after rearrangements) that

$$x_f = \frac{a^2 x}{t + a^2} \qquad y_f = \frac{b^2 y}{t + b^2}$$

where t is a scalar. Substituted into (5) yields

$$\begin{aligned} Q(t) &= \frac{1}{a^2}\left(\frac{a^2 x}{t+a^2}\right)^2 + \frac{1}{b^2}\left(\frac{b^2 y}{t+b^2}\right)^2 - 1 \\ &= \left(\frac{ax}{t+a^2}\right)^2 + \left(\frac{by}{t+b^2}\right)^2 - 1 = 0. \end{aligned}$$

Differentiating w.r.t. t we get

$$\begin{aligned} \frac{d}{dt}Q(t) &= -\frac{2a^2x^2}{(t+a^2)^3} - \frac{2b^2y^2}{(t+b^2)^3} \\ \frac{d}{dt^2}Q(t) &= \frac{6a^2x^2}{(t+a^2)^4} + \frac{6b^2y^2}{(t+b^2)^4} \end{aligned}$$

where $\frac{d}{dt}Q(t) < 0$ and $\frac{d}{dt^2}Q(t) > 0$ (strictly monotonic decreasing) in the domain of interest and

$$\begin{aligned} \lim_{t\to -b^2} Q(t) &= \infty \\ \lim_{t\to \infty} Q(t) &= -1 \end{aligned}$$

therefore a unique root t of $Q(t)$ must exist. One way to find this root is using Newton's method [9].

3.4. *Spectral clustering*

Identifying the optimal neighborhood around each point, and estimating parameters of the associated local model, we can project any data point to the estimated surface. Iterating over all data points yields an asymmetric distance matrix $\mathbf{A}_D$ whose entry a_{ij} represents the distance of data point $\mathbf{x}_i$ from its foot point obtained by projecting $\mathbf{x}_i$ onto the surface around $\mathbf{x}_j$, defined by parameters $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}$, i.e.

$$a_{ij} = d\left(\mathbf{x}_i,\ \mathbf{p}\left(\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)},\ \mathbf{x}_i\right)\right)$$

where d denotes Euclidean distance and $\mathbf{p}\left(\boldsymbol{\theta},\ \mathbf{x}_i\right)$ denotes the projection of the point $\mathbf{x}_i$ to the curve or surface defined by $\boldsymbol{\theta}$, and $\boldsymbol{\theta}_{\mathcal{N}(\mathbf{x}_j)}$ are the parameters of the curve or surface estimated from the set of points $\mathcal{N}(\mathbf{x}_j)$, which is the local neighborhood of $\mathbf{x}_j$.

One approach to identifying groups in a data set with a notion of distance matrix is spectral clustering [26,24]. Spectral clustering is initialized with a (symmetric) scatter matrix $\mathbf{S}_S$ (where the subscript S stands for symmetric) with entries s_{ij} between 0 and 1, not at all similar and most similar, respectively. A straightforward way to convert a distance matrix into a scatter matrix is via the exponential function with a negative exponent.

Once we have a scatter matrix $\mathbf{S}_S$, we may proceed as follows:

- Normalize the scatter matrix $\mathbf{S}_S$. Let

$$d_i = \frac{1}{\sqrt{\sum_j [\mathbf{S}_S]_{ij}}}$$

and $\mathbf{D}_S = \text{diag}\,(d_i)$ such that

$$\mathbf{H}_S = \mathbf{I} - \mathbf{D}_S \mathbf{S}_S \mathbf{D}_S$$

.

- Compute the smallest eigenvectors of the matrix $\mathbf{H}_S = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ where

$\mathbf{U}$	matrix of eigenvectors
$\mathbf{u}_i$	the ith smallest eigenvector
$\mathbf{U}_k$	the bottom k smallest eigenvectors $\mathbf{u}_1$, $\mathbf{u}_2$, ... arranged in a matrix
λ_i	the ith smallest eigenvalue
$\mathbf{\Lambda}_k$	a diagonal matrix of the bottom k smallest eigenvalues λ_1, λ_2, ...

- Perform a k-means clustering on $\mathbf{U}_k \mathbf{\Lambda}_k^{-\frac{1}{2}}$. The clusters are the result of the spectral clustering algorithm.

This approach, however, requires a symmetric scatter matrix to initialize, which may be obtained as

$$\mathbf{S}_S = \exp\left(-\alpha\sqrt{\mathbf{A}_D^\top \mathbf{A}_D}\right)$$

where the operators $\exp(\bullet)$ and $\sqrt{\bullet}$ are to be understood element-wise and $\alpha > 0$ is a scalar parameter.

Unfortunately, such an approach destroys any asymmetry present in the original problem and may produce largely suboptimal results. Asymmetric spectral clustering, an algorithm to find a best cut in a weighted graph [23], remedies the issue by procrastinating enforcing symmetry to a later phase of the algorithm. Let $\mathbf{S}_A$ be a(n asymmetric) matrix (where the subscript A stands for asymmetric) initialized with entries s_{ij} between 0 and 1, not at all similar and most similar, respectively. The steps of the algorithm are as follows:

- Normalize the matrix $\mathbf{S}_A$. Let

$$d_i = \sqrt{\frac{1}{\sum_j [\mathbf{S}_A]_{ij}}}$$

and $\mathbf{D}_A = \text{diag}\,(d_i)$ such that

$$\mathbf{H}_A = \mathbf{I} - \frac{1}{2}\mathbf{D}_A(\mathbf{S}_A + \mathbf{S}_A^\top)\mathbf{D}_A$$

- Compute the smallest eigenvectors of the scatter matrix $\mathbf{H}_A = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ where $\mathbf{U}$ is a matrix of eigenvectors and a $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues.

- Normalize the rows of $\mathbf{U}_k$ to unit length. $\mathbf{U}_k$ is a matrix of the bottom k smallest eigenvectors with $\mathbf{u}_1$, $\mathbf{u}_2$, ... arranged in a matrix.
- Perform a k-means clustering on normalized $\mathbf{U}_k$.

The algorithm allows us to start with an asymmetric distance matrix such as

$$\mathbf{S}_A = \exp\left(-\alpha \mathbf{A}_D\right)$$

where the operator $\exp(\bullet)$ is to be understood element-wise and $\alpha > 0$ is a scalar parameter.

Note that even while asymmetric spectral clustering also starts with a symmetric matrix $\mathbf{S}_A + \mathbf{S}_A^\top$, the normalization matrix $\mathbf{D}_A$ is computed based on the asymmetric matrix $\mathbf{S}_A$, unlike (symmetric) spectral clustering where the normalization $\mathbf{D}_S$ is based on the already symmetric $\mathbf{S}_S$. The row index i of the matrix $\mathbf{S}_A$ represents data points and the column index j represents estimated curves or surfaces, and the entry $[\mathbf{S}_A]_{ij}$ can be interpreted as a support weight of the data point i in favor of the estimated curve or surface j. The normalization $\mathbf{D}_A$ makes all data points cast unit support in favor of the total number of estimated curves and surfaces, and the asymmetric clustering tends to avoid separating the data point from the curve or surface it has large support for.

3.5. *Polynomial grouping*

Once a spectral clustering of data is available, we may use it to seed an algorithm with potentially higher accuracy but higher sensitivity to the initial state. An algorithm in flavor of k-planes using the nonlinear Koopmans method [29] or consistent algebraic least squares [20,22] for estimation and Euclidean distance for projection can be used to refine both data point grouping and parameter estimates.

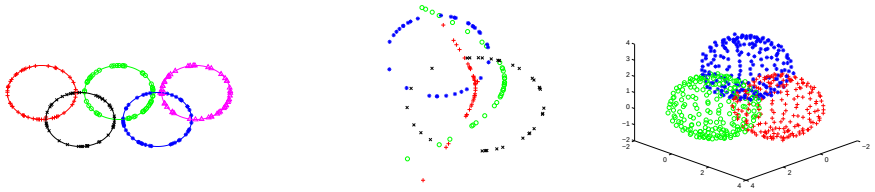
The steps of the algorithm [15,16,17], which resemble standard k-means, alternate as follows:

- *Estimation.* For each cluster, we find the best-fit surface that minimizes μ in the nonlinear estimator (in the manner discussed in Section 3.1).
- *Projection.* For each data point, we find the closest surface that minimizes the distance between the data point and its foot point on the surface (in the manner discussed in Section 3.3).

Only after a few iterations, the algorithm has converged to its final state with no data points re-grouped from one cluster to another.

4. Examples

As with hybrid linear modeling, the difficulty of the problem lies with intersecting and overlapping data sets. Here, we demonstrate the robustness of the algorithm



(a) (b) (c)

Fig. 4: Cluster assignments on some artificial data sets.

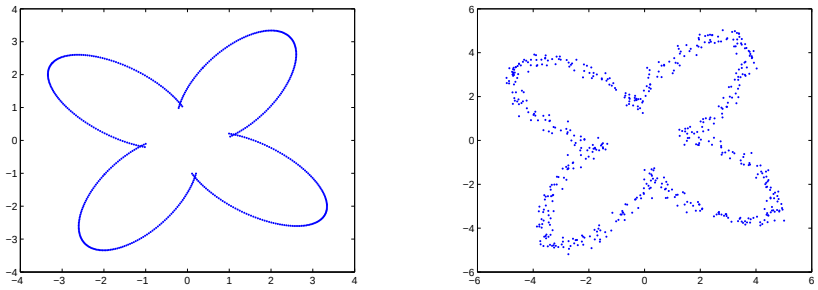
with some artificial data sets that illustrate how the proposed algorithm can tackle these cases.

Figure 4 illustrates cluster outcomes when the algorithm is executed on various artificial data sets used in [4], either with no noise (data points along five intersecting circles in Figure 4a, and data points along a circle, an ellipse, a parabola and a hyperbola in Figure 4b) or low noise level (three intersecting spheres in Figure 4c).

Figure 5a shows the "petal" data set, with data points arranged along elliptical leaves with no data points in the central region. The points are contaminated with a noise of $\sigma = 0.125$ along both axes x and y, see Figure 5b. The different stages of the algorithm are illustrated in Figures 5c and 5d. Figure 5c shows the data point grouping discovered by spectral clustering where the asymmetric distance matrix has been constructed such that it measures the distance of each point from its foot point, while Figure 5d shows how an extended version of k-planes clustering can refine the groups such that the shapes that capture the data almost match the originals the data has been generated from.

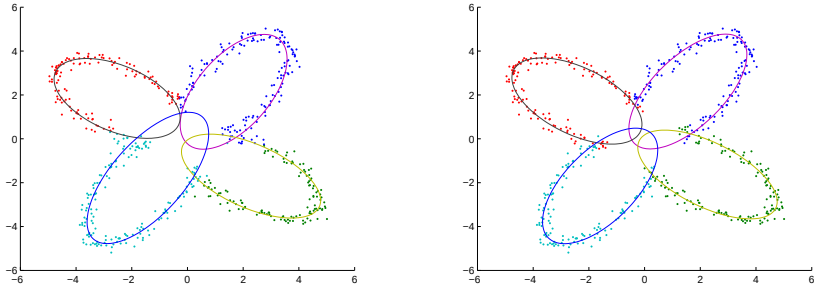
Figures 6a and 6b show how other algorithms fare with the same data set. The K-manifolds algorithm [27] ran with default options and was set to look for four one-dimensional manifolds where each manifold would correspond to an ellipse. The cluster assignment in Figure 6a reveals how the K-manifolds algorithm fails to capture the presence of four distinct shapes and tends towards treating the entire data set as a whole, splitting the data set in a rather arbitrary way. This highlights a major strength of our algorithm, namely that it can not only group data points but also identify the relationship that explains them, which helps produce more insightful groupings. The weakness of the K-manifolds algorithm lies with the junctions where it fails to detect the discontinuity (i.e. split data points between the two ellipses involved). Similar results are obtained with Kernel Spectral Curvature Clustering (KSCC) [4,5], which are shown in Figure 6b. As a kernel function, we used the standard quadratic polynomial kernel $k(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle + \langle \mathbf{x}^2, \mathbf{y}^2 \rangle$ where the operator $\bullet^2$ is to be understood element-wise and $\langle \bullet, \bullet \rangle$ stands for dot product.

The noise-resilience of the algorithm and its robustness against uneven distribution of samples is demonstrated in Figures 7a, 7b, 7c and 7d, which show the



(a) original data set

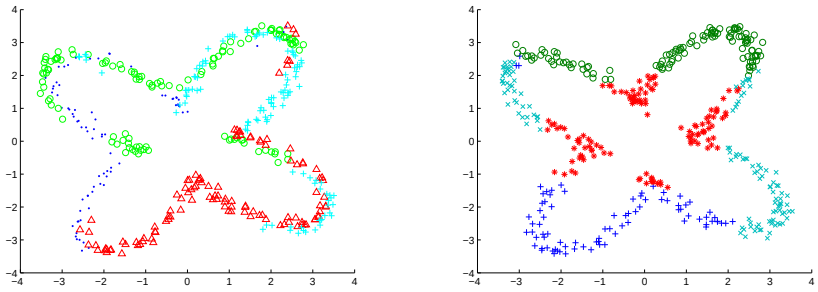
(b) original data set polluted with noise



(c) spectral clustering

(d) modified k-planes clustering

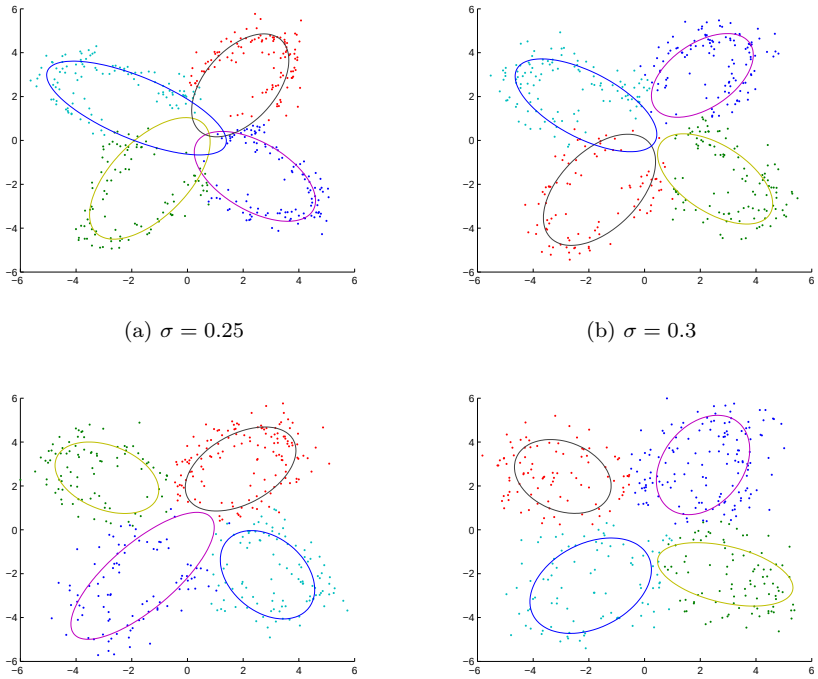
Fig. 5: Grouping data points arranged along quadratic curves with the curves intersecting and the groups slightly overlapping.



(a) K-manifolds algorithm

(b) Kernel Spectral Curvature Clustering

Fig. 6: Cluster assignments by other algorithms executed on the same data set.



(a) $\sigma = 0.25$ (b) $\sigma = 0.3$

(c) $\sigma = 0.4$ (d) $\sigma = 0.5$

Fig. 7: Robustness of the algorithm in response to increasing level of noise.

outcome of the spectral clustering phase with increasing levels of noise contaminating the original data set, $\sigma = 0.25$, $\sigma = 0.3$ and $\sigma = 0.5$, respectively. The distribution of data points is not even: points in the top right quadrant are twice as numerous as points in the bottom left quadrant, while the other two quadrants have approximately equal number of points. Even with substantial noise, when the points themselves hardly lie along ellipses any more, the algorithm is able to identify the four different shapes, albeit with worse precision.

5. Conclusions

We have presented a hybrid nonlinear manifold clustering method to build a model of clusters where intra-cluster data points are related by linear or quadratic functions. The key points of the method are parameter estimation based on data and noise covariance matrices with noise covariance matrix estimated from data, data point projection to linear and quadratic surfaces, and spectral clustering based on an asymmetric distance matrix. The method can tackle disjoint and intersecting data sets, and is capable of not only finding data clusters but also captures the

underlying relationship that explains data in a cluster.

Future work includes extending the method to higher-order functions, automatically identifying the number of clusters and reducing computational complexity.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. István Vajk is affiliated with MTA-BME Control Research Group. This work was supported by the fund of the Hungarian Academy of Sciences for control research and partially by the European Union and the European Social Fund through project FuturICT.hu (grant number TÁMOP-4.2.2.C-11/1/KONV-2012-0013).

References

1. Stefan Van Aelsta, Xiaogang (Steven) Wangb, Ruben H. Zamarc, and Rong Zhud. Linear grouping using orthogonal regression. *Computational Statistics and Data Analysis*, 50(5):1287–1312, March 2006.
2. Martin Aigner and Bert Jüttler. Robust fitting of implicitly defined surfaces using Gauss–Newton-type techniques. *The Visual Computer: International Journal of Computer Graphics*, 25(8):731–741, 2009.
3. Paul S. Bradley and Olvi L. Mangasarian. k-plane clustering. *Journal of Global Optimization*, 16(1):23–32, 2000.
4. Guangliang Chen, Stefan Atev, and Gilad Lerman. Kernel spectral curvature clustering (KSCC). In *Proc. of IEEE International Conference on Computer Vision (ICCV 2009)*, pages 765–772, Kyoto, Japan, September 2009.
5. Guangliang Chen and Gilad M. Lerman. Spectral curvature clustering. *International Journal of Computer Vision*, 81(3):317–330, 2009.
6. Nikolai Chernov and H. Ma. Least squares fitting of quadratic curves and surfaces. *Computer Vision*, pages 285–302, 2011.
7. Wojciech Chojnacki, Michael J. Brooks, Anton van den Hengel, and Darren Gawley. FNS, CFNS and HEIV: A unifying approach. *Journal of Mathematical Imaging and Vision*, 23:175–183, 2005.
8. David Eberly. Distance from point to a general quadratic curve or a general quadric surface, 1998, 2008. `http://www.geometrictools.com/Documentation/`.
9. David Eberly. Distance from a point to an ellipse in 2D, 2004. `www.geometrictools.com/Documentation/`.
10. Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2790–2797, 2009.
11. Andrew W. Fitzgibbon, Maurizio Pilu, and Robert B. Fisher. Direct least squares fitting of ellipses. *IEEE Trans. PAMI*, 21:476–480, 1999.
12. Alvina Goh and René Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
13. Gaël Guennebaud and Markus Gross. Algebraic point set surfaces. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 23, New York, NY, USA, 2007. ACM.
14. Sabine Van Huffel and Philippe Lemmerling, editors. *TLS and EIV modeling*. Kluwer, 2002.
15. Levente Hunyadi and István Vajk. Fitting a model to noisy data using low-order im-

plicit curves and surfaces. In *Proc. of the 2nd Eastern European Regional Conference on the Engineering of Computer Based Systems*, pages 106–114, September 2011.
16. Levente Hunyadi and István Vajk. Identifying unstructured systems in the errors-in-variables context. In *Proc. of the 18th World Congress of the International Federation of Automatic Control*, pages 13104–13109, August–September 2011.
17. Levente Hunyadi and István Vajk. Reconstructing a model of implicit shapes from an unorganized point set. *Scientific Bulletin of the "Politehnica" University of Timisoara, Romania, Transactions on Automatic Control and Computer Science*, 56(70):57–64, 2011.
18. Kenichi Kanatani and Prasanna Rangarajan. Hyper least squares fitting of circles and ellipses. *Computational Statistics and Data Analysis*, 55:2197–2208, 2011.
19. Oluwasanmi Koyejo and Joydeep Ghosh. MiPPS: A generative model for multi-manifold clustering. In *Manifold Learning and its Applications: Papers from the AAAI Fall Symposium*, 2010.
20. Alexander G. Kukush, Ivan Markovsky, and Sabine Van Huffel. Consistent estimation in an implicit quadratic measurement error model. *Computational Statistics and Data Analysis*, 47(1):123–147, 2004.
21. Yi Ma, Harm Derksen, Wei Hong, and John Wright. Segmentation of multivariate mixed data via lossy coding and compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1546–1562, 2007.
22. Ivan Markovsky, Alexander G. Kukush, and Sabine Van Huffel. Consistent least squares fitting of ellipsoids. *Numerische Mathematik*, 98(1):177–194, 2004.
23. Marina Meilă and William Pentney. Clustering by weighted cuts in directed graphs. In *Proc. of the 2007 SIAM International Conference on Data Mining*, 2007.
24. Marina Meilă and Jianbo Shi. A random walks view of spectral segmentation. In T. Jaakkola and T. Richardson, editors, *Artificial Intelligence and Statistics (AISTATS)*, 2001.
25. Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, 2003.
26. Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.
27. Richard Souvenir and Robert Pless. Manifold clustering. In *Proceedings of the 10th International Conference on Computer Vision*, pages 648–653, 2005.
28. Françoise Tisseur and Karl Meerbergen. The quadratic eigenvalue problem. *SIAM Review*, 43(2):235–286, 2001.
29. István Vajk and Jenő Hetthéssy. Identification of nonlinear errors-in-variables models. *Automatica*, 39:2099–2107, 2003.
30. René Vidal. Subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, March 2011.
31. René Vidal, Yi Ma, and Shankar Sastry. Generalized principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(12):1945–1959, December 2005.
32. Ureerat Wattanachon and Chidchanok Lursinsap. SPSM: A new hybrid data clustering algorithm for nonlinear data analysis. *International Journal of Pattern Recognition and Artificial Intelligence*, 23(8):1701, 2009.
33. Teng Zhang, Arthur Szlam, Yi Wang, and Gilad Lerman. Hybrid linear modeling via local best-fit flats. *International Journal of Computer Vision*, June 2012.