Learning and Design of Principal Curves

Balázs Kégl, *Student Member, IEEE*, Adam Krzyzak, *Senior Member, IEEE*,
Tamás Linder, *Member, IEEE*, and Kenneth Zeger, *Fellow, IEEE*

Abstract—Principal curves have been defined as "self-consistent" smooth curves which pass through the "middle" of a d-dimensional probability distribution or data cloud. They give a summary of the data and also serve as an efficient feature extraction tool. We take a new approach by defining principal curves as continuous curves of a given length which minimize the expected squared distance between the curve and points of the space randomly chosen according to a given distribution. The new definition makes it possible to theoretically analyze principal curve learning from training data and it also leads to a new practical construction. Our theoretical learning scheme chooses a curve from a class of polygonal lines with k segments and with a given total length to minimize the average squared distance over n training points drawn independently. Convergence properties of this learning scheme are analyzed and a practical version of this theoretical algorithm is implemented. In each iteration of the algorithm, a new vertex is added to the polygonal line and the positions of the vertices are updated so that they minimize a penalized squared distance criterion. Simulation results demonstrate that the new algorithm compares favorably with previous methods, both in terms of performance and computational complexity, and is more robust to varying data models.

Index Terms—Learning systems, unsupervised learning, feature extraction, vector quantization, curve fitting, piecewise linear approximation.

✦

1 INTRODUCTION

PRINCIPAL component analysis is perhaps the best known technique in multivariate analysis and is used in dimension reduction, feature extraction, and in image coding and enhancement. Consider a d-dimensional random vector $\mathbf{X} = (X_1, \ldots, X_d)$ with finite second moments. The first principal component line for $\mathbf{X}$ is a straight line which has the property that the expected value of the squared Euclidean distance from $\mathbf{X}$ to the line is minimum among all straight lines. This property makes the first principal component a concise one-dimensional approximation to the distribution of $\mathbf{X}$ and the projection of $\mathbf{X}$ to this line gives the best linear summary of the data. For elliptical distributions, the first principal component is also *self-consistent*, i.e., any point of the line is the conditional expectation of $\mathbf{X}$ over those points of the space which project to this point.

Hastie [1] and Hastie and Stuetzle [2] (hereafter HS) generalized the self-consistency property of principal components and introduced the notion of *principal curves*. Let $\mathbf{f}(t) = (f_1(t), \ldots, f_d(t))$ be a smooth (infinitely differentiable) curve in $\mathbb{R}^d$ parametrized by $t \in \mathbb{R}$ and, for any $\mathbf{x} \in \mathbb{R}^d$, let $t_{\mathbf{f}}(\mathbf{x})$ denote the largest parameter value t for which the distance between $\mathbf{x}$ and $\mathbf{f}(t)$ is minimized (see Fig. 1). More formally, the *projection index* $t_{\mathbf{f}}(\mathbf{x})$ is defined by

$$t_{\mathbf{f}}(\mathbf{x}) = \sup\{t : \|\mathbf{x} - \mathbf{f}(t)\| = \inf_{\tau} \|\mathbf{x} - \mathbf{f}(\tau)\|\}, \quad (1)$$

where $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^d$.

By the HS definition, the smooth curve $\mathbf{f}(t)$ is a principal curve if the following hold:

1. $\mathbf{f}$ does not intersect itself,
2. $\mathbf{f}$ has finite length inside any bounded subset of $\mathbb{R}^d$, and
3. $\mathbf{f}$ is self-consistent, i.e., $\mathbf{f}(t) = E(\mathbf{X}|t_{\mathbf{f}}(\mathbf{X}) = t)$.

Intuitively, self-consistency means that each point of $\mathbf{f}$ is the average (under the distribution of $\mathbf{X}$) of all points that project there. Thus, principal curves are smooth self-consistent curves which pass through the "middle" of the distribution and provide a good one-dimensional nonlinear summary of the data.

Based on the self-consistency property, HS developed an algorithm for constructing principal curves. Similar in spirit to the Generalized Lloyd Algorithm (GLA) of vector quantizer design [3], the HS algorithm iterates between a projection step and an expectation step. When the probability density of $\mathbf{X}$ is known, the HS principal algorithm for constructing principal curves is the following:

Step 0. Let $\mathbf{f}^{(0)}(t)$ be the first principal component line for $\mathbf{X}$. Set $j = 1$.

Step 1. Define $\mathbf{f}^{(j)}(t) = E\big(\mathbf{X}|t_{\mathbf{f}^{(j-1)}}(\mathbf{X}) = t\big)$.

Step 2. Set

$$t_{\mathbf{f}^{(j)}}(\mathbf{x}) = \max\{t : \|\mathbf{x} - \mathbf{f}^{(j)}(t)\| = \min_{\tau} \|\mathbf{x} - \mathbf{f}^{(j)}(\tau)\|\}$$

for all $\mathbf{x} \in \mathbb{R}^d$.

Step 3. Compute

$$\Delta(\mathbf{f}^{(j)}) = E\|\mathbf{X} - \mathbf{f}^{(j)}(t_{\mathbf{f}^{(j)}}(\mathbf{X}))\|^2.$$

If $|\Delta(\mathbf{f}^{(j)}) - \Delta(\mathbf{f}^{(j-1)})| < threshold$, then Stop. Otherwise, let $j = j + 1$ and go to Step 1.

- B. Kégl and T. Linder are with the Department of Mathematics and Statistics, Queen's University, Kingston, Ontario, Canada K7L 3N6. E-mail: {kegl, linder}@mast.queensu.ca.
- A. Krzyzak is with the Department of Computer Science, Concordia University, 1450 de Maisonneuve Blvd. West, Montreal PQ, Canada H3G 1M8. E-mail: krzyzak@cs.concordia.ca.
- K. Zeger is with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093-0407. E-mail: zeger@ucsd.edu.
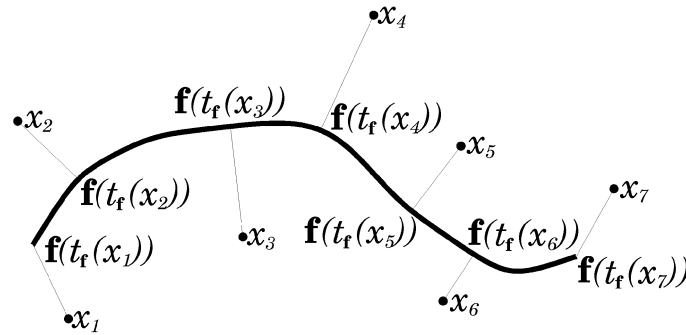
Fig. 1. Projecting points to a curve.

In practice, the distribution of $\mathbf{X}$ is often unknown, but a data set consisting of n samples of the underlying distribution is known instead. In the HS algorithm for data sets, the expectation in Step 1 is replaced by a "smoother" (locally weighted running lines [4]) or a nonparametric regression estimate (cubic smoothing splines). HS provide simulation examples to illustrate the behavior of the algorithm and describe an application in the Stanford Linear Collider Project [2]. It should be noted that there is no known proof of the convergence of the hypothetical algorithm (Steps 0-3; one main difficulty is that Step 1 can produce nondifferentiable curves, while principal curves are differentiable by definition). However, extensive testing on simulated and real examples have not revealed any convergence problems for the practical implementation of the algorithm [2].

Alternative definitions and methods for estimating principal curves have been given subsequent to Hastie and Stuetzle's groundbreaking work. Banfield and Raftery [5] (hereafter BR) modeled the outlines of ice floes in satellite images by closed principal curves and they developed a robust method which reduces the bias in the estimation process. Their method of clustering about principal curves led to a fully automatic method for identifying ice floes and their outlines. Singh et al. [6] used principal curves to extract skeletal structures of hand-written characters in faded documents. Reinhard and Niranjan [7] applied principal curves to model the short time spectrum of speech signals. They found that principal curves can be used efficiently to capture transitional information between phones. Chang and Ghosh [8], [9] combined the HS and the BR algorithms to improve the performance of the principal curve algorithm and used the modified algorithm for nonlinear feature extraction and pattern classification. On the theoretical side, Tibshirani [10] introduced a semiparametric model for principal curves and proposed a method for estimating principal curves using the EM algorithm. Close connections between principal curves and Kohonen's self-organizing maps were pointed out by Mulier and Cherkassky [11]. Recently, Delicado [12] proposed yet another definition based on a property of the first principal components of multivariate normal distributions.

There remains an unsatisfactory aspect of the definition of principal curves in the original HS paper, as well as in subsequent works. Although principal curves have been defined to be *nonparametric*, their existence has been proven only for some special densities, such as radially symmetric densities, and for the uniform density on a rectangle or an annulus in the plane [13]. At present, it is an open problem whether HS principal curves exist for all "reasonable" densities. It is also unknown how the hypothetical HS algorithm behaves for a probability density for which an HS principal curve does not exist. At the same time, the problem of existence makes it difficult to theoretically analyze (in terms of consistency and convergence rates) any estimation scheme for HS principal curves.

In this paper, we propose a new definition of principal curves to resolve this problem. In the new definition, a principal curve is a continuous curve of a given length L which minimizes the expected squared distance between $\mathbf{X}$ and the curve. In Section 2 (Lemma 1), we prove that, for any $\mathbf{X}$ with finite second moments, there always exists a principal curve in the new sense. We also discuss connections between the newly defined principal curves and optimal vector quantizers. Then, we propose a theoretical learning scheme in which the model classes are polygonal lines with k segments and with a given length, and the algorithm chooses a curve from this class which minimizes the average squared distance over n training points. In Theorem 1, we prove that, with k suitably chosen as a function of n, the expected squared distance of the curve trained on n data points converges to the expected squared distance of the principal curve at a rate $O(n^{-1/3})$ as $n \to \infty$.

Two main features distinguish this learning scheme from the HS algorithm. First, the polygonal line estimate of the principal curve is determined via minimizing a data dependent criterion directly related to the definition of principal curves. This facilitates the theoretical analysis of the performance. Second, the complexity of the resulting polygonal line is determined by the number of segments k, which, when optimally chosen, is typically much less than n. This agrees with our mental image that principal curves should provide a concise summary of the data. In contrast, for n data points, the HS algorithm with scatterplot smoothing produces polygonal lines with n segments.

Though amenable to analysis, our theoretical algorithm is computationally burdensome for implementation. In Section 3, we develop a suboptimal algorithm for learning principal curves. The practical algorithm produces polygonal line approximations to the principal curve just as the theoretical method does, but global optimization is replaced by a less complex iterative descent method. In Section 4, we give simulation results and compare our algorithm with

previous work. In general, on examples considered by HS, the performance of the new algorithm is comparable with the HS algorithm, while it proves to be more robust to changes in the data generating model. In Section 4, we also report some preliminary results on applying the polygonal line algorithm to find the medial axes ("skeletons") of pixel templates of handwritten characters.

2 LEARNING PRINCIPAL CURVES WITH A LENGTH CONSTRAINT

A *curve* in d-dimensional Euclidean space is a continuous function $\mathbf{f} : I \to \mathbb{R}^d$, where I is a closed interval of the real line. Let the expected squared distance between $\mathbf{X}$ and $\mathbf{f}$ be denoted by

$$\Delta(\mathbf{f}) = E\big[\inf_t \|\mathbf{X} - \mathbf{f}(t)\|^2\big] = E\|\mathbf{X} - \mathbf{f}(t_\mathbf{f}(\mathbf{X}))\|^2, \quad (2)$$

where the projection index $t_\mathbf{f}(\mathbf{x})$ is given in (1). Let $\mathbf{f}$ be a smooth (infinitely differentiable) curve and, for $\lambda \in \mathbb{R}$, consider the perturbation $\mathbf{f} + \lambda\mathbf{g}$ of $\mathbf{f}$ by a smooth curve $\mathbf{g}$ such that $\sup_t \|\mathbf{g}(t)\| \leq 1$ and $\sup_t \|\mathbf{g}'(t)\| \leq 1$. HS proved that $\mathbf{f}$ is a principal curve, if and only if, $\mathbf{f}$ is a critical point of the distance function in the sense that, for all such $\mathbf{g}$,

$$\frac{\partial \Delta(\mathbf{f} + \lambda\mathbf{g})}{\partial \lambda}\bigg|_{\lambda=0} = 0.$$

It is not hard to see that an analogous result holds for principal component lines if the perturbation $\mathbf{g}$ is a straight line. In this sense, the HS principal curve definition is a natural generalization of principal components. Also, it is easy to check that principal components are in fact principal curves if the distribution of $\mathbf{X}$ is elliptical.

An unfortunate property of the HS definition is that, in general, it is not known if principal curves exist for a given source density. To resolve this problem, we go back to the defining property of the first principal component. A straight line $\mathbf{s}(t)$ is the first principal component, if and only if,

$$E\big[\min_t \|\mathbf{X} - \mathbf{s}(t)\|^2\big] \leq E\big[\min_t \|\mathbf{X} - \hat{\mathbf{s}}(t)\|^2\big]$$

for any other straight line $\hat{\mathbf{s}}(t)$. We would like to generalize this property of the first principal component and define principal curves so that they minimize the expected squared distance over a class of curves, rather than only being critical points of the distance function. To do this, it is necessary to constrain the length[1] of the curve since, otherwise, for any $\mathbf{X}$ with a density and any $\epsilon > 0$, there exists a smooth curve $\mathbf{f}$ such that $\Delta(\mathbf{f}) \leq \epsilon$ and, thus, a minimizing $\mathbf{f}$ has infinite length. On the other hand, if the distribution of $\mathbf{X}$ is concentrated on a polygonal line and is uniform there, the infimum of the squared distances $\Delta(\mathbf{f})$ is 0 over the class of smooth curves, but no smooth curve can achieve this infimum. For this reason, we relax the requirement that $\mathbf{f}$ be differentiable and, instead, we constrain the length of $\mathbf{f}$. Note that, by the definition of curves, $\mathbf{f}$ is still continuous. We give the following new definition of principal curves:

1. For the definition of length for nondifferentiable curves see Appendix A where some basic facts concerning curves in $\mathbb{R}^d$ have been collected from [14].

Definition 1. *A curve $\mathbf{f}^*$ is called a principal curve of length L for $\mathbf{X}$ if $\mathbf{f}^*$ minimizes $\Delta(\mathbf{f})$ over all curves of length less than or equal to L.*

A useful advantage of the new definition is that principal curves of length L always exist if $\mathbf{X}$ has finite second moments, as the next result shows.

Lemma 1. *Assume that $E\|\mathbf{X}\|^2 < \infty$. Then, for any $L > 0$, there exists a curve $\mathbf{f}^*$ with $l(\mathbf{f}^*) \leq L$ such that*

$$\Delta(\mathbf{f}^*) = \inf\{\Delta(\mathbf{f}) : l(\mathbf{f}) \leq L\}.$$

The proof of the lemma is given in Appendix A.

Note that we have dropped the requirement of the HS definition that principal curves be nonintersecting. In fact, Lemma 1 does not hold, in general, for nonintersecting curves of length L without further restricting the distribution of $\mathbf{X}$, since there are distributions for which the minimum of $\Delta(\mathbf{f})$ is achieved only by an intersecting curve, even though nonintersecting curves can arbitrarily approach this minimum. Note also that neither the HS nor our definition guarantees the uniqueness of principal curves. In our case, there might exist several principal curves for a given length constraint L, but each of these will have the same (minimal) squared loss.

Remark (Connection with vector quantization). Our new definition of principal curves has been inspired by the notion of an optimal vector quantizer. A *vector quantizer* maps a point in $\mathbb{R}^d$ to the closest point in a fixed set (called a codebook) $\{y_1, \ldots, y_k\} \subset \mathbb{R}^d$. The codepoints $\mathbf{y}_1^*, \ldots, \mathbf{y}_k^* \in \mathbb{R}^d$ correspond to an optimal k-point vector quantizer if

$$E\big[\min_i \|\mathbf{X} - \mathbf{y}_i^*\|^2\big] \leq E\big[\min_i \|\mathbf{X} - \mathbf{y}_i\|^2\big]$$

for any other collection of k-points $\mathbf{y}_1, \ldots, \mathbf{y}_k \in \mathbb{R}^d$. In other words, the points $\mathbf{y}_1^*, \ldots, \mathbf{y}_k^*$ give the best k-point representation of $\mathbf{X}$ in the mean squared sense. Optimal vector quantizers are important in lossy data compression, speech, and image coding [15], and clustering [16]. There is a strong connection between the definition of an optimal vector quantizer and our definition of a principal curve. Both minimize the same expected squared distance criterion. A vector quantizer is constrained to have at most k points, whereas a principal curve has a constrained length. This connection is further illustrated by a recent work of Tarpey et al. [17] who define points $\mathbf{y}_1, \ldots, \mathbf{y}_k$ to be self-consistent if

$$\mathbf{y}_i = E[\mathbf{X}|\mathbf{X} \in S_i],$$

where $S_1, \ldots, S_k$ are the "Voronoi regions" defined as $S_i = \{\mathbf{x} : \|\mathbf{x} - \mathbf{y}_i\| \leq \|\mathbf{x} - \mathbf{y}_j\|, j = 1, \ldots, k\}$ (ties are broken arbitrarily). Thus, our principal curves correspond to optimal vector quantizers ("principal points" by the terminology of [17]), while the HS principal curves correspond to self-consistent points.

While principal curves of a given length always exist, it appears difficult to demonstrate concrete examples unless the distribution of $\mathbf{X}$ is discrete or is concentrated on a curve. It is presently unknown what principal curves look

like with a length constraint for even the simplest continuous multivariate distributions such as the Gaussian. However, this fact in itself does not limit the practical significance of principal curves. Analogously, for $k \geq 3$ codepoints, there are no known concrete examples of optimal vector quantizers for even the most common model distributions such as Gaussian, Laplacian, or uniform (in a hypercube) in any dimensions $d \geq 2$. Nevertheless, algorithms for quantizer design attempting to find near optimal vector quantizers are of great theoretical and practical interest. In what follows, we consider the problem of principal curve design based on training data.

Suppose that n independent copies $\mathbf{X}_1, \ldots, \mathbf{X}_n$ of $\mathbf{X}$ are given. These are called the *training data* and they are assumed to be independent of $\mathbf{X}$. The goal is to use the training data to construct a curve of length at most L whose expected squared loss is close to that of a principal curve for $\mathbf{X}$.

Our method is based on a common model in statistical learning theory (e.g., see [18]). We consider classes $\mathcal{S}_1, \mathcal{S}_2, \ldots,$ of curves of increasing complexity. Given n data points drawn independently from the distribution of $\mathbf{X}$, we choose a curve as the estimator of the principal curve from the kth model class $\mathcal{S}_k$ by minimizing the empirical error. By choosing the complexity of the model class appropriately as the size of the training data grows, the chosen curve represents the principal curve with increasing accuracy.

We assume that the distribution of $\mathbf{X}$ is concentrated on a closed and bounded convex set $K \subset \mathbb{R}^d$. A basic property of convex sets in $\mathbb{R}^d$ shows that there exists a principal curve of length L *inside* K (see [19], Lemma 1) and, so, we will only consider curves in K.

Let $\mathcal{S}$ denote the family of curves taking values in K and having length not greater than L. For $k \geq 1$, let $\mathcal{S}_k$ be the set of polygonal (piecewise linear) curves in K which have k segments and whose lengths do not exceed L. Note that $\mathcal{S}_k \subset \mathcal{S}$ for all k. Let

$$\Delta(\mathbf{x}, \mathbf{f}) = \min_t \|\mathbf{x} - \mathbf{f}(t)\|^2 \qquad (3)$$

denote the squared distance between a point $\mathbf{x} \in \mathbb{R}^d$ and the curve $\mathbf{f}$. For any $\mathbf{f} \in \mathcal{S}$, the empirical squared error of $\mathbf{f}$ on the training data is the sample average

$$\Delta_n(\mathbf{f}) = \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{X}_i, \mathbf{f}), \qquad (4)$$

where we have suppressed in the notation the dependence of $\Delta_n(\mathbf{f})$ on the training data. Let our theoretical algorithm[2] choose an $\mathbf{f}_{k,n} \in \mathcal{S}_k$ which minimizes the empirical error, i.e,

$$\mathbf{f}_{k,n} = \arg \min_{\mathbf{f} \in \mathcal{S}_k} \Delta_n(\mathbf{f}). \qquad (5)$$

We measure the efficiency of $\mathbf{f}_{k,n}$ in estimating $\mathbf{f}^*$ by the difference $J(\mathbf{f}_{k,n})$ between the expected squared loss of $\mathbf{f}_{k,n}$ and the optimal expected squared loss achieved by $\mathbf{f}^*$, i.e., we let

2. The term "hypothetical algorithm" might appear to be more accurate since we have not shown that an algorithm for finding $\mathbf{f}_{k,n}$ exists. However, an algorithm clearly exists which can approximate $\mathbf{f}_{k,n}$ with arbitrary accuracy in a finite number of steps (consider polygonal lines whose vertices are restricted to a fine rectangular grid). The proof of Theorem 1 shows that such approximating curves can replace $\mathbf{f}_{k,n}$ in the analysis.

$$J(\mathbf{f}_{k,n}) = \Delta(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}^*) = \Delta(\mathbf{f}_{k,n}) - \min_{\mathbf{f} \in \mathcal{S}} \Delta(\mathbf{f}).$$

Since $\mathcal{S}_k \subset \mathcal{S}$, we have $J(\mathbf{f}_{k,n}) \geq 0$. Our main result in this section proves that if the number of data points n tends to infinity, and k is chosen to be proportional to $n^{1/3}$, then $J(\mathbf{f}_{k,n})$ tends to zero at a rate $J(\mathbf{f}_{k,n}) = O(n^{-1/3})$.

Theorem 1. *Assume that* $\mathbf{P}\{\mathbf{X} \in K\} = 1$ *for a bounded and closed convex set* K, *let* n *be the number of training points, and let* k *be chosen to be proportional to* $n^{1/3}$. *Then, the expected squared loss of the empirically optimal polygonal line with* k *segments and length at most* L *converges, as* $n \to \infty$, *to the squared loss of the principal curve of length* L *at a rate*

$$J(\mathbf{f}_{k,n}) = O(n^{-1/3}).$$

The proof of the theorem is given in Appendix B. To establish the result, we use techniques from statistical learning theory (e.g., see [20]). First, the approximating capability of the class of curves $\mathcal{S}_k$ is considered and then the estimation (generalization) error is bounded via covering the class of curves $\mathcal{S}_k$ with ϵ accuracy (in the squared distance sense) by a discrete set of curves. When these two bounds are combined, one obtains

$$J(\mathbf{f}_{k,n}) \leq \sqrt{\frac{kC(L, D, d)}{n}} + \frac{DL + 2}{k} + O(n^{-1/2}), \qquad (6)$$

where the term $C(L, D, d)$ depends only on the dimension d, the length L, and the diameter D of the support of $\mathbf{X}$, but is independent of k and n. The two error terms are balanced by choosing k to be proportional to $n^{1/3}$, which gives the convergence rate of Theorem 1.

Note that, although the constant hidden in the O notation depends on the dimension d, the exponent of n is dimension-free. This is not surprising in view of the fact that the class of curves $\mathcal{S}$ is equivalent in a certain sense to the class of Lipschitz functions $\mathbf{f} : [0, 1] \to K$ such that $\|\mathbf{f}(x) - \mathbf{f}(y)\| \leq L|x - y|$ (see Appendix A). It is known that the ϵ-entropy, defined by the logarithm of the ϵ covering number, is roughly proportional to $1/\epsilon$ for such function classes [21]. Using this result, the convergence rate $O(n^{-1/3})$ can be obtained by considering ϵ-covers of $\mathcal{S}$ directly (without using the model classes $\mathcal{S}_k$) and picking the empirically optimal curve in this cover. The use of the classes $\mathcal{S}_k$ has the advantage that they are directly related to the practical implementation of the algorithm given in the next section.

Note also that even though Theorem 1 is valid for any given length constraint L, the theoretical algorithm itself gives little guidance about how to choose L. This choice depends on the particular application and heuristic considerations are likely to enter here. One example is given in Section 3, where a practical implementation of the polygonal line algorithm is used to recover a "generating curve" from noisy observations.

Finally, we note that the proof of Theorem 1 also provides information on the distribution of the expected squared error of $\mathbf{f}_{k,n}$ given the training data $\mathbf{X}_1, \ldots, \mathbf{X}_n$. In particular, it is shown at the end of the proof that, for all n and k and δ such that $0 < \delta < 1$, with probability at least $1 - \delta$ we have
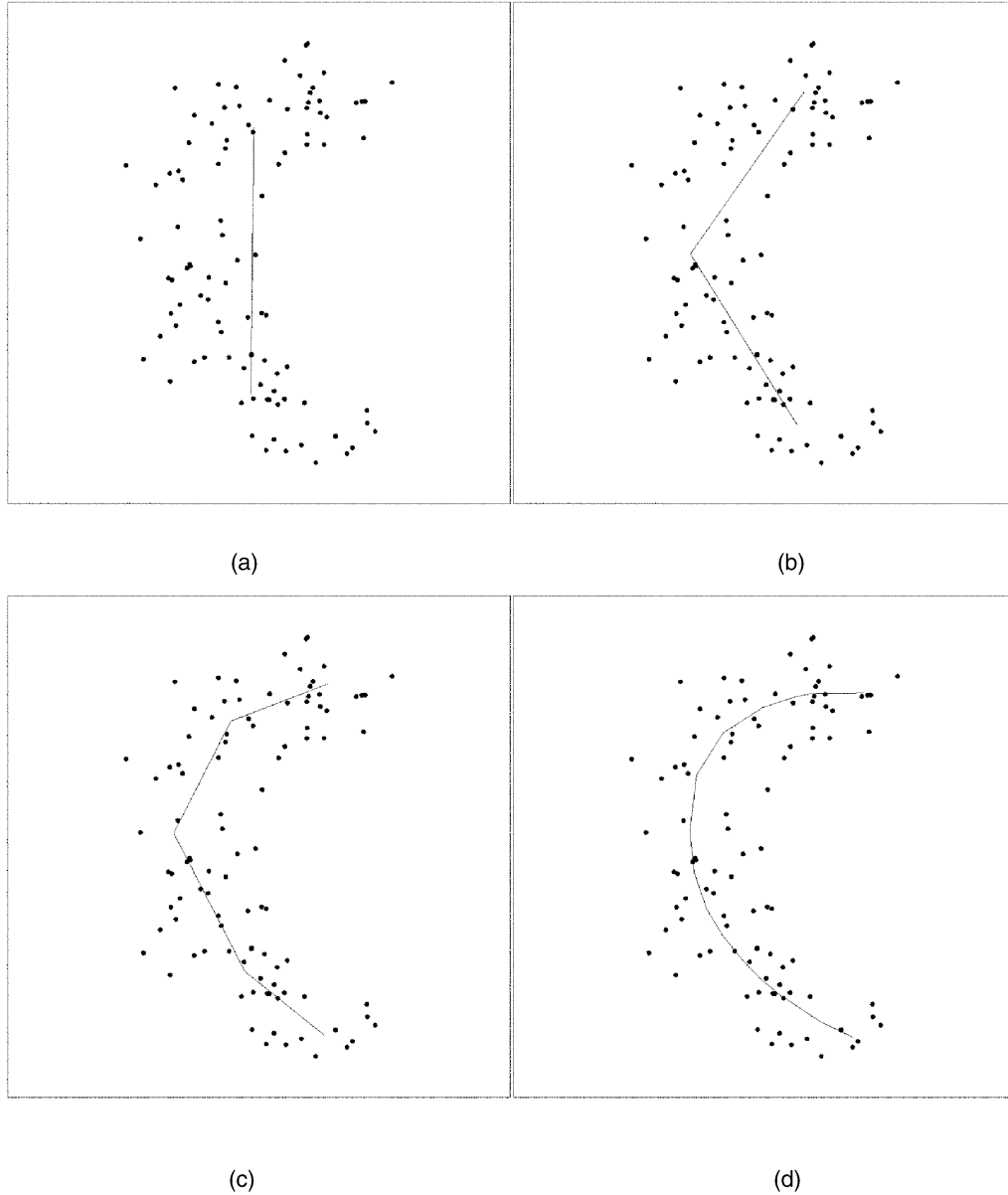
Fig. 2. The curves $\mathbf{f}_{k,n}$ produced by the polygonal line algorithm for $n = 100$ data points. The data was generated by adding independent Gaussian errors to both coordinates of a point chosen randomly on a half circle. (a) $\mathbf{f}_{1,n}$, (b) $\mathbf{f}_{2,n}$, (c) $\mathbf{f}_{4,n}$, and (d) $\mathbf{f}_{15,n}$ (the output of the algorithm).

$$E\big[\Delta(\mathbf{X}, \mathbf{f}_{k,n})|\mathbf{X}_1, \ldots, \mathbf{X}_n\big] - \Delta(\mathbf{f}^*)$$
$$\leq \sqrt{\frac{kC(L,D,d) - D^4 \log(\delta/2)}{n}} + \frac{DL+2}{k}, \qquad (7)$$

where $\log$ denotes natural logarithm and $C(L, D, d)$ is the same constant as in (6).

3 THE POLYGONAL LINE ALGORITHM

Given a set of data points $\mathcal{X}_n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^d$, the task of finding a polygonal curve with k segments and length L which minimizes $\frac{1}{n}\sum_{i=1}^n \Delta(\mathbf{x}_i, \mathbf{f})$ is computationally difficult. We propose a suboptimal method with reasonable complexity which also picks the length L of the principal curve automatically. The basic idea is to start with a straight line segment $\mathbf{f}_{0,n}$, the shortest segment of the first principal component line which contains all of the projected data points, and in each iteration of the algorithm, to increase the

number of segments by one by adding a new vertex to the polygonal line $\mathbf{f}_{k,n}$ produced in the previous iteration. After adding a new vertex, the positions of all vertices are updated in an inner loop.

The inner loop consists of a projection step and an optimization step. In the projection step, the data points are partitioned into "nearest neighbor regions" according to which segment or vertex they project. In the optimization step, the new position of a vertex $\mathbf{v}_i$ is determined by minimizing an average squared distance criterion penalized by a measure of the local curvature, while all other vertices are kept fixed. These two steps are iterated so that the optimization step is applied to each vertex $\mathbf{v}_i$, $i = 1, \ldots, k+1$ in a cyclic fashion (so that after $\mathbf{v}_{k+1}$, the procedure starts again with $\mathbf{v}_1$) until convergence is achieved and $\mathbf{f}_{k,n}$ is produced. Then, a new vertex is added.

The algorithm stops when k exceeds a threshold $c(n, \Delta)$. This stopping criterion is based on a heuristic complexity
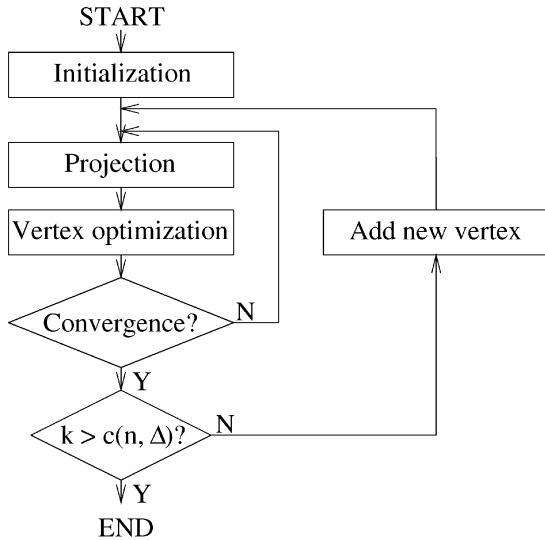
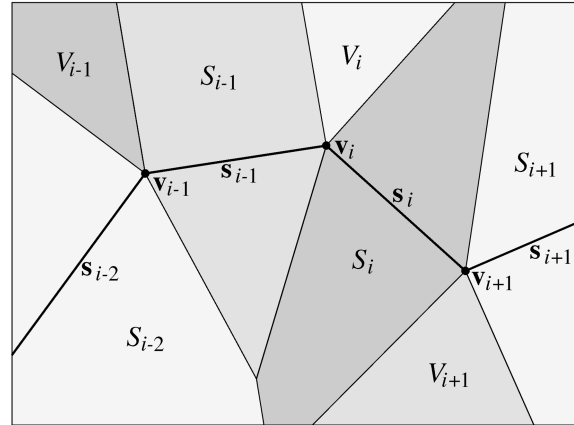Fig. 3. The flow chart of the polygonal line algorithm.

Fig. 4. A nearest-neighbor partition of $\mathbb{R}^2$ induced by the vertices and segments of $\mathbf{f}$. The nearest point of $\mathbf{f}$ to any point in the set V_i is the vertex $\mathbf{v}_i$. The nearest point of $\mathbf{f}$ to any point in the set S_i is a point of the line segment $\mathbf{s}_i$.

measure, determined by the number of segments k, the number of data points n, and the average squared distance $\Delta_n(\mathbf{f}_{k,n})$.

The flow chart of the algorithm is given in Fig. 3. The evolution of the curve produced by the algorithm is illustrated in Fig. 2. As with the HS algorithm, we have no formal proof that the practical algorithm will converge, but, in practice, after extensive testing, it seems to consistently converge.

It should be noted that the two core components of the algorithm, the projection and the vertex optimization steps, are combined with more heuristic elements such as the stopping condition and the form of the penalty term (8) of the optimization step. The heuristic parts of the algorithm have been tailored to the task of recovering an underlying generating curve for a distribution based on a finite data set of randomly drawn points (see the experimental results in Section 4). When the algorithm is intended for an application with a different objective (such as robust signal compression), the core components can be kept unchanged, but the heuristic elements may be replaced according to the new objectives.

3.1 The Projection Step

Let $\mathbf{f}$ denote a polygonal line with vertices $\mathbf{v}_1, \ldots, \mathbf{v}_{k+1}$ and line segments $\mathbf{s}_1, \ldots, \mathbf{s}_k$ such that $\mathbf{s}_i$ connects vertices $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$. In this step, the data set $\mathcal{X}_n$ is partitioned into (at most) $2k + 1$ disjoint sets $V_1, \ldots, V_{k+1}$ and $S_1, \ldots, S_k$, the nearest neighbor regions of the vertices and segments of $\mathbf{f}$, respectively, in the following manner: For any $\mathbf{x} \in \mathbb{R}^d$, let $\Delta(\mathbf{x}, \mathbf{s}_i)$ be the squared distance from $\mathbf{x}$ to $\mathbf{s}_i$ (see Definition 3), let $\Delta(\mathbf{x}, \mathbf{v}_i) = \|\mathbf{x} - \mathbf{v}_i\|^2$, and let

$$V_i = \big\{ \mathbf{x} \in \mathcal{X}_n : \Delta(\mathbf{x}, \mathbf{v}_i) = \Delta(\mathbf{x}, \mathbf{f}),$$
$$\Delta(\mathbf{x}, \mathbf{v}_i) < \Delta(\mathbf{x}, \mathbf{v}_m), m = 1, \ldots, i-1 \big\}.$$

Upon setting $V = \bigcup_{i=1}^{k+1} V_i$, the S_i sets are defined by

$$S_i = \big\{ \mathbf{x} \in \mathcal{X}_n : \mathbf{x} \notin V, \Delta(\mathbf{x}, \mathbf{s}_i) = \Delta(\mathbf{x}, \mathbf{f}),$$
$$\Delta(\mathbf{x}, \mathbf{s}_i) < \Delta(\mathbf{x}, \mathbf{s}_m), m = 1, \ldots, i-1 \big\}.$$

The resulting partition is illustrated in Fig. 4.

As a result of introducing the nearest neighbor regions S_i and V_i, the polygonal line algorithm substantially differs from methods based on the self-organizing map. Although we optimize the positions of the *vertices* of the curve, the distances of the data points are measured from the *segments* and vertices of the curve onto which they project, while the self-organizing map measures distances exclusively from the vertices. Our principle makes it possible to use a relatively small number of vertices and still obtain good approximation to an underlying generating curve.

3.2 The Vertex Optimization Step

In this step, the new position of a vertex $\mathbf{v}_i$ is determined. In the theoretical algorithm, the average squared distance $\Delta_n(\mathbf{x}, \mathbf{f})$ is minimized subject to the constraint that $\mathbf{f}$ is a polygonal line with k segments and length not exceeding L. One could use a Lagrangian formulation and attempt to find a new position for $\mathbf{v}_i$ (while all other vertices are fixed) such that the penalized squared error $\Delta_n(\mathbf{f}) + \lambda l(\mathbf{f})^2$ is minimum. Although this direct length penalty can work well in certain applications, it yields poor results in terms of recovering a smooth generating curve. In particular, this approach is very sensitive to the choice of λ and tends to produce curves which, similarly to the HS algorithm, exhibit a "flattening" estimation bias towards the center of the curvature.

To reduce the estimation bias, we penalize sharp angles between line segments. At inner vertices $\mathbf{v}_i$, $3 \leq i \leq k-1$, we penalize the sum of the cosines of the three angles at vertices $\mathbf{v}_{i-1}$, $\mathbf{v}_i$, and $\mathbf{v}_{i+1}$. The cosine function is convex in the interval $[\pi/2, \pi]$ and its derivative is zero at π, which makes it especially suitable for the steepest descent algorithm. To make the algorithm invariant under scaling, we multiply the cosines by the square of the "radius" of the data defined by $r = \max_{\mathbf{x} \in \mathcal{X}_n} \big\| \mathbf{x} - \frac{1}{n} \sum_{\mathbf{y} \in \mathcal{X}_n} \mathbf{y} \big\|$. Note that the chosen penalty formulation is related to the original principle of penalizing the length of the curve. At inner vertices, since only one vertex is moved at a time, penalizing sharp angles indirectly penalizes long segments. At the endpoints and at their immediate neighbors ($\mathbf{v}_i$, $i = 1, 2, k, k+1$), where penalizing sharp angles does not translate to penalizing long line segments, the penalty on a

nonexistent angle is replaced by a direct penalty on the squared length of the first (or last) segment.

Formally, let γ_i denote the angle at vertex $\mathbf{v}_i$, let $\pi(\mathbf{v}_i) = r^2(1 + \cos \gamma_i)$, let $\mu_+(\mathbf{v}_i) = \|\mathbf{v}_i - \mathbf{v}_{i+1}\|^2$, and let $\mu_-(\mathbf{v}_i) = \|\mathbf{v}_i - \mathbf{v}_{i-1}\|^2$. Then, the penalty $P(\mathbf{v}_i)$ at vertex $\mathbf{v}_i$ is given by

$$P(\mathbf{v}_i) = \begin{cases} \mu_+(\mathbf{v}_i) + \pi(\mathbf{v}_{i+1}) & \text{if } i = 1 \\ \mu_-(\mathbf{v}_i) + \pi(\mathbf{v}_i) + \pi(\mathbf{v}_{i+1}) & \text{if } i = 2 \\ \pi(\mathbf{v}_{i-1}) + \pi(\mathbf{v}_i) + \pi(\mathbf{v}_{i+1}) & \text{if } 2 < i < k \\ \pi(\mathbf{v}_{i-1}) + \pi(\mathbf{v}_i) + \mu_+(\mathbf{v}_i) & \text{if } i = k \\ \pi(\mathbf{v}_{i-1}) + \mu_-(\mathbf{v}_i) & \text{if } i = k + 1. \end{cases} \quad (8)$$

The local measure of the average squared distance is calculated from the data points which project to $\mathbf{v}_i$ or to the line segment(s) starting at $\mathbf{v}_i$ (see Projection Step). Accordingly, let

$$\sigma_+(\mathbf{v}_i) = \sum_{\mathbf{x} \in S_i} \Delta(\mathbf{x}, \mathbf{s}_i)$$

$$\sigma_-(\mathbf{v}_i) = \sum_{\mathbf{x} \in S_{i-1}} \Delta(\mathbf{x}, \mathbf{s}_{i-1})$$

$$\nu(\mathbf{v}_i) = \sum_{\mathbf{x} \in V_i} \Delta(\mathbf{x}, \mathbf{v}_i).$$

Now, define the local average squared distance as a function of $\mathbf{v}_i$ by

$$\Delta_n(\mathbf{v}_i) = \begin{cases} \nu(\mathbf{v}_i) + \sigma_+(\mathbf{v}_i) & \text{if } i = 1 \\ \sigma_-(\mathbf{v}_i) + \nu(\mathbf{v}_i) + \sigma_+(\mathbf{v}_i) & \text{if } 1 < i < k + 1 \\ \sigma_-(\mathbf{v}_i) + \nu(\mathbf{v}_i) & \text{if } i = k + 1. \end{cases} \quad (9)$$

We use an iterative steepest descent method to minimize

$$G(\mathbf{v}_i) = \frac{1}{n} \Delta_n(\mathbf{v}_i) + \lambda_p \frac{1}{k+1} P(\mathbf{v}_i),$$

where $\lambda_p > 0$. Note that the local squared distance term $\Delta_n(\mathbf{v}_i)$ and the local penalty term $P(\mathbf{v}_i)$ are normalized by the number of data points n and the number of vertices $(k+1)$, respectively. This normalization ensures that the magnitude of the global objective function $\sum_{i=1}^{k+1} G(\mathbf{v}_i)$ is approximately invariant to changes in the number of data points or the number of line segments.

We search for a local minimum of $G(\mathbf{v}_i)$ in the direction of the negative gradient of $G(\mathbf{v}_i)$ by using a procedure similar to Newton's method. Then, the gradient is recomputed and the line search is repeated. The iteration stops when the relative improvement of $G(\mathbf{v}_i)$ is less than a preset threshold. It should be noted that $\Delta_n(\mathbf{v}_i)$ is not differentiable at any point $\mathbf{v}_i$ such that at least one data point falls on the boundary of a nearest neighbor region S_{i-1}, S_i, or V_i. Thus, $G(\mathbf{v}_i)$ is only piecewise differentiable and the variant of Newton's method we use cannot guarantee that the global objective function $\sum_{i=1}^{k+1} G(\mathbf{v}_i)$ will always decrease in the optimization step. During extensive test runs, however, the algorithm was observed to always converge. Furthermore, we note that this part of the algorithm is modular, i.e., the procedure we are using can be substituted with a more sophisticated optimization routine at the expense of increased computational complexity.

One important issue is the amount of smoothing required for a given data set. In the HS algorithm, one needs to determine the penalty coefficient of the spline smoother or the span of the scatterplot smoother. In our algorithm, the corresponding parameter is the curvature penalty factor λ_p. If some a priori knowledge about the distribution is available, one can use it to determine the smoothing parameter. However in the absence of such knowledge, the coefficient should be data-dependent. Based on heuristic considerations explained below, and after carrying out practical experiments, we set

$$\lambda_p = \lambda_p' k n^{-1/3} \Delta_n(\mathbf{f}_{k,n})^{1/2} r^{-1},$$

where λ_p' is an experimentally determined constant.

By setting the penalty to be proportional to the average distance of the data points from the curve, we avoid the zigzagging behavior of the curve resulting from overfitting when the noise is relatively large. At the same time, this penalty factor allows the principal curve to closely follow the generating curve when the generating curve itself is a polygonal line with sharp angles and the data is concentrated on this curve (the noise is very small). The penalty is set to be proportional to the number of segments k because, in our experiments, we have found that the algorithm is more likely to avoid local minima if a small penalty is imposed initially and the penalty is gradually increased as the number of segments grows. Since the stopping condition (Section 3.4) indicates that the final number of line segments is proportional to the cube root of the data size, we normalize k by $n^{1/3}$ in the penalty term. The penalty factor is also normalized by the radius of the data to obtain scale independence. The value of the parameter λ_p' was determined by experiments and was set to a constant 0.13.

3.3 Adding a New Vertex

We start with the optimized $\mathbf{f}_{k,n}$ and choose the segment that has the largest number of data points projecting to it. If more than one such segment exists, we choose the longest one. The midpoint of this segment is selected as the new vertex. Formally, let

$$I = \{i : |S_i| \geq |S_j|, \; j = 1, \ldots, k\},$$

and $\ell = \arg\max_{i \in I} \|\mathbf{v}_i - \mathbf{v}_{i+1}\|$. Then, the new vertex is $\mathbf{v}_{new} = (\mathbf{v}_\ell + \mathbf{v}_{\ell+1})/2$.

3.4 Stopping Condition

According to the theoretical results of Section 2, the number of segments k is an important factor that controls the balance between the estimation and approximation errors and it should be proportional to $n^{1/3}$ to achieve the $O(n^{-1/3})$ convergence rate for the expected squared distance. Although the theoretical bounds are not tight enough to determine the optimal number of segments for a given data size, we found that $k \sim n^{1/3}$ works in practice. We also found that, similarly to the penalty factor λ_p, the final value of k should also depend on the average squared distance to achieve robustness. If the variance of the noise is relatively small, we can keep the approximation error low by allowing a relatively large number of segments. On the other hand, when the variance of the noise is large (implying a high estimation error), a low approximation error does not improve the overall performance significantly, so, in this case, a smaller number of segments can be chosen. The

stopping condition blends these two considerations. The algorithm stops when k exceeds

$$c(n, \Delta_n(\mathbf{f}_{k,n})) = \beta n^{1/3} \Delta_n(\mathbf{f}_{k,n})^{-1/2} r, \qquad (10)$$

where β is a parameter of the algorithm which was determined by experiments and was set to the constant value 0.3.

Note that, in a practical sense, the number of segments plays a more important role in determining the computational complexity of the algorithm than in measuring the quality of the approximation. Experiments showed that, due to the data dependent curvature penalty and the constraint that only one vertex is moved at a time, the number of segments can increase even beyond the number of data points without any indication of overfitting. While increasing the number of segments beyond a certain limit offers only marginal improvement in the approximation, it causes the algorithm to slow down considerably. Therefore, in on-line applications where speed has priority over precision, it is reasonable to use a smaller number of segments than indicated by (10) and, if "aesthetic" smoothness is an issue, to fit a spline through the vertices of the curve.

3.5 Computational Complexity

The complexity of the inner loop is dominated by the complexity of the projection step, which is $O(nk)$. Increasing the number of segments one at a time (as described in Section 3.3), the complexity of the algorithm to obtain $\mathbf{f}_{k,n}$ is $O(nk^2)$. Using the stopping condition of Section 3.4, the computational complexity of the algorithm becomes $O(n^{5/3})$. This is slightly better than the $O(n^2)$ complexity of the HS algorithm.

The complexity can be dramatically decreased in certain situations. One possibility is to add more than one vertex at a time. For example, if, instead of adding only one vertex, a new vertex is placed at the midpoint of every segment, then we can reduce the computational complexity for producing $\mathbf{f}_{k,n}$ to $O(nk \log k)$. One can also set k to be a constant if the data size is large since increasing k beyond a certain threshold brings only diminishing returns. Also, k can be naturally set to a constant in certain applications, giving $O(nk)$ computational complexity. These simplifications work well in certain situations, but the original algorithm is more robust.

4 EXPERIMENTAL RESULTS

We have extensively tested the proposed algorithm on two-dimensional data sets. In most experiments, the data was generated by a commonly used (see, e.g., [2], [10], [11]) additive model

$$\mathbf{X} = \mathbf{Y} + \mathbf{e}, \qquad (11)$$

where $\mathbf{Y}$ is uniformly distributed on a smooth planar curve (hereafter called the *generating curve*) and $\mathbf{e}$ is bivariate additive noise which is independent of $\mathbf{Y}$.

In Section 4.1, we compare the polygonal line algorithm, the HS algorithm, and, for closed generating curves, the BR algorithm [5]. The various methods are compared subjectively based mainly on how closely the resulting curve follows the shape of the generating curve. We use varying generating shapes, noise parameters, and data sizes to demonstrate the robustness of the polygonal line algorithm. For the case of a circular generating curve, we also evaluate, in a quantitative manner, how well the polygonal line algorithm approximates the generating curve as the data size grows and as the noise variance decreases.

In Section 4.2, we show two scenarios in which the polygonal line algorithm (along with the HS algorithm) fails to produce meaningful results. In the first, the high number of abrupt changes in the direction of the generating curve causes the algorithm to oversmooth the principal curve, even when the data is concentrated on the generating curve. This is a typical situation when the penalty parameter λ'_p should be decreased. In the second scenario, the generating curve is too complex (e.g., it contains loops, or it has the shape of a spiral), so the algorithm fails to find the global structure of the data if the process is started from the first principal component. To recover the generating curve, one must replace the initialization step by a more sophisticated routine that approximately captures the global structure of the data.

In Section 4.3, an application in feature extraction is briefly outlined. We depart from the synthetic data generating model in (11) and use an extended version of the polygonal line algorithm to find the medial axes ("skeletons") of pixel templates of handwritten characters. Such skeletons can be used in handwritten character recognition and compression of handwritten documents.

4.1 Experiments with the Generating Curve Model

In general, in simulation examples considered by HS, the performance of the new algorithm is comparable with the HS algorithm. Due to the data dependence of the curvature penalty factor and the stopping condition, our algorithm turns out to be more robust to alterations in the data generating model, as well as to changes in the parameters of the particular model.

We use model (11) with varying generating shapes, noise parameters, and data sizes to demonstrate the robustness of the polygonal line algorithm. All plots show the generating curve, the curve produced by our polygonal line algorithm (Polygonal principal curve), and the curve produced by the HS algorithm with spline smoothing (HS principal curve), which we have found to perform better than the HS algorithm using scatterplot smoothing. For closed generating curves, we also include the curve produced by the BR algorithm [5] (BR principal curve), which extends the HS algorithm to closed curves. The two coefficients of the polygonal line algorithm are set in all experiments to the constant values $\beta = 0.3$ and $\lambda'_p = 0.13$.

In Fig. 5a, the generating curve is a circle of radius $r = 1$ and $\mathbf{e} = (e_1, e_2)$ is a zero mean bivariate uncorrelated Gaussian with variance $E(e_i^2) = 0.04$, for $i = 1, 2$. The performance of the three algorithms (HS, BR, and the polygonal line algorithm) is comparable, although the HS algorithm exhibits more bias than the other two. Note that the BR algorithm [5] has been tailored to fit closed curves and to reduce the estimation bias. In Fig. 5b, only half of the circle is used as a generating curve and the other parameters remain the same. Here, too, both the HS and our algorithm behave similarly.

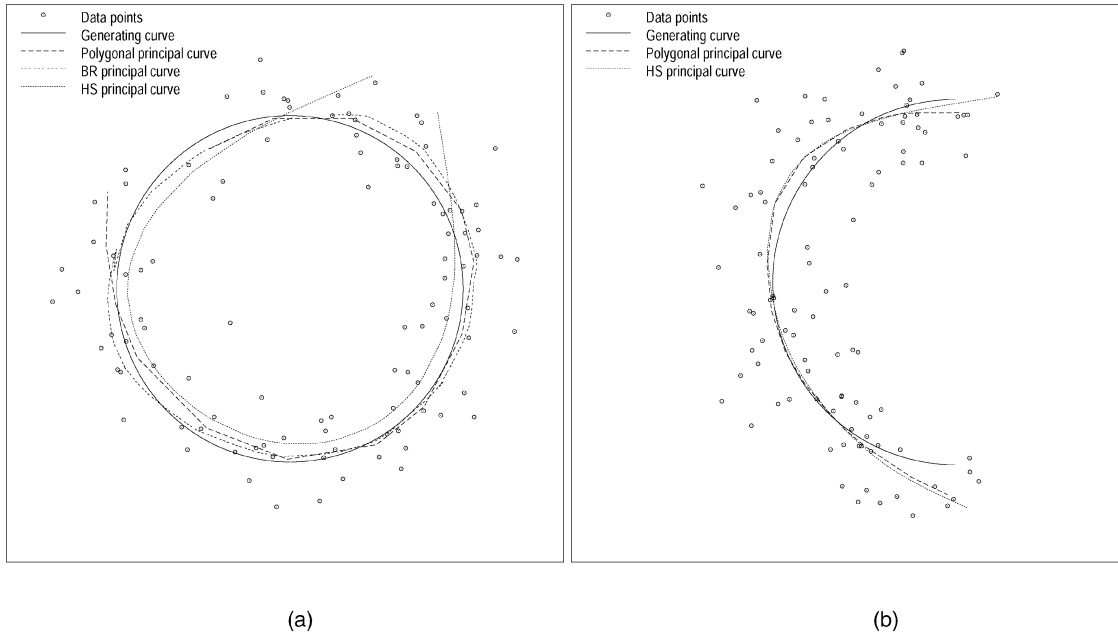When we depart from these usual settings, the polygonal line algorithm exhibits better behavior than the HS

Fig. 5. (a) Circle, 100 data points: The BR and the polynomial line alogorithms show less bias than the HS algorithm. (b) Half circle, 100 data points: The HS and the polygonal line algorithms produce similar curves.
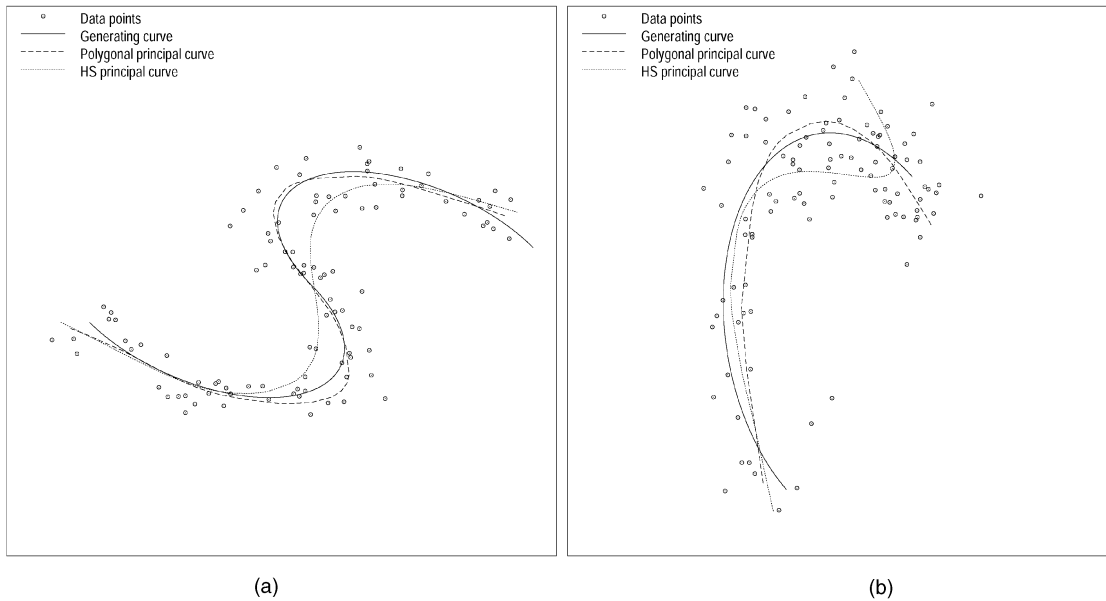

Fig. 6. Transformed Data Sets. The polygonal line algorithm still follows fairly closely the "distorted" shapes. (a) Distorted half circle, 100 data points. (b) Distorted S-shape, 100 data points.

algorithm. In Fig. 6a, the data was generated similarly to the data set of Fig. 5 and, then, it was linearly transformed using the matrix

$$\begin{pmatrix} 0.7 & 0.4 \\ -0.8 & 1.0 \end{pmatrix}.$$

In Fig. 6b, the transformation

$$\begin{pmatrix} -1.0 & -1.2 \\ 1.0 & -0.2 \end{pmatrix}$$

was used. The original data set was generated by an S-shaped generating curve consisting of two half circles of

unit radii to which the same Gaussian noise was added as in Fig. 5. In both cases, the polygonal line algorithm produces curves that fit the generator curve more closely. This is especially noticeable in Fig. 6a, where the HS principal curve fails to follow the shape of the distorted half circle.

There are two situations when we expect our algorithm to perform particularly well. If the distribution is concentrated on a curve, then, according to both the HS and our definitions, the principal curve is the generating curve itself. Thus, if the noise variance is small, we expect both algorithms to very closely approximate the generating curve. The data in Fig. 7a was generated using the same additive Gaussian model as in Fig. 5, but the noise variance
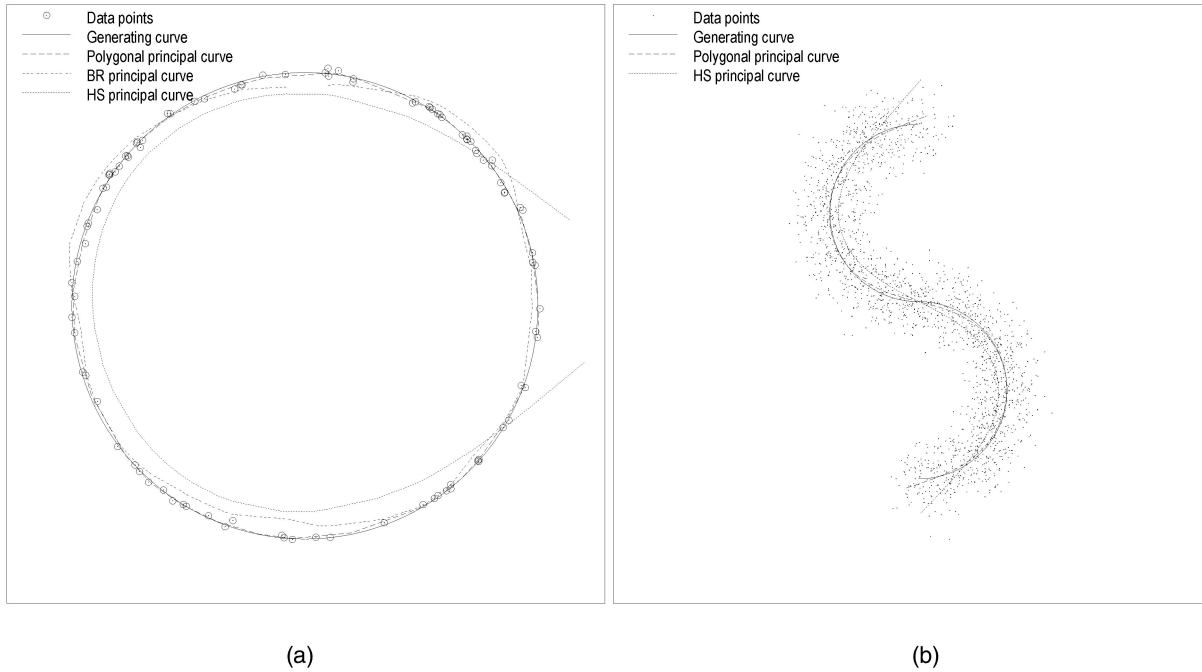
(a)   (b)

Fig. 7. (a) Small noise variance and (b) large sample size. The curves produced by the polygonal line algorithm are nearly indistinguishable from the generating curves.

was reduced to $E(e_i^2) = 0.0001$ for $i = 1, 2$. In this case, we found that the polygonal line algorithm outperformed both the HS and the BR algorithms.

The second case is when the sample size is large. Although the generating curve is not necessarily the principal curve of the distribution, it is natural to expect the algorithm to well approximate the generating curve as the sample size grows. Such a case is shown in Fig. 7b, where $n = 10,000$ data points were generated (but only $2,000$ of these were actually plotted). Here, the polygonal line algorithm approximates the generating curve with much better accuracy than the HS algorithm.

Although in the model (11), the generating curve is, in general, not the principal curve in our definition (or in the HS definition), it is of interest to numerically evaluate how well the polygonal line algorithm approximates the generating curve. In these experiments, the generating curve $\mathbf{g}(t)$ is a circle of unit radius centered at the origin and the noise is zero mean bivariate uncorrelated Gaussian. We chose 21 different data sizes ranging from 10 to $10,000$ and seven different noise standard deviations ranging from $\sigma = 0.01$ to $\sigma = 0.4$. For the measure of approximation, we chose the average distance defined by $\delta = \frac{1}{l(\mathbf{f})} \int \min_s \|\mathbf{f}(t) - \mathbf{g}(s)\| dt$, where the polygonal line $\mathbf{f}$ is parametrized by its arc length. To eliminate the distortion occurring at the endpoints, we initialized the polygonal line algorithm by an equilateral triangle inscribed in the generating circle. For each particular data size and noise variance value, 100 random data sets were generated and the resulting δ values were averaged over these experiments. The dependence of the average distance δ on the data size and the noise variance is plotted on a logarithmic scale in Fig. 8. The resulting curves justify our informal observation made earlier that the approximation substantially improves as the data size grows and as the variance of the noise decreases.

4.2 Failure Modes

We describe two specific situations when the polygonal line algorithm fails to recover the generating curve. In the first scenario, we use zig-zagging generating curves $\mathbf{f}_i$ for $i = 2, 3, 4$, consisting of 2^i line segments of equal length such that two consecutive segments join at a right angle (Fig. 9). In these experiments, the number of the data points generated on a line segment is constant (it is set to 100) and the variance of the bivariate Gaussian noise is $l^2 \cdot 0.0005$, where l is the length of a line segment. Fig. 9 shows the principal curves produced by the HS and the polygonal line algorithms in the three experiments. Although the polygonal principal curve follows the generating curve more closely than the HS principal curve in the first two experiments (Fig. 9a and Fig. 9b), the two algorithms produce equally poor results if the number of line segments exceeds a certain limit (Fig. 9c). The data dependent penalty term explains this behavior of the polygonal line algorithm. Since the penalty factor λ_p is proportional to the number of line segments, the penalty relatively increases as the number of line segments of the generating curve grows. To achieve the same local smoothness in the four experiments, the penalty factor should be gradually decreased as the number of line segments of the generating curve grows. Indeed, if the constant of the penalty term is reset to $\lambda_p' = 0.02$ in the fourth experiment, the polygonal principal curve recovers the generating curve with high accuracy (Fig. 11a).

The second scenario when the polygonal line algorithm fails to produce a meaningful result is when the generating curve is too complex, so the algorithm does not find the global structure of the data. To test the gradual degradation of the algorithm, we used spiral-shaped generating curves of increasing length, i.e., we set $\mathbf{g}_i(t) = (t \sin(i\pi t), t \cos(i\pi t))$ for $t \in [0, 1]$ and $i = 1, \ldots, 6$. The variance of the noise was
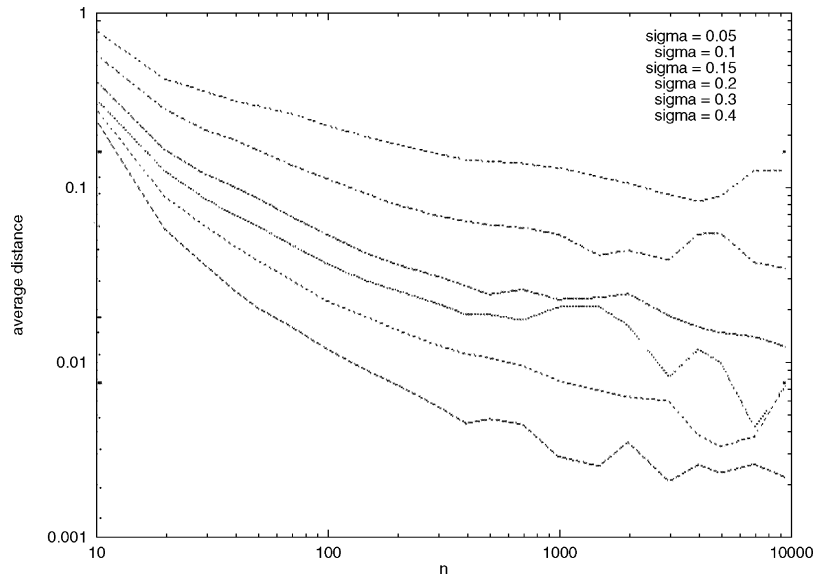
Fig. 8. The approximation error δ decreases as n grows or σ decreases.

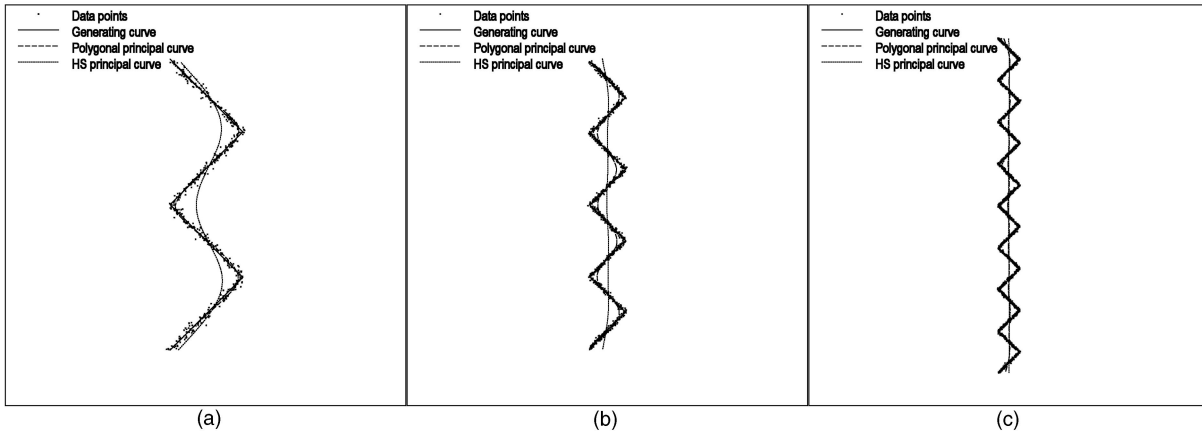(a)   (b)   (c)

Fig. 9. Abrupt changes in the direction of the generating curve. The polygonal line algorithm oversmoothes the principal curve as the number of direction changes increases.

set to 0.0001 and we generated $1,000$ data points in each experiment. Fig. 10 shows the principal curves produced by the HS and the polygonal line algorithms in three experiments ($i = 3, 4, 6$). In the first two experiments (Fig. 10a and Fig. 10b), the polygonal principal curve is almost indistinguishable from the generating curve, while the HS algorithm either oversmoothes the principal curve (Fig. 10a) or fails to recover the shape of the generating curve (Fig. 10b). In the third experiment, both algorithms fail to find the shape of the generating curve (Fig. 10c). The failure here is due to the fact that the algorithm is stuck in a local minima between the initial curve (the first principal component) and the desired solution (the generating curve). If this is likely to occur in an application, the initialization step must be replaced by a more sophisticated routine that approximately captures the global structure of the data. Fig. 11b indicates that this indeed works. Here, we manually initialize both algorithms by a polygonal line with eight vertices. Using this "hint," the polygonal line algorithm produces an almost perfect solution, while the HS algorithm still cannot recover the shape of the generating curve.

4.3 Recovering Smooth Character Skeletons

In this section, we use the polygonal line algorithm to find smooth skeletons of handwritten character templates. The results reported here are preliminary and the full treatment of this application is presented in [19]. Principal curves have been applied by Singh et al. [6] for similar purposes. In [6], the initial trees are produced using a version of the SOM algorithm and then the HS algorithm is applied to extract skeletal structures of handwritten characters. The focus in [6] was on recovering the topological structure of noisy letters in faded documents. Our aim with the polygonal line algorithm is to produce smooth curves which can be used to recover the trajectory of the penstroke.

To transform black-and-white character templates into two-dimensional data sets, we place the midpoint of the bottom-most left-most pixel of the template at the center of a coordinate system. The unit length of the coordinate system is set to the width (and height) of a pixel, so the midpoint of each pixel has integer coordinates. Then, we add the midpoint of each black pixel to the data set.

The polygonal line algorithm was tested on images of isolated handwritten digits from the NIST Special Database 19 [22]. We found that the polygonal line
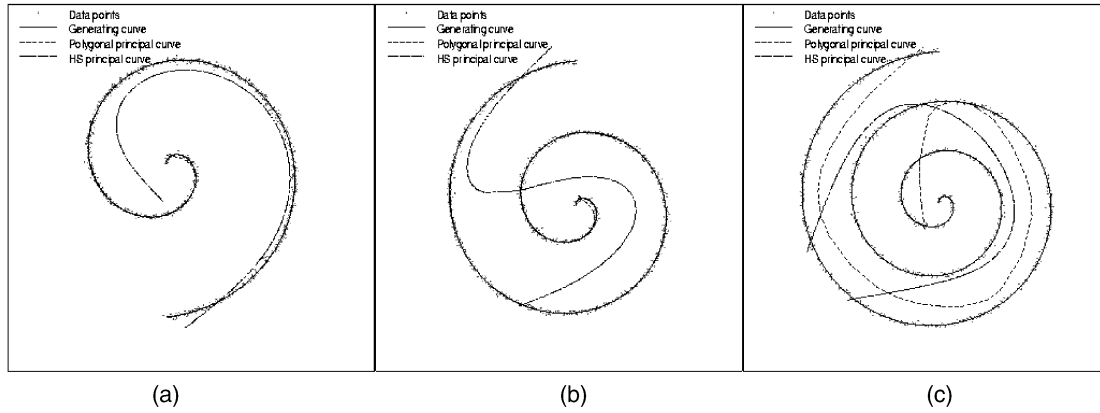
Fig. 10. Spiral-shaped generating curves. The polygonal line algorithm fails to find the generating curve as the length of the spiral is increased.
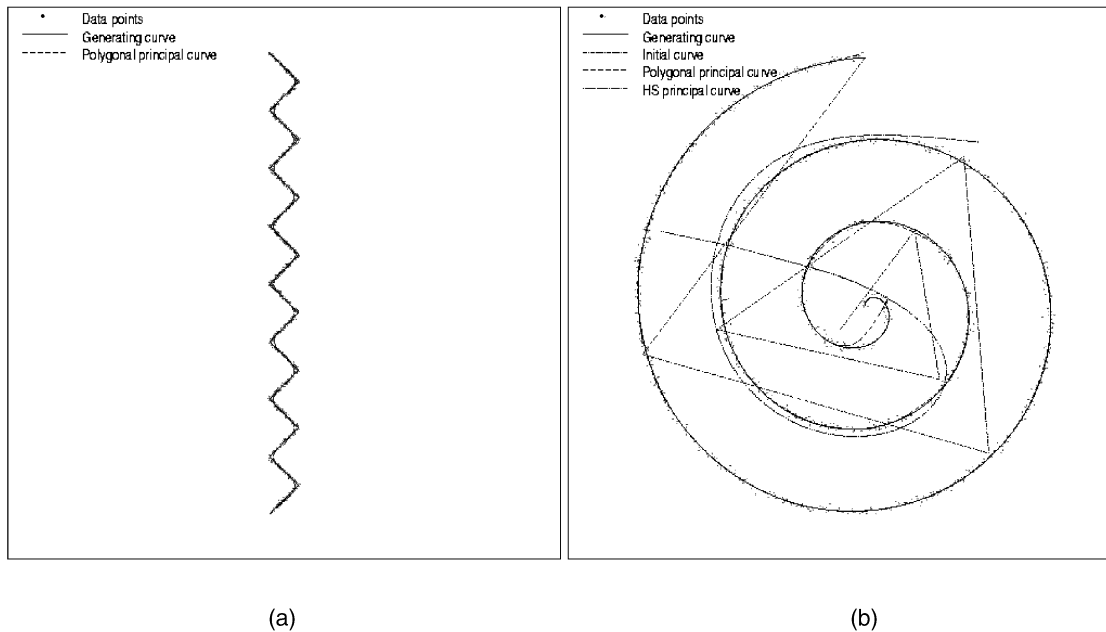

Fig. 11. Improved performance of the polygonal line algorithm. (a) The penalty parameter is decreased. (b) The algorithms are initialized manually.

algorithm can be used effectively to find smooth medial axes of simple digits which contain no loops or crossings of strokes. Fig. 12 shows some of these results.

To find smooth skeletons of more complex characters, we modified and extended the polygonal line algorithm. We introduced new types of vertices incident to more than two line segments to handle loops and crossings, and modified the vertex optimization step accordingly. The initialization step was also replaced by a more sophisticated routine based on a thinning method [23] to produce an initial graph that approximately captures the topology of the character. Fig. 13 shows some of the results of the extended polygonal line algorithm on more complex characters. Details of the extended polygonal line algorithm and more complete testing results will be presented in the future.

5   CONCLUSION

A new definition of principal curves has been offered. The new definition has significant theoretical appeal; the existence of principal curves with this definition can be proven under very general conditions and a learning method for constructing principal curves for finite data sets yields to theoretical analysis.

Inspired by the new definition and the theoretical learning scheme, we have introduced a new practical polygonal line algorithm for designing principal curves. Lacking theoretical results concerning both the HS and our polygonal line algorithm, we compared the two methods through simulations. We have found that, in general, our algorithm has either comparable or better performance than the original HS algorithm and it exhibits better, more robust behavior when the data generating model is varied. We have also reported preliminary results in applying the polygonal line algorithm to the problem of handwritten character skeletonization. We believe that the new principal curve algorithm may also prove useful in other applications such as data compression and feature extraction, where a compact and accurate description of a pattern or an image is required. These are issues for future work.
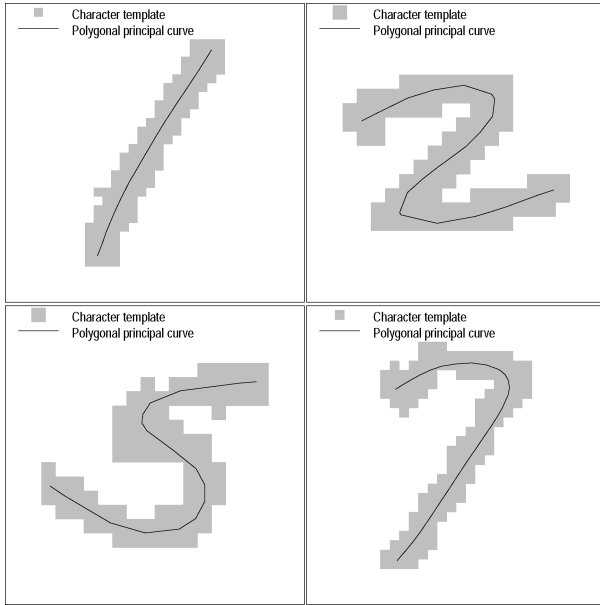
Fig. 12. Results produced by the polygonal line algorithm on characters not containing loops or crossings.

Fig. 13. Results produced by the extended polygonal line algorithm on characters containing loops or crossings.

APPENDIX A

Curves in $\mathbb{R}^d$

Let $\mathbf{f} : [a, b] \to \mathbb{R}^d$ a continuous mapping (curve). The *length* of $\mathbf{f}$ over an interval $[\alpha, \beta] \subset [a, b]$, denoted by $l(\mathbf{f}, \alpha, \beta)$, is defined by

$$l(\mathbf{f}, \alpha, \beta) = \sup \sum_{i=1}^{N} \|\mathbf{f}(t_i) - \mathbf{f}(t_{i-1})\|, \qquad (A.1)$$

where the supremum is taken over all finite partitions of $[\alpha, \beta]$ with arbitrary subdivision points

$$\alpha = t_0 \le t_1 < \cdots \le t_N = \beta$$

for $N \ge 1$. The length of $\mathbf{f}$ over its domain $[a, b]$ is denoted by $l(\mathbf{f})$. If $l(\mathbf{f}) < \infty$, then $\mathbf{f}$ is said to be *rectifiable*. It is well-known that $\mathbf{f} = (f_1, \ldots, f_d)$ is rectifiable if and only if each coordinate function $f_j : [a, b] \to \mathbb{R}$ is of bounded variation.

Two curves $\mathbf{f} : [a, b] \to \mathbb{R}^d$ and $\mathbf{g} : [a', b'] \to \mathbb{R}^d$ are said to be *equivalent* if there exist two nondecreasing continuous real onto functions $\phi : [0, 1] \to [a, b]$ and $\eta : [0, 1] \to [a', b']$ such that

$$\mathbf{f}(\phi(t)) = \mathbf{g}(\eta(t)), \quad \text{for } t \in [0, 1].$$

In this case, we write $\mathbf{f} \sim \mathbf{g}$ and it is easy to see that $\sim$ is an equivalence relation. If $\mathbf{f} \sim \mathbf{g}$, then $l(\mathbf{f}) = l(\mathbf{g})$. A curve $\mathbf{g}$ over $[a, b]$ is said to be *parametrized by its arc length* if $l(\mathbf{g}, a, t) = t - a$ for any $a \le t \le b$. Let $\mathbf{f}$ be a curve over $[a, b]$ with length L. It is not hard to see that there exists a unique arc length parametrized curve $\mathbf{g}$ over $[0, L]$ such that $\mathbf{f} \sim \mathbf{g}$.

Let $\mathbf{f}$ be any curve with length $L' \le L$, and consider the arc length parametrized curve $\mathbf{g} \sim \mathbf{f}$ with parameter interval $[0, L']$. By (A.1), for all $s_1, s_2 \in [0, L']$, we have
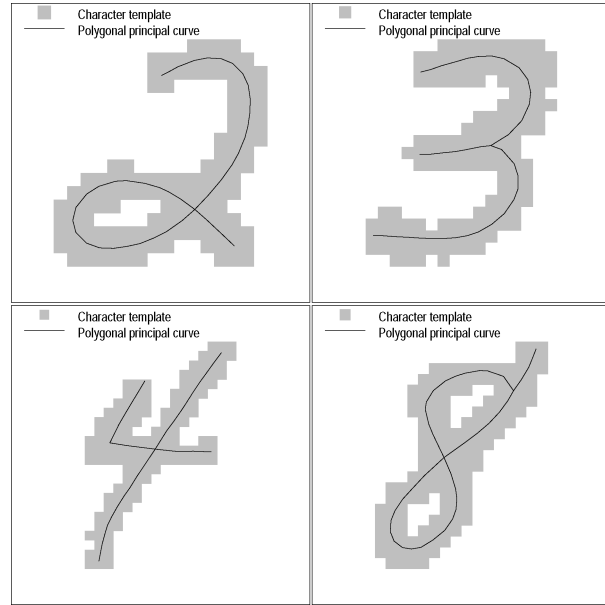
$$\|\mathbf{g}(s_1) - \mathbf{g}(s_2)\| \le |s_1 - s_2|.$$

Define $\hat{\mathbf{g}}(t) = \mathbf{g}(L't)$ for $0 \le t \le 1$. Then, $\mathbf{f} \sim \hat{\mathbf{g}}$, and $\hat{\mathbf{g}}$ satisfies the following Lipschitz condition: For all $t_1, t_2 \in [0, 1]$,

$$\begin{aligned}
\|\hat{\mathbf{g}}(t_1) - \hat{\mathbf{g}}(t_2)\| &= \|\mathbf{g}(L't_1) - \mathbf{g}(L't_2)\| \\
&\le L'|t_1 - t_2| \qquad (A.2) \\
&\le L|t_1 - t_2|.
\end{aligned}$$

On the other hand, note that if $\hat{\mathbf{g}}$ is a curve over $[0, 1]$ which satisfies the Lipschitz condition (A.2), then its length is at most L.

Let $\mathbf{f}$ be a curve over $[a, b]$ and denote the squared Euclidean distance from any $\mathbf{x} \in \mathbb{R}^d$ to $\mathbf{f}$ by

$$\Delta(\mathbf{x}, \mathbf{f}) = \inf_{a \le t \le b} \|\mathbf{x} - \mathbf{f}(t)\|^2.$$

Note that if $l(\mathbf{f}) < \infty$, then, by the continuity of $\mathbf{f}$, its graph

$$G_{\mathbf{f}} = \{\mathbf{f}(t) : a \le t \le b\}$$

is a compact subset of $\mathbb{R}^d$ and the infimum above is achieved for some t. Also, since $G_{\mathbf{f}} = G_{\mathbf{g}}$ if $\mathbf{f} \sim \mathbf{g}$, we also have that $\Delta(\mathbf{x}, \mathbf{f}) = \Delta(\mathbf{x}, \mathbf{g})$ for all $\mathbf{g} \sim \mathbf{f}$.

Proof of Lemma 1. Define

$$\Delta^* = \inf\{\Delta(\mathbf{f}) : l(\mathbf{f}) \le L\}.$$

First, we show that the above infimum does not change if we add the restriction that all $\mathbf{f}$ lie inside a closed sphere $S(r) = \{\mathbf{x} : \|\mathbf{x}\| \le r\}$ of large enough radius r and centered at the origin. Indeed, without excluding nontrivial cases, we can assume that $\Delta^* < E\|\mathbf{X}\|^2$. Denote the distribution of $\mathbf{X}$ by μ and choose $r > 3L$ large enough such that

$$\int_{S(r/3)} \|\mathbf{x}\|^2 \mu(d\mathbf{x}) > \Delta^* + \varepsilon \qquad (A.3)$$

for some $\varepsilon > 0$. If $\mathbf{f}$ is such that $G_{\mathbf{f}}$ is not entirely contained in $S(r)$, then, for all $\mathbf{x} \in S(r/3)$, we have

$\Delta(\mathbf{x}, \mathbf{f}) > \|\mathbf{x}\|^2$ since the diameter of $G_\mathbf{f}$ is at most L. Then, (A.3) implies that

$$\Delta(\mathbf{f}) \geq \int_{S(r/3)} \Delta(\mathbf{x}, \mathbf{f}) \mu(d\mathbf{x}) > \Delta^* + \varepsilon$$

and, thus,

$$\Delta^* = \inf\{\Delta(\mathbf{f}) : l(\mathbf{f}) \leq L, G_\mathbf{f} \subset S(r)\}. \qquad \text{(A.4)}$$

In view of (A.4), there exists a sequence of curves $\{\mathbf{f}_n\}$ such that $l(\mathbf{f}_n) \leq L$, $G_{\mathbf{f}_n} \subset S(r)$ for all n, and $\Delta(\mathbf{f}_n) \to \Delta^*$. By the discussion preceding (A.2), we can assume without loss of generality that all $\mathbf{f}_n$ are defined over $[0, 1]$ and

$$\|\mathbf{f}_n(t_1) - \mathbf{f}_n(t_2)\| \leq L|t_1 - t_2| \qquad \text{(A.5)}$$

for all $t_1, t_2 \in [0, 1]$. Consider the set of all curves $\mathcal{C}$ over $[0, 1]$ such that $\mathbf{f} \in \mathcal{C}$ if and only if $\|\mathbf{f}(t_1) - \mathbf{f}(t_2)\| \leq L|t_1 - t_2|$ for all $t_1, t_2 \in [0, 1]$ and $G_\mathbf{f} \subset S(r)$. It is easy to see that $\mathcal{C}$ is a closed set under the uniform metric $d(\mathbf{f}, \mathbf{g}) = \sup_{0 \leq t \leq 1} \|\mathbf{f}(t) - \mathbf{g}(t)\|$. Also, $\mathcal{C}$ is an equicontinuous family of functions and $\sup_t \|\mathbf{f}(t)\|$ is uniformly bounded over $\mathcal{C}$. Thus, $\mathcal{C}$ is a compact metric space by the Arzela-Ascoli theorem (see, e.g., [24]). Since $\mathbf{f}_n \in \mathcal{C}$ for all n, it follows that there exists a subsequence $\mathbf{f}_{n_k}$ converging uniformly to an $\mathbf{f}^* \in \mathcal{C}$.

To simplify the notation, let us rename $\{\mathbf{f}_{n_k}\}$ as $\{\mathbf{f}_n\}$. Fix $\mathbf{x} \in \mathbb{R}^d$, assume $\Delta(\mathbf{x}, \mathbf{f}_n) \geq \Delta(\mathbf{x}, \mathbf{f}^*)$, and let $t_\mathbf{x}$ be such that $\Delta(\mathbf{x}, \mathbf{f}^*) = \|\mathbf{x} - \mathbf{f}^*(t_\mathbf{x})\|^2$. Then by the triangle inequality,

$$\begin{aligned} |\Delta(\mathbf{x}, \mathbf{f}^*) - \Delta(\mathbf{x}, \mathbf{f}_n)| =& \Delta(\mathbf{x}, \mathbf{f}_n) - \Delta(\mathbf{x}, \mathbf{f}^*) \\ \leq& \|\mathbf{x} - \mathbf{f}_n(t_\mathbf{x})\|^2 - \|\mathbf{x} - \mathbf{f}^*(t_\mathbf{x})\|^2 \\ \leq& (\|\mathbf{x} - \mathbf{f}_n(t_\mathbf{x})\| + \|\mathbf{x} - \mathbf{f}^*(t_\mathbf{x})\|) \\ & \cdot \|\mathbf{f}_n(t_\mathbf{x}) - \mathbf{f}^*(t_\mathbf{x})\|. \end{aligned}$$

By symmetry, a similar inequality holds if $\Delta(\mathbf{x}, \mathbf{f}_n) < \Delta(\mathbf{x}, \mathbf{f}^*)$. Since $G_{\mathbf{f}^*}, G_{\mathbf{f}_n} \subset S(r)$, and $E\|\mathbf{X}\|^2$ is finite, there exists $A > 0$ such that

$$E|\Delta(\mathbf{X}, \mathbf{f}_n) - \Delta(\mathbf{X}, \mathbf{f}^*)| \leq A \sup_{0 \leq t \leq 0} \|\mathbf{f}_n(t) - \mathbf{f}^*(t)\|$$

and, therefore,

$$\Delta^* = \lim_{n \to \infty} \Delta(\mathbf{f}_n) = \Delta(\mathbf{f}^*).$$

Since the Lipschitz condition on $\mathbf{f}^*$ guarantees that $l(\mathbf{f}^*) \leq L$, the proof is complete. □

APPENDIX B

Proof of Theorem 1. Let $\mathbf{f}_k^*$ denote the curve in $\mathcal{S}_k$ minimizing the squared loss, i.e.,

$$\mathbf{f}_k^* = \arg\min_{\mathbf{f} \in \mathcal{S}_k} \Delta(\mathbf{f}).$$

The existence of a minimizing $\mathbf{f}_k^*$ can easily be shown using a simpler version of the proof of Lemma 1. Then, $J(\mathbf{f}_{k,n})$ can be decomposed as

$$J(\mathbf{f}_{k,n}) = (\Delta(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}_k^*)) + (\Delta(\mathbf{f}_k^*) - \Delta(\mathbf{f}^*)),$$

where, using standard terminology, $\Delta(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}_k^*)$ is called the *estimation error* and $\Delta(\mathbf{f}_k^*) - \Delta(\mathbf{f}^*)$ is called the *approximation error*. We consider these terms separately first and then choose k as a function of the training data size n to balance the obtained upper bounds in an asymptotically optimal way.

Approximation Error. For any two curves $\mathbf{f}$ and $\mathbf{g}$ of finite length, define their (nonsymmetric) distance by

$$\rho(\mathbf{f}, \mathbf{g}) = \max_t \min_s \|\mathbf{f}(t) - \mathbf{g}(s)\|.$$

Note that $\rho(\hat{\mathbf{f}}, \hat{\mathbf{g}}) = \rho(\mathbf{f}, \mathbf{g})$ if $\hat{\mathbf{f}} \sim \mathbf{f}$ and $\hat{\mathbf{g}} \sim \mathbf{g}$, i.e., $\rho(\mathbf{f}, \mathbf{g})$ is independent of the particular choice of the parametrization within equivalence classes. Next, we observe that if the diameter of K is D, and $G_\mathbf{f}, G_\mathbf{g} \in K$, then, for all $\mathbf{x} \in K$,

$$\Delta(\mathbf{x}, \mathbf{g}) - \Delta(\mathbf{x}, \mathbf{f}) \leq 2D\rho(\mathbf{f}, \mathbf{g}) \qquad \text{(B.1)}$$

and, therefore,

$$\Delta(\mathbf{g}) - \Delta(\mathbf{f}) \leq 2D\rho(\mathbf{f}, \mathbf{g}). \qquad \text{(B.2)}$$

To prove (B.1), let $\mathbf{x} \in K$ and choose t' and s' such that $\Delta(\mathbf{x}, \mathbf{f}) = \|\mathbf{x} - \mathbf{f}(t')\|^2$ and

$$\min_s \|\mathbf{g}(s) - \mathbf{f}(t')\| = \|\mathbf{g}(s') - \mathbf{f}(t')\|.$$

Then,

$$\begin{aligned} \Delta(\mathbf{x}, \mathbf{g}) - \Delta(\mathbf{x}, \mathbf{f}) \leq& \|\mathbf{x} - \mathbf{g}(s')\|^2 - \|\mathbf{x} - \mathbf{f}(t')\|^2 \\ =& (\|\mathbf{x} - \mathbf{g}(s')\| + \|\mathbf{x} - \mathbf{f}(t')\|) \\ & \cdot (\|\mathbf{x} - \mathbf{g}(s')\| - \|\mathbf{x} - \mathbf{f}(t')\|) \\ \leq& 2D\|\mathbf{g}(s') - \mathbf{f}(t')\| \\ \leq& 2D\rho(\mathbf{f}, \mathbf{g}). \end{aligned}$$

Let $\mathbf{f} \in \mathcal{S}$ be an arbitrary arc length parametrized curve over $[0, L']$, where $L' \leq L$. Define $\mathbf{g}$ as a polygonal curve with vertices $\mathbf{f}(0), \mathbf{f}(L'/k), \ldots, \mathbf{f}((k-1)L'/k), \mathbf{f}(L')$. For any $t \in [0, L']$, we have $|t - iL'/k| \leq L/(2k)$ for some $i \in \{0, \ldots, k\}$. Since $\mathbf{g}(s) = \mathbf{f}(iL'/k)$ for some s, we have

$$\min_s \|\mathbf{f}(t) - \mathbf{g}(s)\| \leq \|\mathbf{f}(t) - \mathbf{f}(iL'/k)\|$$

$$\leq |t - iL'/k| \leq \frac{L}{2k}.$$

Note that $l(\mathbf{g}) \leq L'$, by construction, and, thus, $\mathbf{g} \in \mathcal{S}_k$. Thus, for every $\mathbf{f} \in \mathcal{S}$, there exists a $\mathbf{g} \in \mathcal{S}_k$ such that $\rho(\mathbf{f}, \mathbf{g}) \leq L/(2k)$. Now, let $\mathbf{g} \in \mathcal{S}_k$ be such that $\rho(\mathbf{f}^*, \mathbf{g}) \leq L/(2k)$. Then, by (B.2), we conclude that the approximation error is upper bounded as

$$\begin{aligned} \Delta(\mathbf{f}_k^*) - \Delta(\mathbf{f}^*) \leq& \Delta(\mathbf{g}) - \Delta(\mathbf{f}^*) \\ \leq& 2D\rho(\mathbf{f}^*, \mathbf{g}) \\ \leq& \frac{DL}{k}. \end{aligned} \qquad \text{(B.3)}$$

Estimation Error. For each $\varepsilon > 0$ and $k \geq 1$ let $S_{k,\varepsilon}$ be a *finite* set of curves in K which form an ε-cover of $\mathcal{S}_k$ in the following sense: For any $\mathbf{f} \in \mathcal{S}_k$, there is an $\mathbf{f}' \in \mathcal{S}_{k,\varepsilon}$ which satisfies

$$\sup_{\mathbf{x}\in K} |\Delta(\mathbf{x},\mathbf{f}) - \Delta(\mathbf{x},\mathbf{f}')| \leq \varepsilon. \tag{B.4}$$

The explicit construction of $S_{k,\varepsilon}$ is given in [19]. Since $\mathbf{f}_{k,n} \in \mathcal{S}_k$ (see (5)), there exists an $\mathbf{f}'_{k,n} \in \mathcal{S}_{k,\varepsilon}$ such that $|\Delta(\mathbf{x},\mathbf{f}_{k,n}) - \Delta(\mathbf{x},\mathbf{f}'_{k,n})| \leq \varepsilon$ for all $\mathbf{x} \in K$. We introduce the compact notation $\mathcal{X}_n = (\mathbf{X}_1,\ldots,\mathbf{X}_n)$ for the training data. Thus, we can write

$$E[\Delta(\mathbf{X},\mathbf{f}_{k,n})|\mathcal{X}_n] - \Delta(\mathbf{f}_k^*) =$$
$$E[\Delta(\mathbf{X},\mathbf{f}_{k,n})|\mathcal{X}_n] - \Delta_n(\mathbf{f}_{k,n}) + \Delta_n(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}_k^*)$$
$$\leq 2\varepsilon + E[\Delta(\mathbf{X},\mathbf{f}'_{k,n})|\mathcal{X}_n] - \Delta_n(\mathbf{f}'_{k,n}) + \Delta_n(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}_k^*)$$
$$\tag{B.5}$$

$$\leq 2\varepsilon + E[\Delta(\mathbf{X},\mathbf{f}'_{k,n})|\mathcal{X}_n] - \Delta_n(\mathbf{f}'_{k,n}) + \Delta_n(\mathbf{f}_k^*) - \Delta(\mathbf{f}_k^*) \tag{B.6}$$

$$\leq 2\varepsilon + 2 \cdot \max_{\mathbf{f}\in\mathcal{S}_{k,\varepsilon}\cup\{\mathbf{f}^*\}} |\Delta(\mathbf{f}) - \Delta_n(\mathbf{f})|, \tag{B.7}$$

where (B.5) follows from the approximating property of $\mathbf{f}'_{k,n}$ and the fact that the distribution of $\mathbf{X}$ is concentrated on K. Equation (B.6) holds because $\mathbf{f}_{k,n}$ minimizes $\Delta_n(\mathbf{f})$ over all $\mathbf{f} \in \mathcal{S}_k$, and (B.7) follows because, given $\mathcal{X}_n = (\mathbf{X}_1,\ldots,\mathbf{X}_n)$, $E[\Delta(\mathbf{X},\mathbf{f}'_{k,n})|\mathcal{X}_n]$ is an ordinary expectation of the type $E[\Delta(\mathbf{X},\mathbf{f})]$, $\mathbf{f} \in \mathcal{S}_{k,\varepsilon}$. Thus, for any $t > 2\varepsilon$, the union bound implies

$$P\{E[\Delta(\mathbf{X},\mathbf{f}_{k,n})|\mathcal{X}_n] - \Delta(\mathbf{f}_k^*) > t\}$$
$$\leq P\Big\{\max_{\mathbf{f}\in\mathcal{S}_{k,\varepsilon}\cup\{\mathbf{f}^*\}} |\Delta(\mathbf{f}) - \Delta_n(\mathbf{f})| > \frac{t}{2} - \varepsilon\Big\} \tag{B.8}$$
$$\leq (|S_{k,\varepsilon}| + 1) \max_{\mathbf{f}\in\mathcal{S}_{k,\varepsilon}\cup\{\mathbf{f}^*\}} P\{|\Delta(\mathbf{f}) - \Delta_n(\mathbf{f})| > \frac{t}{2} - \varepsilon\},$$

where $|\mathcal{S}_{k,\varepsilon}|$ denotes the cardinality of $\mathcal{S}_{k,\varepsilon}$.

Recall now Hoeffding's inequality [25] which states that if $Y_1, Y_2, \ldots, Y_n$ are independent and identically distributed real random variables such that $0 \leq Y_i \leq A$ with probability one, then, for all $u > 0$,

$$P\left\{\left|\frac{1}{n}\sum_{i=1}^n Y_i - EY_1\right| > u\right\} \leq 2e^{-2nu^2/A^2}.$$

Since the diameter of K is D, we have $\|\mathbf{x} - \mathbf{f}(t)\|^2 \leq D^2$ for all $\mathbf{x} \in K$ and $\mathbf{f}$ such that $G_\mathbf{f} \in K$. Thus, $0 \leq \Delta(\mathbf{X},\mathbf{f}) \leq D^2$ with probability one and by Hoeffding's inequality, for all $\mathbf{f} \in \mathcal{S}_{k,\varepsilon} \cup \{\mathbf{f}^*\}$, we have

$$P\Big\{|\Delta(\mathbf{f}) - \Delta_n(\mathbf{f})| > \frac{t}{2} - \varepsilon\Big\} \leq 2e^{-2n((t/2)-\varepsilon)^2/D^4},$$

which implies, by (B.8), that

$$P\{E[\Delta(\mathbf{X},\mathbf{f}_{k,n})|\mathcal{X}_n] - \Delta(\mathbf{f}_k^*) > t\}$$
$$\leq 2(|S_{k,\varepsilon}| + 1)e^{-2n((t/2)-\varepsilon)^2/D^4} \tag{B.9}$$

for any $t > 2\varepsilon$. Using the fact that $E[Y] = \int_0^\infty P\{Y > t\}\,dt$ for any nonnegative random variable Y, we can write for any $u > 0$,

$$\Delta(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}_k^*) \leq \int_0^\infty P\{E[\Delta(\mathbf{X},\mathbf{f}_{k,n})|\mathcal{X}_n] - \Delta(\mathbf{f}_k^*) > t\}\,dt$$
$$\leq u + 2\varepsilon + 2(|S_{k,\varepsilon}| + 1)$$
$$\cdot \int_{u+2\varepsilon}^\infty e^{-2n((t/2)-\varepsilon)^2/D^4}\,dt \tag{B.10}$$
$$\leq u + 2\varepsilon + 2(|S_{k,\varepsilon}| + 1)D^4 \cdot \frac{e^{-nu^2/(2D^4)}}{nu} \tag{B.11}$$
$$\leq \sqrt{\frac{2D^4 \log(|S_{k,\varepsilon}| + 1)}{n}} + 2\varepsilon + O(n^{-1/2}), \tag{B.12}$$

where (B.11) follows from the inequality

$$\int_x^\infty e^{-t^2/2}\,dt < (1/x)e^{-x^2/2},$$

for $x > 0$, and (B.12) follows by setting

$$u = \sqrt{\frac{2D^4 \log(|S_{k,\varepsilon}| + 1)}{n}},$$

where log denotes natural logarithm. The following lemma, which is proven in [19], demonstrates the existence of a suitable covering set $S_{k,\varepsilon}$.

Lemma 2. *For any $\varepsilon > 0$, there exists a finite collection of curves $S_{k,\varepsilon}$ in K such that*

$$\sup_{\mathbf{x}\in K} |\Delta(\mathbf{x},\mathbf{f}) - \Delta(\mathbf{x},\mathbf{f}')| \leq \varepsilon$$

and

$$|S_{k,\varepsilon}| \leq 2^{\frac{LD}{\varepsilon}+3k+1} V_d^{k+1}\left(\frac{D^2\sqrt{d}}{\varepsilon} + \sqrt{d}\right)^d \left(\frac{LD\sqrt{d}}{k\varepsilon} + 3\sqrt{d}\right)^{kd},$$

where V_d is the volume of the d-dimensional unit sphere and D is the diameter of K.

It is not hard to see that setting $\varepsilon = 1/k$ in Lemma 2 gives the upper bound

$$2D^4 \log(|S_{k,\varepsilon}| + 1) \leq kC(L,D,d), \tag{B.13}$$

where $C(L,D,d)$ does not depend on k. Combining this with (B.12) and the approximation bound given by (B.3) results in

$$\Delta(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}^*) \leq \sqrt{\frac{kC(L,D,d)}{n}} + \frac{DL+2}{k} + O(n^{-1/2}).$$

The rate at which $\Delta(\mathbf{f}_{k,n})$ approaches $\Delta(\mathbf{f}^*)$ is optimized by setting the number of segments k to be proportional to $n^{1/3}$. With this choice, $J(\mathbf{f}_{k,n}) = \Delta(\mathbf{f}_{k,n}) - \Delta(\mathbf{f}^*)$ has the asymptotic convergence rate

$$J(\mathbf{f}_{k,n}) = O(n^{-1/3}),$$

and the proof of Theorem 1 is complete.

To show the bound (7), let $\delta \in (0,1)$ and observe that, by (B.9), we have

$$P\{E[\Delta(\mathbf{X},\mathbf{f}_{k,n})|\mathcal{X}_n] - \Delta(\mathbf{f}_k^*) \leq t\} > 1 - \delta$$

whenever $t > 2\varepsilon$ and

$$\delta = 2\big(|S_{k,\varepsilon}| + 1\big)e^{-2n((t/2)-\varepsilon)^2/D^4}.$$

Solving this equation for t and letting $\varepsilon = 1/k$ as before, we obtain

$$t = \sqrt{\frac{2D^4 \log\big(|S_{k,1/k}| + 1\big) - 2D^4 \log(\delta/2)}{n}} + \frac{2}{k}$$

$$\leq \sqrt{\frac{kC(L, D, d) - 2D^4 \log(\delta/2)}{n}} + \frac{2}{k}.$$

Therefore, with probability at least $1 - \delta$, we have

$$E[\Delta(\mathbf{X}, \mathbf{f}_{k,n})|\mathcal{X}_n] - \Delta(\mathbf{f}_k^*)$$

$$\leq \sqrt{\frac{kC(L, D, d) - 2D^4 \log(\delta/2)}{n}} + \frac{2}{k}.$$

Combining this bound with the approximation bound $\Delta(\mathbf{f}_k^*) - \Delta(\mathbf{f}^*) \leq (DL)/k$ gives (7). $\square$

ACKNOWLEDGMENTS

REFERENCES

[1] T. Hastie, "Principal Curves and Surfaces," PhD thesis, Stanford Univ., 1984.

[2] T. Hastie and W. Stuetzle, "Principal Curves," *J. Am. Statistical Assoc.,* vol. 84, pp. 502-516, 1989.

[3] Y. Linde, A. Buzo, and R.M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. Comm.,* vol. 28, pp. 84-95, 1980.

[4] W.S. Cleveland, "Robust Locally Weighted Regression and Smoothing Scatterplots," *J. Am. Statistical Assoc.,* vol. 74, pp. 829-835, 1979.

[5] J.D. Banfield and A.E. Raftery, "Ice Floe Identification in Satellite Images Using Mathematical Morphology and Clustering about Principal Curves," *J. Am. Statistical Assoc.,* vol. 87, pp. 7-16, 1992.

[6] R. Singh, M.C. Wade, and N.P. Papanikolopoulos, "Letter-Level Shape Description by Skeletonization in Faded Documents," *Proc. Fourth IEEE Workshop Applications of Computer Vision,* pp. 121-126, 1998.

[7] K. Reinhard and M. Niranjan, "Subspace Models for Speech Transitions Using Principal Curves," *Proc. Inst. of Acoustics,* vol. 20, no. 6, pp. 53-60, 1998.

[8] K. Chang and J. Ghosh, "Principal Curves for Nonlinear Feature Extraction and Classification," *Applications of Artificial Neural Networks in Image Processing III,* vol. 3307, pp. 120-129, 1998.

[9] K. Chang and J. Ghosh, "Principal Curve Classifier—A Nonlinear Approach to Pattern Classification," *Proc. IEEE Int'l Joint Conf. Neural Networks,* pp. 695-700, 1998.

[10] R. Tibshirani, "Principal Curves Revisited," *Statistics and Computation,* vol. 2, pp. 183-190, 1992.

[11] F. Mulier and V. Cherkassky, "Self-Organization as An Iterative Kernel Smoothing Process," *Neural Computation,* vol. 7, pp. 1,165-1,177, 1995.

[12] P. Delicado, "Principal Curves and Principal Oriented Points," Technical Report 309, Dept. d'Economia i Empresa, Universitat Pompeu Fabra, 1998.

[13] T. Duchamp and W. Stuetzle, "Geometric Properties of Principal Curves in the Plane," *Robust Statistics, Data Analysis, and Computer Intensive Methods: In Honor of Peter Huber's 60th Birthday,* H. Rieder, ed., vol. 109, pp. 135-152, Springer-Verlag, 1996.

[14] A.N. Kolmogorov and S.V. Fomin, *Introductory Real Analysis.* New York: Dover, 1975.

[15] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression.* Boston: Kluwer, 1992.

[16] J.A. Hartigan, *Clustering Algorithms.* New York: Wiley, 1975.

[17] T. Tarpey, L. Li, and B.D. Flury, "Principal Points and Self-Consistent Points of Elliptical Distributions," *Annals of Statistics,* vol. 23, no. 1, pp. 103-112, 1995.

[18] V.N. Vapnik, *The Nature of Statistical Learning Theory.* New York: Springer-Verlag, 1995.

[19] B. Kégl, "Principal Curves: Learning, Design, and Applications," PhD thesis, Concorida Univ., Montreal, Canada, 1999.

[20] L. Devroye, L. Györfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition.* New York: Springer-Verlag, 1996.

[21] A.N. Kolmogorov and V.M. Tikhomirov, "ε-Entropy and ε-Capacity of Sets in Function Spaces," *Translations Am. Math. Soc.,* vol. 17, pp. 277-364, 1961.

[22] P. Grother, *NIST Special Database 19.* Nat'l Inst. of Standards and Technology, Advanced Systems Division, 1995.

[23] S. Suzuki and K. Abe, "Sequential Thinning of Binary Pictures Using Distance Transformation," *Proc. Eighth Int'l Conf. Pattern Recognition,* pp. 289-292, 1986.

[24] R.B. Ash, *Real Analysis and Probability.* New York: Academic Press, 1972.

[25] W. Hoeffding, "Probability Inequalities for Sums of Bounded Random Variables," *J. Am. Statistical Assoc.,* vol. 58, pp. 13-30, 1963.

Balázs Kégl received the MSc degree in computer science from the Technical University of Budapest, Hungary, in 1994, and the PhD degree in computer science from Concordia University, Montréal, Canada, in 1999. Since January 2000, he has been a postdoctoral fellow in the Department of Mathematics and Statistics at Queen's University, Kingston, Canada, receiving an NSERC Postdoctoral Fellowship. He is a student member of the IEEE.

Adam Krzyzak received the MSc and PhD degrees in computer engineering from the Technical University of Wroclaw, Poland, in 1977 and 1980, respectively, the DSc degree (habilitation) in electrical engineering from the Warsaw University of Technology, Poland in 1998. In 1980, he became an assistant professor at the Institute of Engineering Cybernetics, Technical University of Wroclaw, Poland. From November 1982 to July 1983, he was a postdoctoral fellow receiving the International Scientific Exchange Award at the School of Computer Science, McGill University, Montréal, Canada. Since August 1983, he has been with the Department of Computer Science, Concordia University, Montréal, where he is currently an associate professor. In 1991, he held the Vineberg Memorial Fellowship at Technion-Israel Institute of Technology and in 1992, Humboldt Research Fellowship at the University of Erlangen-Nürnberg, Germany. He has published in the areas of pattern recognition, neural networks, image processing, identification, and nonparametric estimation. He served as an associate editor of the *IEEE Transactions on Neural Networks* in 1998-1999, and is presently on the editorial board of the journal *Pattern Recognition* and the *International Journal of Applied Software Technology*. He is a senior member of the IEEE.

Tamás Linder received the MS degree in electrical engineering from the Technical University of Budapest, Hungary, in 1988, and the PhD degree from the Hungarian Academy of Sciences in electrical engineering in 1992. He was a postdoctoral fellow at the University of Hawaii in 1992 and a visiting Fulbright scholar at the Coordinated Science Laboratory, University, of Illinois at Urbana-Champaign, in 1993-1994. From 1994 to 1998, he was a faculty member of the Technical University of Budapest in the Department of Computer Science and Information Theory. He was a visiting research scholar in the Department of Electrical and Computer Engineering, University of California, San Diego from 1996 to 1998. He is now an associate professor of mathematics and engineering in the Department of Mathematics and Statistics, Queen's University, Kingston, Ontario, Canada. His research interests include communications and information theory, source coding and vector quantization, machine learning, and statistical pattern recognition. He is a member of the IEEE.

Kenneth Zeger received both the SB and SM degrees in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT) in 1984, and both the MA degree in mathematics and the PhD degree in electrical engineering from the University of California, Santa Barbara, in 1989 and 1990, respectively. He was an assistant professor of electrical engineering at the University of Hawaii from 1990 to 1992. He was in the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory at the University of Illinois at Urbana-Champaign as an assistant professor from 1992 to 1995 and as an associate professor from 1995 to 1996. He has been in the Department of Electrical and Computer Engineering at the University of California at San Diego as an associate professor from 1996 to 1998 and as a professor from 1998 to present. He served as the associate editor at-large for the *IEEE Transactions on Information Theory* during 1995-1998, is serving as a member of the Board of Governors of the IEEE Information Theory Society during 1998-2000, and is a fellow of the IEEE.