Riemannian Manifold Learning

Tong Lin and Hongbin Zha

Abstract—Recently, manifold learning has been widely exploited in pattern recognition, data analysis, and machine learning. This paper presents a novel framework, called Riemannian manifold learning (RML), based on the assumption that the input high-dimensional data lie on an intrinsically low-dimensional Riemannian manifold. The main idea is to formulate the dimensionality reduction problem as a classical problem in Riemannian geometry, that is, how to construct coordinate charts for a given Riemannian manifold? We implement the Riemannian normal coordinate chart, which has been the most widely used in Riemannian geometry, for a set of unorganized data points. First, two input parameters (the neighborhood size k and the intrinsic dimension d) are estimated based on an efficient simplicial reconstruction of the underlying manifold. Then, the normal coordinates are computed to map the input high-dimensional data into a low-dimensional space. Experiments on synthetic data, as well as real-world images, demonstrate that our algorithm can learn intrinsic geometric structures of the data, preserve radial geodesic distances, and yield regular embeddings.

Index Terms—Dimensionality reduction, manifold learning, manifold reconstruction, Riemannian manifolds, Riemannian normal coordinates.

1 Introduction

In appearance-based object recognition, a 64×64 image is directly represented as a vector in a 4,096-dimensional vector space. Obviously, this high-dimensional vector space is too sparse to allow any efficient processing and analysis. A typical way to circumvent "the curse-of-dimensionality" problem [1] is to use *dimensionality reduction* techniques. The purpose of dimensionality reduction is to map a set of high-dimensional data into a low-dimensional space while preserving the intrinsic structure in the data. In pattern recognition, dimensionality reduction serves as an automatic-learning approach to feature extraction by combining all important cues (for example, shape, pose, and lighting for image data) into a unified framework. Essentially, dimensionality reduction offers an efficient approach for redundancy removal or data reduction. The low-dimensional representation of high-dimensional data has been regarded as a central problem in data analysis [2]. Due to the pervasiveness of high-dimensional data, dimensionality reduction techniques have found widespread use in many applications such as pattern recognition, data analysis, and machine learning. Among the family of dimensionality reduction approaches, *manifold learning*[1] algorithms have attracted extensive attention recently due to their nonlinear nature, geometric intuition, and computational feasibility.

1. Throughout this paper, manifold learning refers to a large family of nonlinear dimensionality reduction methods based on the assumption that the input data are sampled from a smooth manifold. It may have other meanings. For instance, in [46], manifold learning is concerned with the problem of constructing a simplicial complex to approximate the underlying manifold. More generally, manifold learning can be defined as a process that automatically learns the geometric and topological properties of a given manifold.

• *The authors are with the State Key Laboratory of Machine Perception, Science Building, School of EECS, Peking University, Beijing 100871, China. E-mail: {lintong, zha}@cis.pku.edu.cn.*

Manuscript received 9 July 2006; revised 16 Jan. 2007; accepted 11 June 2007; published online 2 July 2007.
Recommended for acceptance by A. Srivastava.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0505-0706.
Digital Object Identifier no. 10.1109/TPAMI.2007.70735.

1.1 Previous Work

Classical linear dimensionality reduction methods such as the principal component analysis (PCA) [3], [4], multidimensional scaling (MDS) [5], and linear discriminant analysis (LDA) [6] can only deal with flat euclidean structures. They fail to discover the curved or nonlinear structures of the input data. Different nonlinear extensions of PCA and MDS have been proposed, including self-organizing maps (SOMs) [7], principal curves [8], [9], autoencoder neural networks [10], and generative topographic maps (GTMs) [11]. However, these methods often suffer from the difficulties in designing cost functions or tuning too many free parameters. Moreover, most of these methods are computationally expensive, thus limiting their utility in high-dimensional data sets. Kernel-based extensions [12] such as the kernel PCA (KPCA) and kernel Fisher discriminant analysis (KFD) provide new means to perform PCA and LDA in an implicit higher dimensional feature space.

A large number of nonlinear manifold learning methods have been proposed recently, including isometric feature mapping (ISOMAP) [13], [14], [15], locally linear embedding (LLE) [16], [17], Laplacian eigenmaps [18], [19], Hessian eigenmaps [20], semidefinite embedding (SDE) [21], manifold charting [22], local tangent space alignment (LTSA) [23], diffusion maps [24], [25], [2], and conformal eigenmaps [26]. The major algorithms are listed chronologically in Table 1. The basic assumption is that the input data lie on or close to a smooth low-dimensional manifold [27]. Each manifold learning algorithm attempts to preserve a different geometrical property of the underlying manifold. Local approaches such as LLE [16], [17], Laplacian eigenmaps [18], [19], and LTSA [23] aim to preserve the proximity relationship among the data. They are also called *spectral embedding methods* [21], as the low-dimensional embedding is reduced to solving a sparse eigenvalue problem under the unit covariance constraint. However, due to this imposed constraint, the aspect ratio is lost, and the global shape of the embedding data cannot reflect the underlying manifold. In contrast, global approaches like ISOMAP [13] aim to preserve the metrics at all scales, hence giving a more faithful embedding. However,

TABLE 1
Major Manifold Learning Algorithms

Authors	Year	Algorithm	Property	Description and Comments
Tenenbaum et al. [13]	2000	ISOMAP	Isometric mapping	Computes the geodesic distances, and then uses MDS. Computationally expensive.
Roweis et al. [16]	2000	LLE	Preserving linear reconstruction weights	Computes the reconstruction weights for each point, and then minimizes the embedding cost by solving an eigenvalue problem.
Silva et al. [14]	2003	C-ISOMAP and L-ISOMAP	Conformal ISOMAP and landmark ISOMAP	C-ISOMAP preserves angles. L-ISOMAP efficiently approximates the original ISOMAP by choosing a small number of landmark points.
Belkin et al. [18]	2003	Laplacian eigenmaps	Locality preserving	Minimizing the squared gradient of an embedding map is equal to finding eigenfunctions of the Laplace-Beltrami operator.
Donoho et al. [20]	2003	HLLE or Hessian eigenmaps	Locally isometric to an open, connected subset	A modification of Laplacian eigenmaps by substituting the Hessian for the Laplacian. Computationally demanding.
Brand [22]	2003	Manifold charting	Preserving local variance and neighborhood	Decomposes the input data into locally linear patches, and then merges these patches into a single low-dimensional coordinate system by using affine transformations.
Zhang et al. [23]	2004	LTSA	Minimizing the global reconstruction error	First constructs the tangent space at each data point, and then aligns these tangent spaces with a global coordinate system.
Weinberger et al. [21]	2004	SDE	Local isometry	Maximizing the variance of the outputs, subject to the constraints of zero mean and local isometry. Computationally expensive by using semidefinite programming.
He et al. [19]	2005	Laplacianfaces	Linear version of Laplacian eigenmaps	The minimization problem reduces to a generalized eigenvalues problem.
Coifman et al. [24][25]	2005	Diffusion maps	Preserving diffusion distances	Given a Markov random walk on the data, the diffusion map is constructed based on the first few eigenvalues and eigenvectors of the transition matrix P.
Sha et al. [26]	2005	Conformal eigenmaps	Angle-preserving embedding	Maximizing the similarity of triangles in each neighborhood. More faithfully preserving the global shape and the aspect ratio. Semidefinite programming is used to for optimization.
Law et al. [15]	2006	Incremental ISOMAP	Data are collected sequentially.	Efficiently updates all-pair shortest path distances, and solves an incremental eigenvalue problem.

ISOMAP, which is based on the rigid constraint of isometric mapping, can only be applied to intrinsically flat manifolds, for example, 2D developable surfaces (cylinders, cones, and tangent surfaces). ISOMAP first computes the shortest path distances between all pairs of points on the manifold and then applies MDS to the distance matrix. Both the distance computation and MDS are computationally expensive.

1.2 Manifold Assumption

Most manifold learning algorithms assume that the input data resides on or close to a low-dimensional manifold embedded in the ambient space. For most applications, the data is generated by continuously varying a set of parameters. Often, the number of parameters is not very large. Otherwise, there will not be much benefit from dimensionality reduction. For example, a set of face images can be captured by varying the face pose, scale, position, and lighting conditions. Following the study in [27], here, we present an explicit representation of the underlying manifold on which the face images lie. This representation is obtained by considering a simple geometric imaging model (Fig. 1a) to acquire face images. For simplicity, only varying poses and lighting conditions are considered in this model, as they are the most important factors in face recognition. This model may be adapted to the image data of other objects (for example, cars) if similar imaging conditions are encountered.

We model the human head as the unit sphere S^2, where the human face is on the frontal hemisphere. Different poses are obtained by moving the camera while the human face is fixed. We require that the distance from the camera to the face and the focal length of the camera are fixed, so the acquired images have similar scales. The optical axis of the camera is set to pass through the center of the sphere. In order to capture rotated face images on the image plane, the camera is allowed to have some degrees of freedom to rotate around its optical axis. The lighting is modeled by a point light source far away from the sphere. Under these settings, the face data is sampled from a five-dimensional manifold, which is homeomorphic to

$$M = \{(P, Q, e) | P \in S^2, Q \in S^2, e \in S^1\}, \tag{1}$$

where P and Q are two intersection points on S^2 by the optical axis of the camera and the light ray, respectively, and e is a unit vector indicating the planar rotation angle of the camera. If the illumination variation is ignored, a 3D manifold may be generated

$$M' = \{(P, e) | P \in S^2, e \in S^1\}. \tag{2}$$

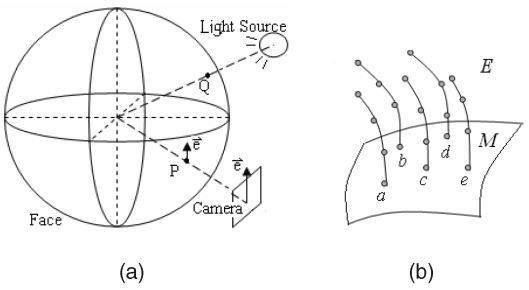


Fig. 1. Manifold assumption. (a) A geometric imaging model for face image data of one person. (b) A vector bundle model for face images of multiple persons (a, b, c, d, and e).

This manifold can be conceived as the earth running on a circular orbit in the four-dimensional space-time.

For face images of multiple persons, a vector bundle is suitable to model the relation between interperson and intraperson variation (Fig. 1b). Each fiber (shown in Fig. 1b as a curve for visualization) represents the face data manifold of one person, and the base manifold M (shown in Fig. 1b as a surface) collects the face images of all persons in a standard setting (such as the frontal view, frontal illumination, and natural facial expression). In face recognition, a new test sample can be projected from the total space E (formed by gluing together all the face image manifolds of each person) onto the base M in order to remove intraperson variation, provided that the projection map $\pi : E \rightarrow M$ has been learned. It is important to point out that both the total space E and the base M are manifolds. Therefore, manifold learning algorithms can be applied to a set of face image data of multiple persons in face recognition.

In general, the manifold assumption may not hold for some complex data sets acquired from the real world. Nevertheless, this assumption has been widely exploited in practice. First, it is still possible to approximate the true data model as a manifold by considering only a small set of significant factors. Second, the popular manifold assumption can greatly simplify the problem by exploiting the classical frameworks of differentiable manifolds and Riemannian geometry.

In the remainder of this paper, we assume that the input data lie on a d-dimensional Riemannian manifold M embedded in a euclidean space R^n. The Riemannian metric on M is induced from the standard euclidean inner product in R^n. Mathematically, Nash proved that every Riemannian manifold M can be isometrically embedded into some euclidean space R^m [35]. This theorem reduces the study of Riemannian manifolds to the study of submanifolds of euclidean spaces.

1.3 The Proposed Framework

In this paper, we propose a general framework called *Riemannian manifold learning*[2] (RML). The dimensionality reduction problem is formulated as constructing coordinate charts for a Riemannian manifold. One of the most widely used is the (*Riemannian*) *normal coordinate* chart, which was introduced by Riemann in his famous 1854 lecture [28]. Riemannian normal coordinates are a vital tool for calculations in Riemannian geometry. One advantage is that normal coordinates can preserve geodesic metrics to a certain extent if the complete preservation of the metrics is not possible. The metric preserving property is a cornerstone in geometry and also a desirable feature for dimensionality reduction. In supervised or unsupervised classification, better preserving the metrics often leads to better results in the reduced low-dimensional space. Particularly, in normal coordinates, geodesic distances on each radial geodesic curve emanating from the coordinate origin can be faithfully preserved. On the other hand, distances between two radial geodesic curves can freely stretch or shrink to fit this mapping. Fig. 2a provides an example of radial geodesic curves, and Fig. 2b presents the basic idea of normal coordinates. These radial geodesic curves play a key role in unfolding a curved manifold onto a flat space. This is very similar to the role of an umbrella skeleton when we open an umbrella. Because of this set of radial geodesic curves, which is a "skeleton" of the manifold, normal coordinates can preserve the intrinsic geometric structures with minimal distortions.

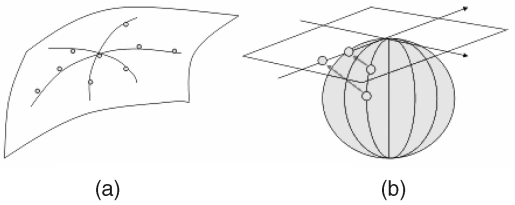


Fig. 2. (a) Radial geodesic curves on a surface. (b) Charting a sphere with Riemannian normal coordinates on the tangent plane.

This paper presents a novel algorithm for manifold learning based on the calculations of Riemannian normal coordinates. This algorithm has a number of desirable properties and overcomes several problems in previous work reported in the literature:

1. The *metric preserving* problem. ISOMAP attempts to preserve geodesic distances of all pairs of points on the manifold. For intrinsically flat manifolds of zero Gaussian curvature, such as the well-known "Swiss roll" data (Fig. 3a), ISOMAP yields the ideal embedding by isometrically unfolding the curved data onto a planar region (Fig. 3b). However, for general manifolds with a nonzero Gaussian curvature (for example, a sphere), ISOMAP fails to perform the isometric mapping onto a flat euclidean space, as Gaussian curvature is isometry invariant. On the contrary, global metric information is totally lost in those spectral embedding methods under the unit variance constraint. For the example of Swiss roll data, spectral methods tend to generate an embedding into a square region (Fig. 3c) or even sometimes yield unpredictable irregular results (Fig. 3d). For pattern recognition applications, large differences in global shape and aspect ratios pose serious threats to any classifier (for example, (KNN)), because neighborhood relationship is changed greatly. Mathematically, the metric preserving problem is of paramount importance in Gaussian intrinsic geometry and Riemannian geometry. Compared with ISOMAP and spectral methods, the proposed RML can preserve radial geodesic distances, give a more faithful representation of the data's global structure, and achieve a trade-off between the rigid constraint of isometry and the deficiency of global metrics.
2. The *cost averaging* problem. Previous methods based on global cost optimization attempt to average the cost among all data points, thus preventing the correct unfolding in large deformation areas. For instance, the boundary areas of a punctured sphere (Fig. 4a) often shrink into a circular curve by using Hessian eigenmaps or LTSA (Fig. 4b). This shrinkage significantly decreases the global cost but yields incorrect overlapping. Our algorithm computes the embedding by

2. Although several existing methods mentioned Riemannian manifolds in their papers, their proposed algorithms essentially had little connection with Riemannian geometry. This work, together with [32], points out that the manifold learning problem can be translated into the mathematical problem of constructing coordinate charts for the manifold, thus unifying various ideas within a common mathematical framework. In this paper, only the Riemannian normal coordinate chart is implemented, but other coordinate charts can also be employed.

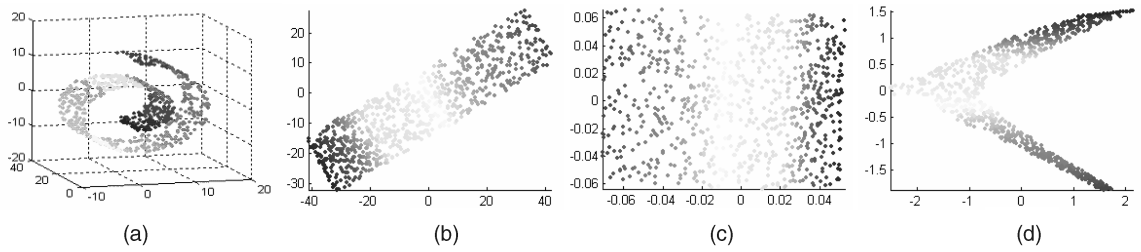


Fig. 3. The metric preserving problem. (a) A Swiss roll data set. (b) An ideal isometric embedding. (c) Embedding results using the spectral method LTSA. (d) Embedding results using another spectral method LLE.

using a local optimization procedure for each point, thus avoiding this cost averaging problem.

3. The *incremental-learning* problem. Most manifold learning methods operate in a "batch" mode, in which low-dimensional coordinates of all data points are computed simultaneously. This batch mode cannot handle the problem of out-of-sample extension when new data points become available. Incremental learning is essential for practical applications (such as pattern recognition), where new test samples need to be mapped into the same low-dimensional embedding space. Recently, out-of-sample extensions for ISOMAP, LLE, and eigenmaps are proposed in [15], [29], [30], [31]. Our algorithm is essentially incremental learning, since each time, we compute the embedding coordinates for one new data point.
4. The *global optimization* problem. Most existing algorithms aim to solve a global optimization problem to obtain the low-dimensional representation of all data points simultaneously. Commonly, these global optimization problems are very challenging to solve, or the computational load is extremely heavy. For example, ISOMAP solves the global optimization by using MDS, which is very slow. Several methods based on semidefinite programming are also computationally demanding. In eigenmaps methods, a typical trick is to impose a unit variance constraint, thus reducing the optimization to an eigenvalue problem that can be efficiently solved. However, this imposed constraint leads to the first problem mentioned above. In contrast, our algorithm divides the global embedding problem into multiple local optimization steps, which can be easily and efficiently solved.

The study of Brun et al. [32] is the most closely related to our work, which is also based on the concept of Riemannian normal coordinates. Their method achieved good results on three synthetic data sets. However, densely sampled data sets are required in their algorithm in order to calculate geodesic distances and directions. For example, a "Swiss roll" data set of 2,000 points is used in their experiments, whereas commonly 800 or 1,000 points are enough for other methods. In addition, their method cannot handle the data sets with holes, for example, the "Swiss hole" data set. In contrast, our proposed algorithm overcomes these problems.

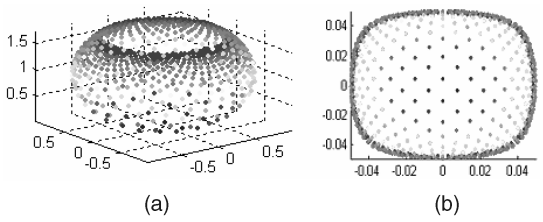


Fig. 4. The cost averaging problem. (a) A punctured sphere data set. (b) Embedding results using the spectral method LTSA.

The main contributions of this paper include the following: 1) It provides a manifold charting algorithm that efficiently computes Riemannian normal coordinates for a d-dimensional Riemannian manifold M. The calculation of geodesic curves can also be useful to many other geometric problems. 2) It provides an algorithm that adaptively selects the neighborhood size k_i for each data point x_i and reliably estimates the intrinsic dimension d. 3) It provides a thorough overview of prior work and a detailed comparison between the proposed RML and other algorithms.

It is important to state, at the outset, that the algorithm presented in this paper will *not* attempt to deal with two problems that plague any manifold learning algorithm: *noise* and *undersampling*. From a practical point of view, a manifold learning algorithm must be able to address both problems when dealing with real data sets. However, given arbitrary and unknown geometry and topology, the problem of manifold learning from noiseless and sufficiently dense data is still a very difficult challenge. The proposed algorithm may be thought of as a first step, and the ultimate goal would be to extend the framework to more realistic and challenging cases that involve these two problems.

1.4 Organization

The paper is organized as follows: We begin with a brief review of mathematical preliminaries in Section 2. In Section 3, we describe how to select two important input parameters, k_i and d, by using manifold reconstruction. Section 4 presents the charting algorithm to compute Riemannian normal coordinates. Experimental results are given in Section 5, and Section 6 is devoted to the complexity analysis. An earlier version of this work appeared in the Ninth European Conference on Computer Vision (ECCV '06) [33].

2 Mathematical Preliminaries

We briefly review some necessary concepts in Riemannian geometry [34], [35]. A bijective map is called a *homeomorphism* if it is continuous in both directions. A *(topological) manifold* M of dimension d is a connected Hausdorff space for which every point has a neighborhood U that is homeomorphic to an open subset V of R^d. Such a homeomorphism $x : U \to V$ is called a *(coordinate) chart*. An *atlas* is a family $\{U_\alpha, x_\alpha\}$ of charts for which the $\{U_\alpha\}$ constitute an open covering of M. A manifold M is called a *differentiable manifold* if there is an atlas of M, $\{U_\alpha, x_\alpha\}$, such that all chart transitions

$$x_\beta \circ x_\alpha^{-1} : x_\alpha(U_\alpha \cap U_\beta) \rightarrow x_\beta(U_\alpha \cap U_\beta) \quad (3)$$

are differentiable of class C^∞. A differentiable manifold M endowed with a smooth inner product (called Riemannian metric) $g(u, v)$ or $\langle u, v\rangle$ on each tangent space T_pM is called a *Riemannian manifold* (M, g).

The *exponential map* $\exp_p(v)$ is a transform from a tangent vector $v \in T_pM$ into a point $q \in \gamma \subset M$ such that $\mathrm{dist}(p, q) = \|v\| = \langle v, v\rangle^{1/2}$, where γ is the unique geodesic traveling through p with the tangent vector v. A *geodesic* is a smooth curve that locally joins their points along the shortest path. All the geodesics passing through p are called *radial geodesics*. The local coordinates defined by the chart $(U, \exp_p^{-1})$ are called (*Riemannian*) *normal coordinates* with center p. Note that normal coordinates preserve the distances on radial geodesics. For example, unfolding a sphere onto a plane in normal coordinates can preserve the distances on great circles, like paring an orange. Note that for a manifold with complex topology, for example, a torus, multiple normal coordinate charts, rather than one single chart, are needed to cover the whole manifold.

3 Parameter Estimation

There are two important input parameters for most manifold learning algorithms: the neighborhood size k and the intrinsic dimension d. An appropriately chosen neighborhood size is the key to the success of these algorithms [36]. Choosing a large neighborhood may introduce "short-circuit" edges between two separating branches, which drastically alter the original topological connectivity. On the other hand, choosing a small neighborhood might fragment the manifold into a large number of disconnected regions. In [36], a "trial-and-error" method is proposed for choosing a reasonable neighborhood size, based on a trade-off between two cost functions. Clearly, this method is costly and cumbersome, as the embedding algorithm runs repeatedly by testing different sizes. Moreover, a fixed neighborhood size (k or radius ε) may not be suitable for data sets with large variations. In [37], a neighborhood contraction and expansion method is presented to adaptively select k_i at each point x_i.

Intrinsic dimensionality estimation is a classical problem in pattern recognition [38]. The existing approaches can be roughly divided into two groups: eigenvalue projection methods [39], [40] and geometric methods based on NN distances [41], [42], [43], [44] or fractal dimensions [45]. In [13], ISOMAP determines the intrinsic dimension by looking for an "elbow" point, at which the residual variance ceases to decrease significantly with added dimensions. Nevertheless, it is cumbersome to try each dimension. Moreover, sometimes there may not be a clear "elbow" point.

The two parameters, the neighborhood size k and the intrinsic dimension d, reflect the topological properties of the underlying manifold M. We present a manifold reconstruction method to estimate these parameters. Our manifold reconstruction method greatly simplifies Freedman's method [46], which is computationally expensive on the optimization of convex hulls. In our method, edge connections are first constructed to adaptively determine the neighborhood size, and then, the underlying manifold is reconstructed as a simplicial complex. Naturally, the dimension of this complex serves as a reliable estimation of the intrinsic dimension of M.

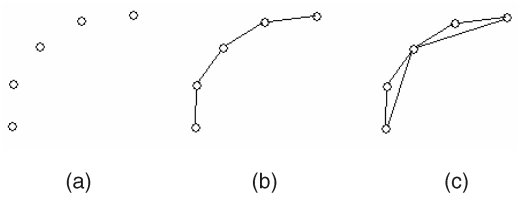


Fig. 5. Reconstruction of five points sampled from a curve. (a) Unorganized data points. (b) One-dimensional reconstruction. (c) Two-dimensional reconstruction.

The key to the manifold reconstruction from a set of unorganized data points is recovering edge connections correctly. Consider a simple example of five points sampled from a curve (Fig. 5). It is apparent that the one-dimensional (1D) reconstruction in Fig. 5b is much better than the 2D reconstruction in Fig. 5c. These points are more likely to be sampled from a 1D curve than a 2D surface. The width of the 2D surface in Fig. 5c is too small and thus can be negligible. In fact, any thin rope in the physical world can be modeled as a 1D curve by ignoring its radius. This simple example motivates the concepts of "visible" neighbors in order to avoid the "unreasonably" long edge connections that occurred in Fig. 5c.

The neighborhood selection algorithm for each point x_i consists of the following three steps (illustrated in Fig. 6 on an S-curve example):

1. Search its K-NNs, denoted $KNN(x_i)$, with a large enough K. Connect an edge between x_i and each of its K-NN.
2. Determine the "visible" neighbors $VN(x_i) = \{y \in KNN(x_i) | \langle x_i - z, y - z\rangle \geq 0, \forall z \in KNN(x_i)\}$ of x_i and delete edge connections with nonvisible neighbors. A point y is said to be a visible neighbor of x_i if there is no other point z that separates y and x_i. Equivalently, it requires that the angle between any two adjacent edges should be acute or right. Obtuse angles are prohibited. This property guarantees well-shaped simplices in manifold reconstruction.
3. Obtain the "safe" neighborhood, denoted $SN(x_i)$, by removing the short-circuit neighbors. We observe that short-circuit edges are often much longer than other "safe" edges. Otherwise, the data is certainly ill posed (or too sparse), and we cannot discern disjoint branches. Based on this observation, we use a simple "jump" detection method to remove these short-circuit edges. First, edges are sorted in the ascending order of lengths, denoted $\{\vec{e}_1, \cdots, \vec{e}_k\}$, if there are k visible neighbors. Second, we use PCA[3] to estimate the local intrinsic dimension d_j of the first $j (1 < j \leq k)$ edges $\{\vec{e}_1, \cdots, \vec{e}_j\}$. Then, we have an array of dimensions $\{d_1, \cdots, d_k\}$ for edges $\{\vec{e}_1, \cdots, \vec{e}_k\}$. If $d_j > d_{j-1}$, compute the jump of increased length $|e_j| - |e_{j-1}|$. If

3. This method, based on nonzero eigenvalues of the covariance matrix, was proposed by Fukunaga and Olsen [38], [39]. Eigenvalues are first normalized by dividing them by the largest eigenvalue. The intrinsic dimension is defined as the number of normalized eigenvalues that are larger than a threshold T. We set $T = 0.08$ in all experiments.

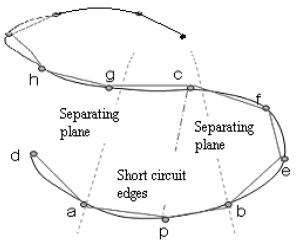


Fig. 6. An S-curve example to illustrate the three steps of adaptive neighborhood selection for the current point p. 1) Determine the K-NN neighborhood $(K = 8)$, $KNN = \{a, b, c, d, e, f, g, h\}$. 2) Determine the visible neighborhood $VN = \{a, b, c\}$. 3) Obtain the safe neighborhood $SN = \{a, b\}$ by removing the short-circuit neighbor c.

the maximal jump is larger than a given threshold,[4] posterior edges are thought of as short-circuit edges and, thus, are removed. If it is less than the threshold, all edges are thought of as safe edges.

As 1D simplices (that is, edges) have been determined, simplices in higher dimensions are constructed by grouping adjacent edges. For example, if (a, b) is an edge and c is any other point, then a triangle (a, b, c) is constructed when there are two edges (a, c) and (b, c) in the edge list. This procedure repeats from a low dimension to a high dimension until there are no new simplices generated. The target simplicial complex is composed of all the simplices. The dimension of the complex, defined as the maximal dimension of its simplices, serves as a good estimate[5] of the intrinsic dimension of M.

4 Manifold Charting

4.1 Our Previous Algorithm

We first briefly describe our previous charting algorithm presented in ECCV '06 [33]. It consists of three steps: 1) choose a base point p as the origin of the normal coordinate chart, 2) compute the tangent space T_pM at p and set up a Cartesian coordinate chart, 3) use Dijkstra's algorithm to find single-source shortest paths and calculate the normal coordinates for each endpoint of the shortest paths.

In general, the base point p may be selected arbitrarily. Recall that for a manifold with complex topology, multiple charts are needed and, therefore, multiple base points will be chosen. However, for a manifold with boundaries, we prefer choosing a base point close to the manifold center so that one single chart at this point can represent the entire manifold. Computationally, the center can be found by solving a minimax problem. That is, the maximal geodesic distance between a candidate point and any other point is called a geodesic radius, and the candidate point with the minimal geodesic radius is the center. Note that here, we need *not* determine the boundary explicitly, which would be difficult and unstable for real data sets. Instead, Dijkstra's algorithm [47], [48] is utilized to calculate the shortest path between a candidate point and another point on the manifold, which approximates the geodesic curve between these two points.

A coordinate chart is set up by computing the tangent space T_pM

$$x_0 + \text{span}\{x_1 - x_0, \ldots, x_d - x_0\}, \tag{4}$$

where $\{x_0, x_1, \ldots, x_d\}$ are $(d + 1)$ geometrically independent edge points (or NNs) of p. Points $\{x_0, x_1, \ldots, x_d\}$ are said to be *geometrically independent* if the vectors $\{x_1 - x_0, \ldots, x_d - x_0\}$ are linearly independent. Any point on the tangent space can be represented as

$$x_0 + \sum_{i=1}^{d} \lambda_i (x_i - x_0). \tag{5}$$

An orthonormal frame, denoted $(p; \vec{e}_1, \cdots, \vec{e}_d)$, is computed from the vectors $\{x_1 - x_0, \ldots, x_d - x_0\}$ by using the Gram-Schmidt orthogonalization.

Then, Dijkstra's algorithm [47], [48] is exploited to find single-source shortest paths in the graph determined by the simplicial complex. Each time a new shortest path is found, the normal coordinates of the endpoint on this new path are computed. If the endpoint q is an edge point of p, we directly compute the projection of q, denoted $q' \in R^d$, onto the tangent space $(p; \vec{e}_1, \cdots, \vec{e}_d)$ by solving the following least squares problem:

$$\min_x \left\| q - \left(p + \sum_{i=1}^{d} x_i \vec{e}_i \right) \right\|_{R^n}^2, \tag{6}$$

where $x = (x_1, x_2, \ldots, x_d) \in R^d$ are the projection coordinates of q' in the tangent space. The normal coordinates of q are given by

$$\frac{\|q - p\|_{R^n}}{\|x\|_{R^d}} x \tag{7}$$

since normal coordinates attempt to preserve distances on each radial geodesic.

If the endpoint $q \in M \subset R^n$ is not an edge point of p, the normal coordinates of q (denoted $q' \in R^d$) is computed by solving a quadratically constrained linear least squares problem. Let point b be the previous point on the shortest path from p to q. Suppose that b has k edge points $\{c_1, \ldots, c_k\}$ whose normal coordinates have been computed previously. The number k of these points is required to be larger than or equal to d in order to guarantee the correct solution. (One exception may occur at the beginning of Dijkstra's algorithm, when k is less than d. In this case, point q is treated as an edge point of p to compute its normal coordinates.) Fig. 7a shows such an example with $k = 3$. The basic idea is that we aim to preserve the angles in the neighborhood of b while at the same time keeping the distance between q and b unchanged. This leads to the following linear least squares problem:

$$\cos\theta_i = \frac{\langle q - b, c_i - b \rangle}{\|q - b\| \cdot \|c_i - b\|} \approx \cos\theta_i' = \frac{\langle q' - b', c_i' - b' \rangle}{\|q' - b'\| \cdot \|c_i' - b'\|}, \quad i = 1, 2, \ldots, k \tag{8}$$

4. The threshold can be set as ρ times the average pairwise distance of these neighbors, which reflects the idea of local sampling density in this small region. ρ is a parameter that can be set. It is not our intention to state that the proposed method can succeed to remove all short-circuit edges in any cases. In fact, we have no "ground-truth" short-circuit edges for a set of unorganized data points. The parameter ρ provides a way to control the quality of short-circuit edge removal. Improving the robustness of this method is our future work.

5. Certainly, sparse sampling and noise pose serious threats to any attempt to estimate the intrinsic dimension of an unorganized data set. Here, our motivation is to provide a new method to estimate the dimension, which would be an input parameter to the following charting algorithm. Although dimension estimation is only a preprocessing step in the whole framework and is not our main purpose in this paper, the objective is to point out a new solution to estimating dimensions and neighbors via manifold reconstruction.

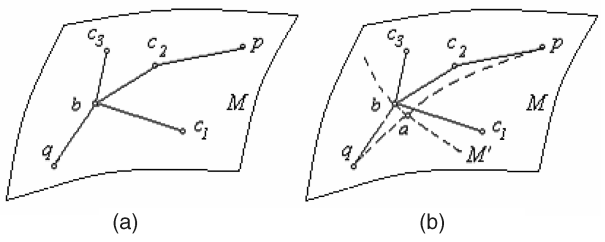


Fig. 7. An example illustrating how to compute the normal coordinates of point q, which is not in the local region of the base point p. (a) Previous algorithm. Point b is the preceding node on the shortest path from p to q. Points c_1, c_2, and c_3 are neighbors of b. Suppose that the normal coordinates of b, c_1, c_2, and c_3 have been computed. The normal coordinates of q are computed based on geometric relations to b, c_1, c_2, and c_3. (b) The proposed new algorithm based on the geodesic anchor point a. M' is the level set passing through points b and a. The normal coordinates of q are computed based on geometric relations to a, c_1, c_2, and c_3, where a replaces the role of b in the new algorithm.

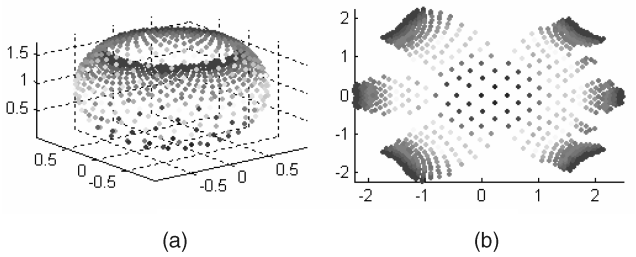


Fig. 8. An example illustrating a failure case of our previous algorithm by using the shortest path approximation to geodesic curves. This approximation causes large errors in the calculation of geodesic directions, which splits the outer region of the punctured sphere data into several separated pieces.

with a quadratic constraint

$$\|q - b\| = \|q' - b'\|, \tag{9}$$

where q', b', and c_i' are the normal coordinates of q, b, and c_i, respectively. Our goal is to compute $q' \in R^d$. The problem can be rewritten in a standard form

$$\min_x \|Ax - y\|^2 \text{ subject to } \|x\| = \alpha, \tag{10}$$

where $x = q' - b'$, $\alpha = \|q - b\|$, $y = [\cos\theta_1, \ldots, \cos\theta_k]^T$, and A is a $k \times d$ matrix, with each row being $\frac{(c_i' - b')^T}{\alpha \|c_i' - b'\|}$. The method given in Appendix A can efficiently solve this least squares problem.

4.2 Limitations

Our previous algorithm has two limitations. One is in finding a set of geometrically independent points, $\{x_0, x_1, \ldots, x_d\}$, in order to set up a coordinate chart. If this set of points is not geometrically independent, a new set of points must be examined again. This "trial-and-error" procedure is cumbersome.

Another limitation is that shortest paths are exploited to approximate geodesic curves. In the calculation of Riemannian normal coordinates, geodesic curves are essential to determine the geodesic direction and geodesic distance for a point. Generally speaking, if the underlying manifold is densely sampled, the shortest path approximation may still offer satisfactory accuracy. However, if data is sparsely sampled, this approximation will result in large errors. Fig. 8 shows such a failure case by using the shortest path approximation on the sparsely sampled punctured sphere.

4.3 The New Algorithm

Two new features are incorporated to address the limitations of our previous algorithm. The first one is replacing the Gram-Schmidt orthogonalization by a PCA projection at the base point p. In a PCA projection, more than $(d+1)$ points are selected to obtain a stable coordinate chart. This is a one-step procedure, avoiding the "trial-and-error" iteration in the Gram-Schmidt orthogonalization. In addition, the tangent space does not need to be computed explicitly; thus, the step of tangent space computation can be skipped. We directly apply PCA to a local region of p, consisting of points $\{x_0, x_1, \ldots, x_k\} (k > d)$.[6] We set $x_0 = p$, and other k points are obtained by running Dijkstra's algorithm from the base point p. As the shortest paths are found, the first k endpoints of these paths are collected into the local region of p. The normal coordinates of these points are computed by using a PCA projection.

The second new feature is replacing the "inaccurate" shortest path approximation by "true" geodesic curves. These "true" geodesic curves can produce more accurate normal coordinates. A basic idea to find a geodesic curve from p to another point q, as shown in Fig. 7b, is searching for a length-minimizing curve among an admissible family of curves connecting p and q. This idea leads to the subject known as the *calculus of variations*. For our purpose, we only need to consider a slight perturbation of point b, which is the preceding node on the shortest path from p to q. This perturbed point, denoted as a (see Fig. 7b), is called a *geodesic anchor point*. In our new algorithm, the anchor point a replaces the role of point b. This replacement can provide more accurate results in calculating normal coordinates of the current point q. In [49], several methods to compute a geodesic curve are based on the geodesic equation that any geodesic curve has to satisfy. These methods require that the manifold has been parameterized, that is, a coordinate chart has been constructed to cover p and q. However, this requirement is just our goal, hence making these methods unsuitable here.

A geodesic anchor point a can be computed in a local region of point b by minimizing the curve length from p to q. To make the optimization problem tractable, we restrict the search of anchor point a on a $(d-1)$-dimensional level set (or submanifold) M' (see Fig. 7b) to q. The level set M' consists of all points that have identical geodesic distances (to base point p) to that of b. To compute the level set, we first fit the geodesic distance function, denoted as $u(.)$, with a two-order polynomial in a local region of b. Then, the anchor point a, which is the closest point on M' to q, can be found by using the numerical method given in Appendix B.

For simplicity, a two-order polynomial $u(.)$ is exploited to fit the geodesic distance function in a local region of b. To reduce the number of fitting coefficients, points in the local region of b are first projected onto the tangent space at b using PCA. There are $(d+2)(d+1)/2$ coefficients in the two-order polynomial. For example, if $d = 2$, the two-order polynomial $u(.)$ can be written as

6. k is required to be larger than d in order to generate a stable tangent space. On the other hand, to avoid an inaccurate approximation, k cannot be too large. Nevertheless, an appropriate value of k can be chosen over a certain range, for example, from $d+1$ to $2d$, and can be applied to most cases.

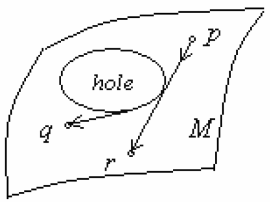


Fig. 9. An example illustrating that the original definition of normal coordinates "fails" when the data has a hole. Points q and r have the same geodesic directions at p, thus leading to "incorrect" normal coordinates under that definition.

$$u(x_1, x_2) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_1 x_2 + w_5 x_2^2. \quad (11)$$

Let the local region of b consist of k points $\{c_1, c_2, \ldots, c_k\}$, whose normal coordinates have been computed. Accordingly, the geodesic distances of these k points are known. A standard least squares solution is used to fit this geodesic distance function. Note that k is required to be larger than (or equal to) $(d+2)(d+1)/2$ in order to solve this fitting problem correctly.

The level set M' can be represented by $u(x) - u(b) = 0$, which is a general quadratic manifold. The closest point in M' to q is the geodesic anchor point a. Accordingly, a whole geodesic curve segment in the local region can be computed by replacing $u(b)$ with other geodesic distances in the equation of M'. Moreover, starting from a, we can compute a preceding geodesic curve segment. Therefore, by repeating this procedure, we can compute a complete geodesic curve starting from p to q, though only an anchor point a is required in this work.

The next step is to solve the following linear least squares problem:

$$\cos\theta_i = \frac{(q-a)\cdot(c_i-a)}{\|q-a\|.\|c_i-a\|} \approx \cos\theta_i' = \frac{(q'-a')\cdot(c_i'-a')}{\|q'-a'\|\cdot\|c_i'-a'\|}, \quad i = 1, 2, \ldots, k \quad (12)$$

subject to a quadratic constraint $\|q-a\| = \|q'-a'\|$, where q, a, and c_i are tangent coordinates obtained by using PCA, and q', a', and c_i' are the corresponding normal coordinates. Note that the normal coordinates a' can be obtained by locally fitting the coordinate mapping with a linear transform. All coordinates in (12) are of R^d. Again, the problem can be rewritten as

$$\min_x \|Ax - y\|^2 \text{ subject to } \|x\| = \alpha, \quad (13)$$

where $x = q' - a'$, $\alpha = \|q-a\|$, $y = [\cos\theta_1, \ldots, \cos\theta_k]^T$, and A is a $k \times d$ matrix with each row being

$$\frac{(c_i' - a')^T}{\alpha\|c_i' - a'\|}.$$

The method given in Appendix A is also used to solve the unknown q'.

4.4 Discussions

The original definition of normal coordinates needs to compute the geodesic direction of q at the base point p. To this end, one has to compute the complete geodesic curve from p to q. The geodesic direction at p must be of high accuracy in order to discern subtle direction differences. In addition, the original definition cannot handle the case that the data manifold has "holes" (see the failure example shown in Fig. 9). Due to these reasons, we modify the original definition by exploiting the geodesic direction in a local region of q. Since a normal coordinate chart preserves the distances on a radial geodesic curve, we only need to preserve the distance between q and a, denoted as $dist(q, a)$, if $dist(a, p)$ has been preserved. Therefore, our algorithm can restrict the calculation of normal coordinates in a local region only.

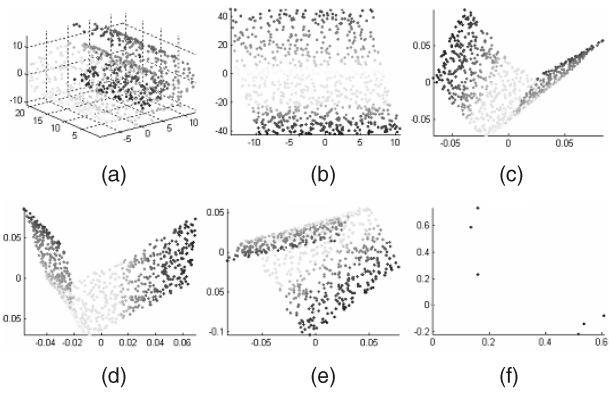


Fig. 10. Comparison between adaptive and fixed neighborhood selection. (a) Swiss roll data. (b) Results of our charting algorithm with adaptive neighborhood selection. (c) Results of LTSA with $k = 8$. (d) Results of LTSA with $k = 7$. (e) Results of LTSA with $k = 6$. (f) Results of LTSA with $k = 5$.

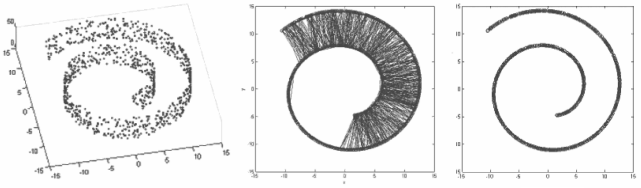
Compared with most existing algorithms, which attempt to map the whole data points into an embedding space simultaneously, our algorithm operates in an incremental-learning fashion. In each step, we compute low-dimensional coordinates only for one data point. This incremental fashion can efficiently solve the problem of out-of-sample extension: A new test point x can be easily mapped after the original coordinate chart has been constructed. First, the neighborhood selection method described in Section 3 can be used to determine the neighbors of x. Second, find the preceding node b (among the neighbors of x) on the shortest path from the base point p to x and then compute the geodesic anchor point a using the abovementioned method. Finally, the normal coordinates of x can be calculated by solving the above least squares problem.

5 Experimental Results

We tested the proposed framework on seven synthetic data sets (Swiss roll, Swiss hole, punctured sphere, twin peaks, 3D clusters, Gaussian, and cylinder) and three face data sets (ISOMAP face data [50], LLE face data [51], and Olivetti Research Laboratory (ORL) face data [52]). The first six synthetic data sets are generated by the *Mani* Matlab demo [53], whereas the cylinder data are sampled from a cylinder with a height of 21 and radius of 30. All experiments were conducted on a 2.80-GHz Pentium IV PC.

5.1 Parameter Estimation

We first investigate the robustness of the proposed adaptive neighborhood selection method on randomly sampled data from a Swiss roll. Our method performs consistently to "unroll" the Swiss roll data correctly. Fig. 10 shows an example that our method succeeds in choosing neighborhood sizes adaptively, whereas other methods (such as LTSA) using a fixed neighborhood size k may result in failure. This example demonstrates an extreme case that any



(a) (b) (c)

Fig. 11. The proposed adaptive neighborhood selection method. (a) Swiss roll data. (b) Visible neighbors in a bird's view. (c) Safe neighbors after removing the short-circuit edges in (b).

TABLE 2
Numbers of Simplices in Each Dimension

Dimension	0	1	2	3	4	5	6	7
Swiss roll	800	1492	1176	0				
Swiss hole	800	1445	1077	0				
Pun. sphere	800	1582	168	0				
Twin peaks	800	1497	1182	0				
3D clusters	800	1359	1029	36	0			
Gaussian	800	1487	1053	0				
Cylinder	800	1441	1005	0				
Isomap face	698	1507	2472	1824	360	0		
LLE face	1965	4268	7551	8232	6420	3960	2520	0
ORL face	400	626	1194	2088	3780	5760	5040	0

fixed neighborhood size k fails: larger k's introduce short-circuit edges, whereas smaller k's segment the data into pieces. Fig. 11 shows the edge connections using our algorithm. As shown in Fig. 11b, there are a large number of short-circuit edges within visible neighbor connections. These short-circuit edges are successfully removed in Fig. 11c and, thus, we obtain "safe" neighbors.

Second, we test the proposed dimension estimation method on these data sets. Table 2 reports the numbers of simplices in each dimension. Recall that the dimension of a complex is the maximal dimension of its simplices. As shown in the table, the dimensions of all synthetic data are correctly estimated: 3D cluster data are 3D, whereas others are intrinsically 2D. Our estimate for the ISOMAP face data is 4, though it is rendered with three parameters (one for lighting and two for pose). Several estimates [54] for this data set are 4.3, 4.0, and 3.5, respectively, which are very close to our estimate. Our estimate for the LLE face data is 6, which is justified by other estimates reported in [55], where different estimates are given as 4.25, 5.63, 5.70, 6.39, and 8.30, respectively.

5.2 Manifold Charting

To evaluate the performance of our RML algorithm, several competing algorithms (PCA, ISOMAP, LLE, Hessian-based LLE (HLLE), Laplacian eigenmaps, diffusion maps, and LTSA) are compared on the seven sets of synthetic data. The objective of this comparison is to map each data set, originally embedded in a 3D space, onto a 2D plane. These synthetic data provide a standard benchmark to evaluate the embedding performance, because both input and output data are low-dimensional and, thus, can be easily visualized.

In the following, we compare the results of each data set (Figs. 12, 13, 14, 15, 16, 17, and 18) in detail:

1. *Swiss roll data in* Fig. 12. RML produces an "ideal" output, showing that both geodesic distances and angles are preserved almost perfectly. PCA fails miserably, since linear projection methods cannot unfold curved structures. ISOMAP attempts to preserve all shortest path distances, but its output is far from satisfactory. In contrast, HLLE and LTSA yield more faithful results than ISOMAP does. However, as the outputs of HLLE and LTSA are compressed into a square region, both scale information and aspect ratio are lost. The other three methods (LLE, Laplacian eigenmaps, and diffusion maps) yield irregular 2D embeddings.
2. *Swiss hole data in* Fig. 13. RML also works perfectly by preserving the geometry around the hole. PCA produces an incorrect mixed point cloud. ISOMAP and LLE yield distorted shapes around the hole. HLLE and LTSA can maintain the shape around the hole but not the aspect ratio and scale information. Laplacian eigenmaps produce a curvelike shape, and diffusion maps output an incorrect mixed point cloud.
3. *Punctured sphere data in* Fig. 14. This data set is sampled from a punctured sphere, rather than a complete sphere. The reason is that a sphere is *not* homeomorphic to a 2D patch. To embed a sphere onto a 2D space, the sphere must be segmented into multiple patches or be punctured. Three algorithms (RML, LLE, and Laplacian eigenmaps) yield good results, even in the boundary area that undergoes large deformation. HLLE and LTSA produce satisfactory results in the central area, but the boundary area shrinks. PCA, ISOMAP, and diffusion maps produce incorrect mixed point clouds.

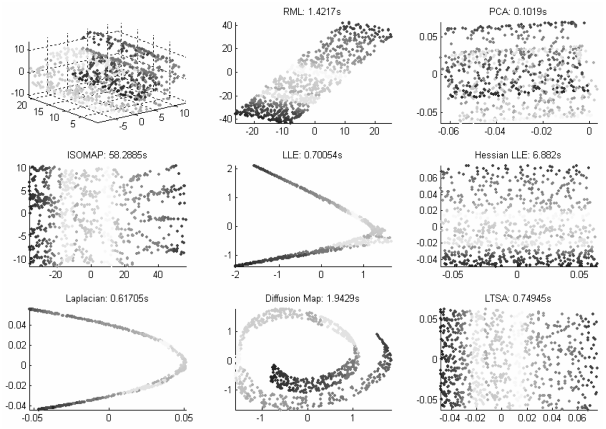


Fig. 12. Comparison results of Swiss roll data.

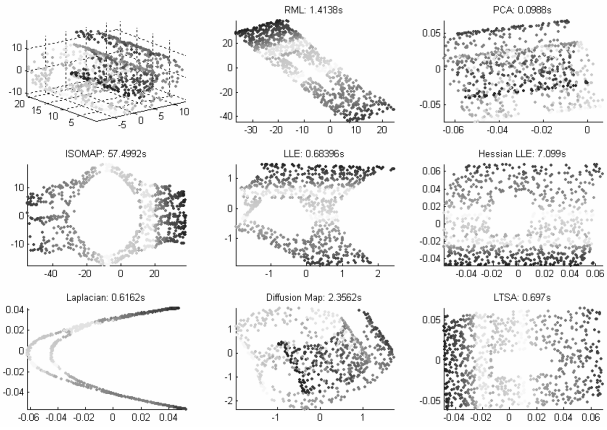


Fig. 13. Comparison results of Swiss hole data.

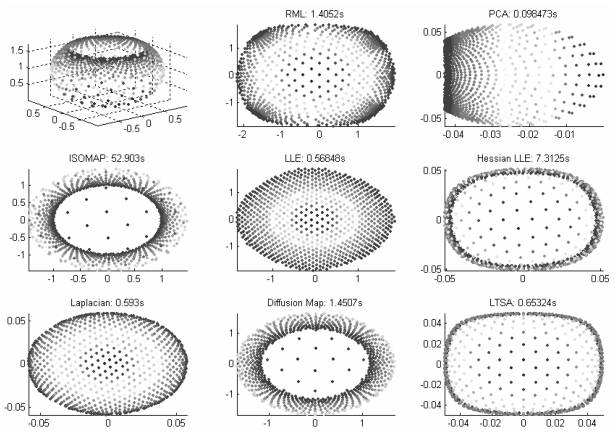


Fig. 14. Comparison results of punctured sphere data.

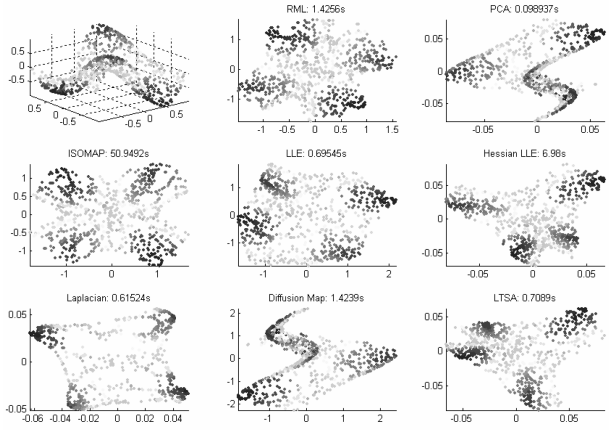


Fig. 15. Comparison results of twin peaks data.

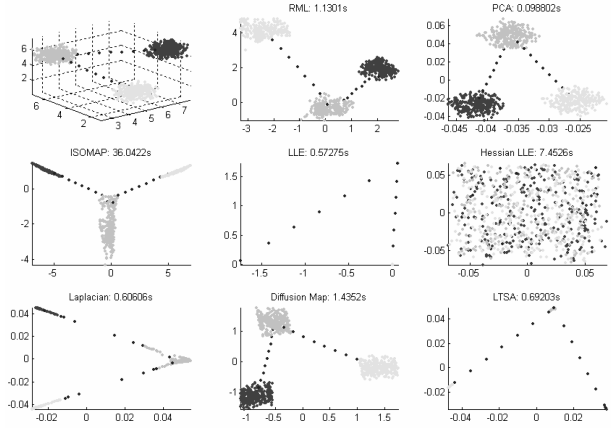


Fig. 16. Comparison results of 3D clusters data.

4. *Twin peaks data in* Fig. 15. Most algorithms produce satisfactory results, except that PCA and diffusion maps yield incorrect mixed point clouds. In the output of Laplacian eigenmaps, data points appear to aggregate around the four corners, yielding a sparse distribution in the central area.
5. *Three-dimensional clusters data in* Fig. 16. RML, PCA, and diffusion map can yield satisfactory results, in which the shapes of the three clusters are preserved and the global connection is maintained. Other algorithms (except HLLE) can maintain the global connection, but the shapes of the three clusters are degenerated. HLLE produces incorrect mixed point clouds.
6. *Gaussian data in* Fig. 17. Most algorithms (including PCA) can yield satisfactory results. Again, HLLE produces incorrect mixed point clouds. In the result of Laplacian eigenmaps, data points appear to aggregate around the outer circular boundary, and in other areas, the data points are very dispersed.
7. *Cylinder data in* Fig. 18. RML cuts the cylinder along one generatrix line and unroll it to form a long stripe. Four methods (PCA, ISOMAP, Laplacian eigenmaps, and diffusion maps) project the cylinder data along the direction of generatrix lines and yield a closed circle. The other three methods (LLE, HLLE, and LTSA) produce incorrect mixed point clouds.

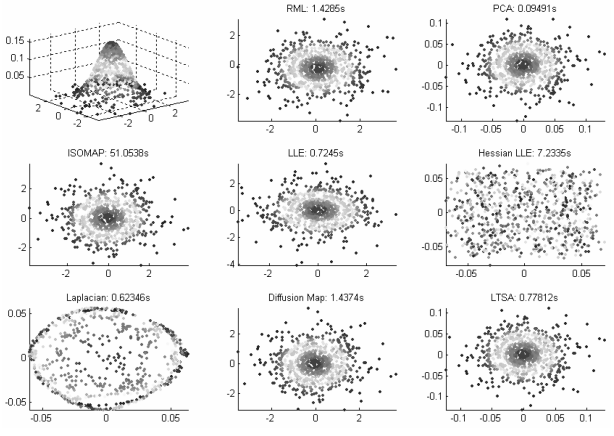


Fig. 17. Comparison results of Gaussian data.

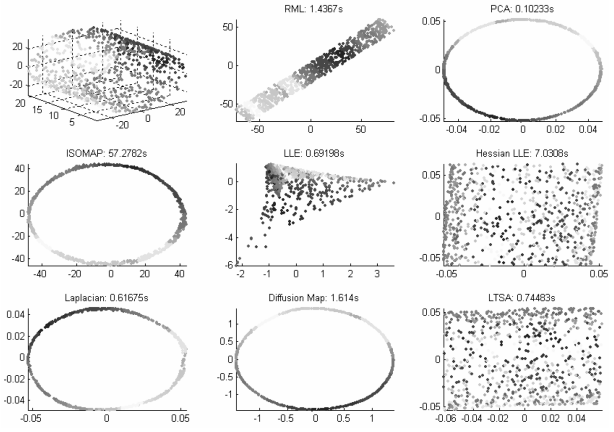


Fig. 18. Comparison results of cylinder data.

The average runtime of RML (including parameter estimation) is about 1.4 seconds, which is approximately two times of that of LLE, Laplacian eigenmaps, and LTSA. Often, HLLE and diffusion maps spend several seconds, whereas ISOMAP needs about 1 minute.

To evaluate the proposed RML algorithm on real-world data, three face data sets (ISOMAP face data [50], LLE face data [51], and ORL face data [52]) are used to perform dimensionality reduction. The objective is to embed each original high-dimensional data into a 3D space, in which data point distributions can be visualized. We present the results in detail:

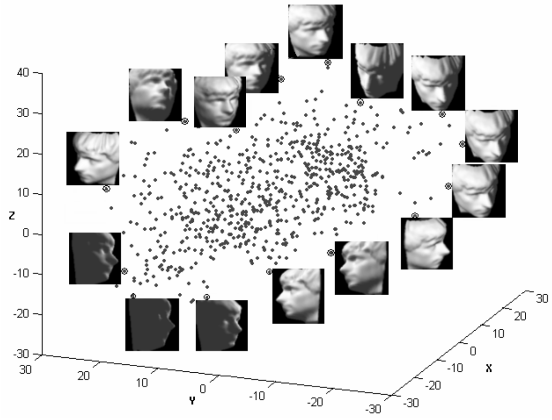


Fig. 19. Three-dimensional embedding of ISOMAP face data using RML.

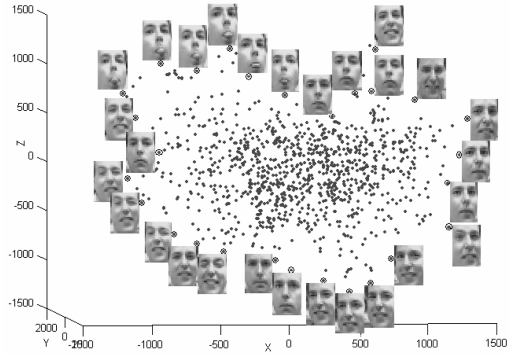


Fig. 20. Three-dimensional embedding of LLE face data using RML.

1. *ISOMAP face data in* Fig. 19. In this 3D representation, data points are uniformly distributed and well organized. Representative face images are shown next to circled points. Note that because this figure can only display a 2D projection of the 3D space; nearby points in this projection may not correspond to nearby points in the 3D space. Despite this projection, the representative face images still show several particular modes of variation in pose and lighting condition.
2. *LLE face data in* Fig. 20. A 2D projection of the 3D embedding representation is shown, and a set of representative face images are superimposed on data points. The uniformly distributed point cloud implements a perceptually natural 3D embedding of the face image data. Although we cannot directly examine the similarity among each local region, the representative faces reveal several groups of similar facial expressions (smile, wink, and so on). The same LLE face data were previously tested using LLE [16], Laplacian eigenmaps [19], SDE [21], and conformal eigenmaps [26]. In comparison, these spectral methods often tend to produce irregular embeddings that are not easy to understand and interpret.
3. *ORL face data in* Fig. 21. The goal of this experiment is to show a 3D embedding of a set of face images of multiple persons, which is important in applying manifold learning algorithms to face recognition. The 3D embedding of this data is similar to that of the above two data sets, recovering a well-organized underlying geometric structure of the original data. It is interesting to observe that the face images of a single person appear to be sampled from a curve, and often, several curves twist together. These twisted curves reveal a certain degree of interperson similarity, which makes face recognition difficult.

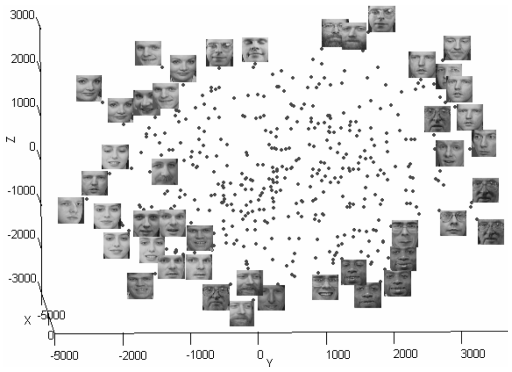


Fig. 21. Three-dimensional embedding of ORL face data using RML.

In summary, the proposed RML algorithm outperforms most existing methods in the visualization task on seven synthetic data sets and also achieves satisfactory embedding results on three real-world data sets.

5.3 Discussion

In our experiments, a large number of synthetic and real-world data sets are used to compare several major manifold learning algorithms, which is more comprehensive and more objective than most work in the literature. It is *not* our intention to convince the reader that the proposed RML algorithm offers an optimal solution to any dimensionality reduction problem. In fact, each algorithm is derived from a different motivation and has its own strength and weakness. This poses an important question to any practitioner who attempts to use manifold learning algorithms: "How does one know which method to pick?"

From our experience, we list several important factors that should be considered:

1. *Data sets.* First, try to learn more about the data set at hand: sample size, input dimension, intrinsic dimension, noise level, sampling density, and so on. For instance, ISOMAP fails when the underlying manifold is not isomorphic to a convex region of a euclidean space [56].
2. *Applications.* Until now, there is no common criterion to evaluate and compare the performance of different manifold learning algorithms. One can define his or her own criterion to fit the application best. In the literature, two major applications are data visualization and pattern classification. For instance, incremental learning is essential to classification. If this capability is not easily available in certain methods, do not use them for classification purposes.
3. *Two parameters, k and d.* Most algorithms require these two parameters as inputs. One can use the method described in Section 3 or other similar methods to estimate them beforehand.
4. *Computational complexity.* A heavy computational load often hinders one method to be practically used. Particularly, for a data set of thousands of samples in a high-dimensional space, this problem becomes more critical.
5. *Robustness to sparse sampling and noise.* As few theoretical results are reported from this aspect, one tractable way is to examine the robustness by using

some low-dimensional test data sets, for example, the data sets provided by the Mani demo [53].

6. *Empirical examination on real-world data sets.* These large-scale tests facilitate us to observe, identify, and explain the limitations of different manifold learning algorithms.

6 Computational Complexity

In this section, we provide an analysis of the computational complexity of each step as a function of the number of samples N, the input dimension n, the intrinsic dimension d, and other related factors if necessary.

Adaptive neighborhood selection. For a given point x, computing the distances needs $O(Nn)$ calculation, and sorting to get its K-NNs takes $O(KN)$. Thus, for all N points, finding K-NNs is $O(N^2(n+K))$.

The next step is to find visible neighbors. For a given point x, all pairs of its neighbors should be examined. The complexity is $O(NK^2)$ for all points.

The final step is to remove short-circuit edges. Note that the size of visible neighbors for one point is not larger than K. The complexity is $O(NK)$ for all points if a simple threshold method is used to detect short-circuit edges. When PCA is used to find out where the dimension increases, an extra cost is to compute eigenvalues for a covariance matrix. Given an $n \times n$ real symmetric matrix, the complexity is $O(2n^3/3 + 30n^2)$ for eigenvalues only, by using the most typical combination of Householder reduction and tridiagonal QL implicit (routines $tred2$ and $tqli$ in [57, Chapter 11]). When n is large, computing in this way is demanding. Let an $n \times k$ matrix X be the data. Alternatively, one can compute the eigenvalues of a $k \times k$ matrix X^TX, which is much faster than handling the covariance matrix XX^T. This is because $XX^T = US^2U^T$, and $X^TX = VS^2V^T$, by using a singular value decomposition (SVD) $X = USV^T$. Compared with the large n, k is often small. Therefore, the complexity $O(2k^3/3 + 30k^2)$ can be negligible.

Intrinsic dimension estimation. Given a simplex, say, (a, b, c), the goal is to find the higher dimensional simplex (a, b, c, d). We only need to examine if d has been a neighbor to any other point. Thus, d can be obtained by computing the common intersection of three neighbor sets of a, b, and c. Assume that the average size of neighbors is k. The complexity is $O(k^2)$ for comparing any two sets. However, the intersection set will become smaller with more intersection operations, which can save comparisons. In practice, the total time of dimension estimation is related to the number of simplices in each dimension.

Base point selection. For a given candidate point x, most time is spent on running Dijkstra's algorithm to find the maximal path length. Given a graph $G = (V, E)$, the complexity is $O(V^2 + E)$ with a simple min-priority queue, $O((V + E)\log V)$ with a binary min-heap, or even $O(V \log V + E)$ with a Fibonacci heap [47], [48]. Here, V, the number of vertices, is equal to N. Therefore, the total time is $O(N(V \log V + E)) = O(N^2 \log N + NE))$ with a Fibonacci heap, when iterating through all data points.

PCA projection. Suppose this is performed upon a local region that consists of points $\{x_0, x_1, \ldots, x_k\}(k > d)$. Let the $n \times k$ data matrix be $X = [x_1, \ldots, x_k]$. We need to compute the unit eigenvectors corresponding to the d largest eigenvalues of the $n \times n$ covariance matrix. The complexity is $O(4n^3/3 + 3n^3)$ (see [57]). When n is larger than k, alternatively, we can use the eigendecomposion of the $k \times k$ matrix X^TX. Let $Y = U^TX = SV^T$ if $X = USV^T$. We have $X^TX = VS^2V^T$. Then, we can obtain the PCA projection $Y = SV^T$ by using the first d eigenvalues and eigenvectors. Thus, the time is reduced to $O(4k^3/3 + 3k^3)$, which is negligible if k is small.

Anchor point calculation. The two-order polynomial fitting is to solve a least squares problem $\min \|Ax - b\|$, where A is a $k \times m$ matrix, $k > m$, $m = (d+2)(d+1)/2$, and b is an $m \times 1$ vector. Most time is spent on matrix multiplication and inversion. The next step is to compute the distance from one point y to a general quadratic manifold $Q(x) = x^TAx + b^Tx + c = 0$, where A is a symmetric $d \times d$ matrix, b is a $d \times 1$ vector, and c is a scalar. In this step, most time is spent on the eigendecomposition of A and finding the roots of a polynomial (see Appendix B).

Quadratically constrained least squares problem (13). Most time is spent on the SVD of a $k \times d$ matrix A (where $k > d$) and the calculation of $x = (A^TA + \lambda I)^{-1}A^Tb$. In fact, we only need to compute the eigendecomposition of AA^T to get U and S, provided that $A = USV^T$, since V does not occur in the solutions. Note that using Newton's method to solve the Lagrange multiplier λ is independent of N, n, and d.

Overall complexity of the entire framework. When N is large (which is the usual case in manifold learning applications), the complexity of neighbor selection, dimension estimation, and base point selection dominates. For real-world applications, the input dimension n is often large too. Compared with the large N and n, other factors (such as d, k, and m) are rather small in typical settings. Due to this fact, the normal coordinates of each data point can still be efficiently computed, in which the PCA projection, anchor point calculation, and least squares solution are the principle workload.

7 Conclusion

We have presented a general framework called RML for nonlinear dimensionality reduction, which can learn the intrinsic geometry of the manifold with metric preserving properties. Experimental results demonstrate the excellent performance of our algorithm on both synthetic and real data sets. Manifold learning is closely related to several significant themes of the mathematics in the last century [58], such as from linear to nonlinear, from low-dimensional to high-dimensional, and from local to global. Our work translates the concept of Riemannian normal coordinates onto a cloud of unorganized data points. Future work may include the development of new tools to learn more topological and geometrical properties of the underlying manifold, which may not be applied to dimensionality reduction only.

Appendix A
Quadratically Constrained Least Squares Problem

In [59], two approaches are provided to solve the linear least squares problem $\min_x \|Ax - b\|^2$ with a quadratic constraint $\|x\| = \alpha$. One approach uses Lagrange multipliers by constructing the following potential function:

$$\begin{aligned}\phi(x,\lambda) &= \|b - Ax\|_2^2 + \lambda(\|x\|_2^2 - \alpha^2) \\ &= (b^T - x^T A^T)(b - Ax) + \lambda(x^T x - \alpha^2).\end{aligned}$$

Setting the gradient of this function with respect to x equal to zero yields the following equation:

$$\frac{\partial \phi}{\partial x} = -2A^T b + 2A^T Ax + 2\lambda x = 0,$$

which has the solution $x = (A^T A + \lambda I)^{-1} A^T b$, provided that the inverse of $(A^T A + \lambda I)$ exists. Substituting this result into the constraint $\|x\|_2^2 = \alpha^2$, we have

$$\psi(\lambda) = b^T A(A^T A + \lambda I)^{-2} A^T b - \alpha^2 = 0.$$

Let $A = U\Sigma V^T$ be the SVD of A. Then, our constraint equation becomes

$$\begin{aligned}\psi(\lambda) &= b^T U\Sigma V^T (V\Sigma^T U^T U\Sigma V^T + \lambda I)^{-2} V\Sigma^T U^T b - \alpha^2 \\ &= b^T U\Sigma V^T (V(\Sigma^T\Sigma + \lambda I)V^T)^{-2} V\Sigma^T U^T b - \alpha^2 \\ &= b^T U\Sigma V^T (V(\Sigma^T\Sigma + \lambda I)V^T \\ &\quad V(\Sigma^T\Sigma + \lambda I)V^T)^{-1} V\Sigma^T U^T b - \alpha^2 \\ &= b^T U\Sigma(\Sigma^T\Sigma + \lambda I)^{-2}\Sigma^T U^T b - \alpha^2 = 0.\end{aligned}$$

Letting $\beta = U^T b$, we get

$$\psi(\lambda) = \sum_{i=1}^{d} \frac{\beta_i^2 \sigma_i^2}{(\sigma_i^2 + \lambda)^2} - \alpha^2 = 0.$$

Note that $\psi(\lambda)$ decreases from ∞ to $-\alpha^2$ as λ goes from $-\sigma_d^2$ to ∞. We can use Newton's method to find the root λ. A good initial value for λ is zero, and the objective function vanishes to zero very fast.

APPENDIX B

DISTANCE FROM ONE POINT TO A GENERAL QUADRATIC MANIFOLD

For the completeness of the paper, we briefly summarize the algorithm in [60] for computing the distance from one point to a general quadratic manifold (a quadratic curve or a quadratic surface in low dimensions), defined implicitly by the following quadratic equation:

$$Q(x) = x^T Ax + b^T x + c = 0,$$

where A is a symmetric $d \times d$ matrix, b is a $d \times 1$ vector, and c is a scalar. The parameter is x, a $d \times 1$ vector. Given the manifold $Q(x) = 0$ and a point y, find the distance from y to the manifold and compute the closest point x on the manifold to y. Geometrically, the closest point x must satisfy the condition that $y - x$ is normal to the manifold. That is, $y - x = t \cdot \nabla Q(x) = t(2Ax + b)$ for some scalar t, since the gradient is normal to the manifold. Thus, we have $x = (I + 2tA)^{-1}(y - tb)$, where I is the identity matrix. Factoring A into an eigendecomposition $A = RDR^T$ yields

$$\begin{aligned}x &= (I + 2tA)^{-1}(y - tb) \\ &= (RR^T + 2tRDR^T)^{-1}(y - tb) \\ &= [R(I + 2tD)R^T]^{-1}(y - tb) \\ &= R(I + 2tD)^{-1}R^T(y - tb) \\ &= R(I + 2tD)^{-1}(\alpha - t\beta),\end{aligned}$$

where the last equation defines α and β. Replacing x in the quadratic equation yields

$$\begin{aligned}&(\alpha - t\beta)^T(I + 2tD)^{-1}D(I + 2tD)^{-1}(\alpha - t\beta) \\ &\quad + \beta^T(I + 2tD)^{-1}(\alpha - t\beta) + c = 0.\end{aligned}$$

The inverse diagonal matrix is

$$(I + 2tD)^{-1} = diag\{1/(1 + 2t\lambda_1), \ldots, 1/(1 + 2t\lambda_d)\},$$

where $D = diag\{\lambda_1, \ldots, \lambda_d\}$. Multiplying through by $\prod_{i=1}^{d}(1 + 2t\lambda_i)^2$ leads to a polynomial of at most order $2d$. The roots of the polynomial are computed, and we get $x = (I + 2tA)^{-1}(y - tb)$ for each root t. Finally, choose the x that achieves the minimal distance to y.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their helpful suggestions. This work was partially supported by the National Science Foundation of China (NSFC) Grants 60302005 and 60333010, National Key Basic Research Program (NKBRP) Grant 2004CB318005, and Foundation for the Author of National Excellent Doctoral Dissertation of the People's Republic of China (FANEDD) Grant 200038.

REFERENCES

[1] D.L. Donoho, "High-Dimensional Data Analysis: The Curses and Blessings of Dimensionality," *Proc. AMS Math. Challenges of the 21st Century,* 2000.
[2] B. Nadler, S. Lafon, R.R. Coifman, and I.G. Kevrekidis, "Diffusion Maps, Spectral Clustering and Reaction Coordinates of Dynamical Systems," *Applied and Computational Harmonic Analysis,* vol. 21, pp. 113-127, 2006.
[3] I.T. Jolliffe, *Principal Component Analysis.* Springer, 1989.
[4] M. Turk and A. Pentland, "Eigenfaces for Recognition," *J. Cognitive Neuroscience,* vol. 3, no. 1, pp. 71-86, 1991.
[5] T. Cox and M. Cox, *Multidimensional Scaling.* Chapman and Hall, 1994.
[6] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification,* second ed. John Wiley & Sons, 2001.
[7] T. Kohonen, *Self-Organizing Maps,* third ed. Springer, 2001.
[8] B. Kégl, A. Krzyzak, T. Linder, and K. Zeger, "Learning and Design of Principal Curves," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 3, pp. 281-297, Mar. 2000.
[9] A.J. Smola, S. Mika, B. Schölkopf, and R.C. Williamson, "Regularized Principal Manifolds," *J. Machine Learning Research,* vol. 1, pp. 179-209, June 2001.
[10] P. Baldi and K. Hornik, "Neural Networks and Principal Component Analysis: Learning from Examples without Local Minima," *Neural Networks,* vol. 2, no. 1, pp. 53-58, 1989.
[11] C.M. Bishop, M. Svensen, and C.K.I. Williams, "GTM: The Generative Topographic Mapping," *Neural Computation,* vol. 10, no. 1, pp. 215-234, 1998.
[12] J. Yang, A.F. Frangi, J.-Y. Yang, D. Zhang, and Z. Jin, "KPCA plus LDA: A Complete Kernel Fisher Discriminant Framework for Feature Extraction and Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 2, pp. 230-244, Feb. 2005.
[13] J. Tenenbaum, V. de Silva, and J. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science,* vol. 290, pp. 2319-2323, Dec. 2000.
[14] V. de Silva and J. Tenenbaum, "Global versus Local Methods in Nonlinear Dimensionality Reduction," *Proc. Advances in Neural Information Processing Systems,* vol. 15, pp. 705-712, 2003.
[15] M.H. Law and A.K. Jain, "Incremental Nonlinear Dimensionality Reduction by Manifold Learning," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 3, pp. 377-391, Mar. 2006.
[16] S. Roweis and L. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science,* vol. 290, pp. 2323-2326, Dec. 2000.

[17] L.K. Saul and S.T. Roweis, "Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifold," *J. Machine Learning Research,* vol. 4, pp. 119-155, 2003.
[18] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," *Neural Computation,* vol. 15, no. 6, pp. 1373-1396, 2003.
[19] X. He, S. Yan, Y. Hu, P. Niyogi, and H.J. Zhang, "Face Recognition Using Laplacianfaces," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 3, pp. 328-340, Mar. 2005.
[20] D. Donoho and C. Grimes, "Hessian Eigenmaps: New Locally Linear Embedding Techniques for High-Dimensional Data," *Proc. Nat'l Academy of Sciences,* vol. 100, no. 10, pp. 5591-5596, 2003.
[21] K. Weinberger and L. Saul, "Unsupervised Learning of Image Manifolds by Semidefinite Programming," *Proc. IEEE Int'l Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 988-995, 2004.
[22] M. Brand, "Charting a Manifold," *Proc. Advances in Neural Information Processing Systems,* vol. 15, pp. 961-968, 2003.
[23] Z. Zhang and H. Zha, "Principal Manifolds and Nonlinear Dimension Reduction via Local Tangent Space Alignment," *SIAM J. Scientific Computing,* vol. 26, no. 1, pp. 313-338, 2005.
[24] R.R. Coifman and S. Lafon, "Diffusion Maps," *Applied and Computational Harmonic Analysis,* vol. 21, pp. 5-30, July 2006.
[25] S. Lafon and A.B. Lee, "Diffusion Maps and Coarse-Graining: A Unified Framework for Dimensionality Reduction, Graph Partitioning and Data Set Parameterization," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 9, pp. 1393-1403, Sept. 2006.
[26] F. Sha and L.K. Saul, "Analysis and Extension of Spectral Methods for Nonlinear Dimensionality Reduction," *Proc. 22nd Int'l Conf. Machine Learning,* pp. 785-792, 2005.
[27] H.S. Seung and D.D. Lee, "The Manifold Ways of Perception," *Science,* vol. 290, pp. 2268-2269, Dec. 2000.
[28] B. Riemann, "On the Hypotheses Which Lie at the Bases of Geometry," *Nature,* vol. 8, pp. 14-17, pp 36-37 1873, translated by W.K. Clifford.
[29] Y. Bengio, J.F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet, "Out-of-Sample Extensions for LLE, ISOMAP, MDS, Eigenmaps, and Spectral Clustering," *Proc. Advances in Neural Information Processing Systems 16,* pp. 177-184, 2003.
[30] R.R. Coifman and S. Lafon, "Geometric Harmonics: A Novel Tool for Multiscale Out-of-Sample Extension of Empirical Functions," *Applied and Computational Harmonic Analysis,* vol. 21, no. 1, pp. 31-52, July 2006.
[31] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral Grouping Using the Nystrom Method," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 2, pp. 214-225, Feb. 2004.
[32] A. Brun, C.F. Westin, M. Herberthson, and H. Knutsson, "Fast Manifold Learning Based on Riemannian Normal Coordinates," *Proc. 14th Scandinavian Conf. Image Analysis,* pp. 920-929, 2005.
[33] T. Lin, H. Zha, and S. Lee, "Riemannian Manifold Learning for Nonlinear Dimensionality Reduction," *Proc. Ninth European Conf. Computer Vision,* pp. 44-55, May 2006.
[34] J.M. Lee, *Riemannian Manifolds: An Introduction to Curvature.* Springer, 1997.
[35] J. Jost, *Riemannian Geometry and Geometric Analysis,* third ed. Springer, 2002.
[36] M. Balasubramanian, E.L. Schwartz, J.B. Tenenbaum, V. de Silva, and J.C. Langford, "The Isomap Algorithm and Topological Stability," *Science,* vol. 295, p. 7a, Jan. 2002.
[37] J. Wang, Z. Zhang, and H. Zha, "Adaptive Manifold Learning," *Proc. Advances in Neural Information Processing Systems,* vol. 17, pp. 1473-1480, 2005.
[38] F. Camastra, "Data Dimensionality Estimation Methods: A Survey," *Pattern Recognition,* vol. 36, no. 12, pp. 2945-2954, Dec. 2003.
[39] K. Fukunaga and D.R. Olsen, "An Algorithm for Finding Intrinsic Dimensionality of Data," *IEEE Trans. Computers,* vol. 20, no. 2, pp. 165-171, Feb. 1976.
[40] J. Bruske and G. Sommer, "Intrinsic Dimensionality Estimation with Optimally Topology Preserving Maps," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 5, pp. 572-575, May 1998.
[41] G.V. Trunk, "Statistical Estimation of the Intrinsic Dimensionality of a Noisy Signal Collection," *IEEE Trans. Computers,* vol. 25, no. 2, pp. 165-171, Feb. 1976.
[42] K. Pettis, T. Bailey, T. Jain, and R. Dubes, "An Intrinsic Dimensionality Estimator from Near-Neighbor Information," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 1, no. 1, pp. 25-37, 1979.
[43] P.J. Verveer and R. Duin, "An Evaluation of Intrinsic Dimensionality Estimators," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 17, no. 1, pp. 81-86, Jan. 1995.
[44] J.A. Costa and A.O. Hero, "Geodesic Entropic Graphs for Dimension and Entropy Estimation in Manifold Learning," *IEEE Trans. Signal Processing,* vol. 52, no. 8, pp. 2210-2221, Aug. 2004.
[45] F. Camastra and A. Vinciarelli, "Estimating the Intrinsic Dimension of Data with a Fractal-Based Method," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 10, pp. 1404-1407, Oct. 2002.
[46] D. Freedman, "Efficient Simplicial Reconstructions of Manifolds from Their Samples," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 10, pp. 1349-1357, Oct. 2002.
[47] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms,* second ed. MIT Press, 2001.
[48] http://en.wikipedia.org/wiki/Dijkstra's_algorithm, July 2005.
[49] D. Eberly, *Computing Geodesics on a Riemannian Manifold,* http://www.geometrictools.com/Documentation/RiemannianGeodesics.pdf, Sept. 2005.
[50] http://isomap.stanford.edu/face_data.mat.Z, 2008.
[51] http://www.cs.toronto.edu/~roweis/data/frey_rawface.mat, 2008.
[52] http://www.cs.toronto.edu/~roweis/data/olivettifaces.mat, 2008.
[53] T. Wittman, *Mani Matlab Demo,* http://www.math.umn.edu/~wittman/mani/, 2005.
[54] E. Levina and P. Bickel, "Maximum Likelihood Estimation of Intrinsic Dimension," *Proc. Advances in Neural Information Processing Systems,* vol. 17, pp. 777-784, 2005.
[55] M. Raginsky and S. Lazebnik, "Estimation of Intrinsic Dimensionality Using High-Rate Vector Quantization," *Proc. Advances in Neural Information Processing Systems,* Dec. 2005.
[56] D.L. Donoho and C. Grimes, "When Does ISOMAP Recover the Natural Parameterization of Families of Articulated Images?" Technical Report 2002-27, Dept. of Statistics, Stanford Univ., 2002.
[57] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing,* second ed. Cambridge Univ. Press, 1992.
[58] M. Atiyah, "Mathematics in the 20th Century," *Bull. London Math. Soc.,* vol. 34, pp. 1-15, 2002.
[59] G.H. Golub, "Constrained Least Squares and Introduction to Lanczos Method," Course Notes on Matrix Computation, http://www.mat.uniroma1.it/~bertaccini/seminars/CS339/syllabus.html, 2004.
[60] D. Eberly, "Distance from Point to a General Quadratic Curve or a General Quadratic Surface," http://www.geometrictools.com/Documentation/DistancePointToQuadratic.pdf, Mar. 1999.

Tong Lin received the PhD degree in applied mathematics from Peking University, Beijing, in 2001. In 1999 and 2000, he was a visiting student at Microsoft Research Asia, Beijing. In 2002, he joined the State Key Laboratory of Machine Perception, Peking University, where he is currently an associate professor. From 2004 to 2005, he was an exchange scholar at Seoul National University, Korea. His research interests include wavelet analysis, computer vision, pattern recognition, and machine learning.

Hongbin Zha received the BE degree in electrical engineering from the Hefei University of Technology, China, in 1983 and the MS and PhD degrees in electrical engineering from Kyushu University, Japan, in 1987 and 1990, respectively. After working as a research associate at Kyushu Institute of Technology, he joined Kyushu University in 1991 as an associate professor. He was also a visiting professor in the Centre for Vision, Speech, and Signal Processing, Surrey University, Unite Kingdom, in 1999. Since 2000, he has been a professor at the State Key Laboratory of Machine Perception, Peking University, China. His research interests include computer vision, digital geometry processing, and robotics. He has published more than 160 technical publications in journals, books, and international conference proceedings. He received the Franklin V. Taylor Award from the IEEE Systems, Man, and Cybernetics Society in 1999.