

Riemannian Manifold Clustering and Dimensionality Reduction for Vision-Based Analysis

Alvina Goh

Abstract Segmentation is one fundamental aspect of vision-based motion analysis, thus it has been extensively studied. Its goal is to group the data into clusters based upon image properties such as intensity, color, texture, or motion. Most existing segmentation algorithms proceed by associating a feature vector to each pixel in the image or video and then segmenting the data by clustering these feature vectors. This process can be phrased as a manifold learning and clustering problem, where the objective is to learn a low-dimensional representation of the underlying data structure and to segment the data points into different groups. Over the past few years, various techniques have been developed for learning a low-dimensional representation of a nonlinear manifold embedded in a high-dimensional space. Unfortunately, most of these techniques are limited to the analysis of a single connected nonlinear manifold. In addition, all these manifold learning algorithms assume that the feature vectors are embedded in a Euclidean space and make use of (at least locally) the Euclidean metric or a variation of it to perform dimensionality reduction. While this may be appropriate in some cases, there are several computer vision problems where it is more natural to consider features that live in a Riemannian space. To address these problems, algorithms for performing simultaneous nonlinear dimensionality reduction and clustering of data sampled from multiple submanifolds of a Riemannian manifold have been recently proposed. In this book chapter, we give a summary of these newly developed algorithms as described in Goh and Vidal (Conference on Computer Vision and Pattern Recognition, 2007 and 2008; European Conference on Machine Learning, 2008; and European Conference on Computer Vision, 2008) and demonstrate their applications to vision-based analysis.

A. Goh (✉)

National University of Singapore, Singapore, Singapore
e-mail: alvinagoh@gmail.com

1 Introduction

Nonlinear dimensionality reduction (NLDR) refers to the problem of finding a low-dimensional representation for a set of points lying in a nonlinear manifold embedded in a high-dimensional space. This question of how to detect and represent low-dimensional structure in high-dimensional data is fundamental to many disciplines and several attempts have been made in different areas to address this question. For example, the number of pixels in an image can be rather large, yet most computer vision models use only a few parameters to describe the geometry, photometry and dynamics of the scene. Since most datasets often have fewer degrees of freedom than the dimension of the ambient space, NLDR is fundamental to many problems in computer vision, machine learning, and pattern recognition.

When the data lives in a low-dimensional linear subspace of a high-dimensional space, simple linear methods such as Principal Component Analysis (PCA) [26] and metric Multi-Dimensional Scaling (MDS) [11] can be used to learn the subspace and its dimension. However, when the data lies in a low-dimensional submanifold, its structure may be highly nonlinear, hence linear dimensionality reduction methods are likely to fail. This has motivated extensive efforts toward developing NLDR algorithms for computing low-dimensional embeddings.

A huge family of such algorithms computes a low-dimensional representation from the eigenvectors of a matrix constructed from the local geometry of the manifold. Such algorithms include ISOMAP [42], Kernel PCA (KPCA) [38], locally linear embedding (LLE) [35], and its variants such as Laplacian Eigenmaps (LE) [5], Hessian LLE [14], Local Tangent Space Alignment (LTSA) [50], maximum variance unfolding [47], and conformal eigenmaps [39]. A recent survey of many of these algorithms can be found in [8]. Most of these NLDR techniques can be categorized into two main groups: global and local techniques.

Global techniques attempt to preserve global properties of the data lying in a submanifold, similar to what PCA attempts to preserve for data lying in a linear subspace. In addition, they are also capable of constructing a nonlinear transformation between the high-dimensional data and the low-dimensional representation. Two of the best-known examples of this family of algorithms are ISOMAP and KPCA. ISOMAP attempts to preserve the geodesic distance between two data points on a manifold by approximating it as the length of the shortest path in the graph connecting the two points. KPCA reformulates linear PCA in a high-dimensional space via the kernel trick. Rather than considering the covariance matrix, KPCA computes the principal components of the kernel matrix. The kernel function allows KPCA to construct nonlinear mappings from the high-dimensional space to the low-dimensional space.

Local techniques are based on the preservation of local properties obtained from small neighborhoods around the data points. The key idea of such techniques is that by preserving local properties of the data, one can also retain global properties of the data. LLE, LE, Hessian LLE and LTSA fall under this category of algorithms. It has been proven that LLE is a special form of kernel principal component analysis

(KPCA) [23]. However, unlike conventional KPCA where one defines a kernel function in order to map the input space to a higher dimensional feature space, deriving the analytic form for the LLE kernel is not straightforward.

Although the goals of dimensionality reduction, classification and segmentation have always been intertwined with each other, considerably less work has been done on extending NLDR techniques for the purpose of clustering data living on different manifolds. For linear manifolds, there are many existing subspace clustering methods including K-subspaces [25], Local Subspace Affinity [49], Mixtures of Probabilistic PCA (MPPCA) [43], Generalized Principal Component Analysis (GPCA) [45], Agglomerative Lossy Compression (ALC) [48], and spectral clustering [1, 10, 22]. K-subspaces [25] proceeds similarly to K -means: it is initialized with a collection of K subspace bases of dimension d , and then it alternates between assigning points to their nearest subspace, and computing a subspace that minimizes the sum-of-the-squares distance to all points in each cluster. MPPCA [43] applies Expectation Maximization (EM) to a mixture of probabilistic PCAs. It assumes that the distribution of the data inside each subspace is Gaussian and uses EM to learn the parameters of the mixture model. GPCA [45] is an algebraic solution to subspace clustering based on fitting a union of m subspaces with a polynomial of degree m . The gradient of this polynomial at a point gives a vector normal to the subspace containing that point. The subspace clustering problem is then equivalent to fitting and differentiating a set of homogeneous polynomials. ALC [48] models each subspace with a degenerate Gaussian, and iteratively merges pairs of points so as to minimize the coding length needed to encode these points with a mixture of Gaussians. The multi-way spectral clustering algorithm [10] applies spectral clustering to a multi-way similarity that captures the curvature of a collection of points within an affine subspace.

All the aforementioned subspace clustering methods are formulated specifically for mixtures of linear manifolds, and thus they do not work in the presence of nonlinear manifolds. Existing works that extend NLDR techniques to clustering nonlinear manifolds include [7, 33, 40]. The work of [40] develops an EM-like extension of ISOMAP for clustering multiple nonlinear manifolds. However, this method is very sensitive to good initialization and is not a principled EM method as it uses heuristics in the E-step to assign points to manifolds. The work of [33] applies LLE to a manifold with m connected components. It shows that m eigenvalues of the matrix M are zero and that the clustering of the data can be obtained from the corresponding eigenvectors. However, this LLE clustering algorithm suffers from degeneracies in the presence of linear subspaces. In [7], the authors proved that in spectral clustering, it is possible to obtain a lower dimensional hypersphere representation via an eigendecomposition of the affinity matrix. In particular, [7] shows that it is possible to find a low-dimensional embedding in spaces having a mixture of linear and cyclic axes, and to cluster the data by repeated projection. However, the embedding algorithm maps the data only to a mixed vector and toric space, with the linear or cyclic nature of each axis determined from statistical tests. Also, the clustering is done via an iterative method that requires several projections.

A significant amount of work [4, 17, 24, 29, 31] has also been done on clustering data according to the dimensionality of the manifolds that contain the data

rather than the manifolds themselves. For example, in human activity recognition, the videos describing different activities such as walking, jumping, and running can be described with a different number of parameters. Barbará and Chen [4] proposes a new clustering algorithm that is based on the use of the fractal dimension and clusters the data so that points in the same cluster are more self-affine among themselves than points in different clusters. The dimension of a manifold is estimated using a tensor voting scheme [31]. In [17], the local correlation dimension and density of a point is estimated and used as the input to standard clustering techniques. In [29], a maximum likelihood estimator of the intrinsic dimension of a dataset is derived by a Poisson process approximation. This idea is extended in [24] by modeling the high-dimensional sample points as a mixtures of Poisson processes, with regularizing restrictions and spatial continuity constraints. By proceeding in a EM-like manner, it is shown that it is possible to simultaneously estimate the soft clustering and the intrinsic dimension and density of each cluster. Even though such methods have proven to be efficient in clustering when the manifolds are of different dimensions, it is common in computer vision problems that this assumption is violated. In motion segmentation, for example, the manifolds for two translational motions are of the same dimension.

Chapter summary In this book chapter, we give a summary of the newly developed algorithms for performing simultaneous nonlinear dimensionality reduction and clustering of data sampled from multiple submanifolds of a Riemannian manifold and demonstrate their applications to vision-based analysis. More specifically, this chapter first reviews how to perform locally linear manifold clustering and dimensionality reduction for data sampled from nonlinear manifolds using the Euclidean metric as the distance between feature vectors and its limitations [18]. Unfortunately, such manifold clustering algorithms assume that the feature vectors are embedded in a Euclidean space and use (at least locally) the Euclidean metric or a variation of it to perform clustering. While this may be appropriate in some cases, there are several computer vision problems where it is more natural to consider features that live in a non-Euclidean space. For example, Grassmann manifolds and Lie groups are used for motion segmentation and multibody factorization; symmetric positive semi-definite matrices are common in diffusion tensor imaging and structure tensor analysis; and the statistical manifold, that is, the space of probability density functions, is found in texture analysis. The second part of this chapter shows a novel algorithm for clustering data sampled from multiple submanifolds of a Riemannian manifold [19–21]. First, a representation of the data using generalizations of local nonlinear dimensionality reduction algorithms from Euclidean to Riemannian spaces is learnt. Such generalizations exploit geometric properties of the Riemannian space, particularly its Riemannian metric. Then, assuming that the data points from different groups are separated, it is shown that the null space of a matrix built from the local representation gives the segmentation of the data. Finally, the applications to various vision problems are shown.

2 Review of Local Nonlinear Dimensionality Reduction Methods in Euclidean Spaces

In this section, we review three local nonlinear dimensionality reduction algorithms for data lying in a single manifold. Section 2.1 reviews the classical LLE, Laplacian Eigenmaps, and Hessian LLE algorithms for a nonlinear manifold. Section 2.2 shows that it is possible that such NLDR algorithms become degenerate when the data lies in a linear subspace.

2.1 NLDR for a Nonlinear Manifold

In this section, we review three local NLDR algorithms. Let $X = \{\mathbf{x}_i \in \mathcal{M}\}_{i=1}^n$ be a set of n data points sampled from a d -dimensional manifold \mathcal{M} embedded in \mathbb{R}^D , $d \ll D$. We assume that the n points are k -connected, that is, for any two points $\mathbf{x}_i, \mathbf{x}_j \in X$ there is an ordered sequence of points in X having \mathbf{x}_i and \mathbf{x}_j as endpoints, such that any two consecutive points in the sequence have at least one k -nearest neighbor in common. The goal of dimensionality reduction is to find a set of vectors $\{\mathbf{y}_i \in \mathbb{R}^d\}_{i=1}^n$, such that nearby points remain close and distant points remain far.

Locally Linear Embedding (LLE) [36] assumes that the local neighborhood of a point in the manifold can be well approximated by the affine subspace spanned by the k -nearest neighbors of the point, and finds a low-dimensional embedding of the data based on these affine approximations. *Laplacian Eigenmaps (LE)* [6] are based on computing the low dimensional representation that best preserves *locality* instead of *local linearity* in LLE. *Hessian LLE (HLL)* [14] bears substantial resemblance to LLE and LE, with the main difference being that the local neighborhood is represented by the tangent space at each point and the Laplacian matrix is replaced by the Hessian matrix. The main steps of these local NLDR algorithms are as follows:

1. *Nearest neighbor search*: for each data point $\mathbf{x}_i \in X$, find its k -nearest neighbors (k NN) $\{\mathbf{x}_{i_j}\}_{j=1}^k$ according to the Euclidean distance.
2. *Construction of similarity matrix*: construct a weighted graph whose elements encode the local geometry of the data. Define a similarity matrix M based on these weights. M is symmetric and positive semidefinite.
3. *Sparse eigenvalue problem*: obtain the embedding coordinates, that is, the columns of $Y = [\mathbf{y}_1, \dots, \mathbf{y}_n]^\top \in \mathbb{R}^{n \times d}$, from the d (generalized) eigenvectors of the matrix M associated with its second to $(d + 1)$ th smallest (generalized) eigenvalues. The vector of all ones, $\mathbf{1} \in \mathbb{R}^n$, is a eigenvector of M associated with eigenvalue 0.

We now describe the construction of M for each NLDR algorithm in greater detail.

2.1.1 Calculation of M in LLE

1. *Weight matrix*: find a matrix of weights $W \in \mathbb{R}^{n \times n}$ whose entries W_{ij} minimize the reconstruction error

$$\varepsilon(W) = \sum_{i=1}^n \left\| \sum_{j=1}^n W_{ij} \mathbf{x}_j - \mathbf{x}_i \right\|^2 = \sum_{i=1}^n \text{dist}^2(\hat{\mathbf{x}}_i, \mathbf{x}_i), \quad (1)$$

subject to the constraints (i) $W_{ij} = 0$ if \mathbf{x}_j is not a k -nearest neighbor of \mathbf{x}_i and (ii) $\sum_{j=1}^n W_{ij} = 1$. In (1), $\hat{\mathbf{x}}_i = \mathbf{x}_i + \sum_{j=1}^n W_{ij} \overrightarrow{\mathbf{x}_i \mathbf{x}_j}$ is the linear interpolation of \mathbf{x}_i and its k NN. The solution to this problem can be computed as

$$\begin{bmatrix} W_{i i_1} & W_{i i_2} & \dots & W_{i i_k} \end{bmatrix} = \frac{\mathbf{1}^\top C_i^{-1}}{\mathbf{1}^\top C_i^{-1} \mathbf{1}} \in \mathbb{R}^{1 \times k}, \quad (2)$$

where $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones, and $C_i \in \mathbb{R}^{k \times k}$ is the local Gram matrix at \mathbf{x}_i , that is, $C_i(j, l) = (\mathbf{x}_j - \mathbf{x}_i) \cdot (\mathbf{x}_l - \mathbf{x}_i)$.

2. *Objective function*: find vectors $\{\mathbf{y}_i \in \mathbb{R}^d\}_{i=1}^n$ that minimize the error

$$\phi(Y) = \sum_{i=1}^n \left\| \mathbf{y}_i - \sum_{j=1}^n W_{ij} \mathbf{y}_j \right\|^2 = \text{trace}(Y^\top M Y), \quad (3)$$

subject to the constraints (i) $\sum_{i=1}^n \mathbf{y}_i = \mathbf{0}$ and (ii) $\frac{1}{n} \sum_{i=1}^n \mathbf{y}_i \mathbf{y}_i^\top = I$. The solution to this optimization problem is given by the d eigenvectors of $M = (I - W)^\top \times (I - W)$ associated with its second to $(d + 1)$ th smallest eigenvalues.

2.1.2 Calculation of M in LE

1. *Weight matrix*: construct a matrix of weights $W \in \mathbb{R}^{n \times n}$ where the entries of W , W_{ij} , measure the proximity between two points \mathbf{x}_i and \mathbf{x}_j subject to the constraint $W_{ij} = 0$ if \mathbf{x}_j is not a k -nearest neighbor of \mathbf{x}_i . A possible weight of generating W is to use the heat kernel

$$W_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / \sigma^2). \quad (4)$$

2. *Objective function*: find vectors $\{\mathbf{y}_i \in \mathbb{R}^d\}_{i=1}^n$ that minimize the error

$$\phi(Y) = \sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij} = \text{trace}(Y^\top M Y), \quad (5)$$

subject to the constraints (i) $Y^\top D \mathbf{1} = \sum_{i=1}^n D_{ii} \mathbf{y}_i = \mathbf{0}$ (weighted low-dimensional coordinates centered at the origin) and (ii) $Y^\top D Y = I$ (weighted low-dimensional coordinates having unit covariance). In (5), $M = D - W$ is the

graph Laplacian matrix and D is a diagonal matrix whose entries are given by $D_{ii} = \sum_j W_{ij}$. The solution to this optimization problem is given by the d generalized eigenvectors of (M, D) associated with its second to $(d + 1)$ th smallest generalized eigenvalues.

2.1.3 Calculation of M in HLLE

1. *Tangent coordinates*: for each data point \mathbf{x}_i , let $\{\mathbf{x}_{i_j}\}_{j=1}^k$ be its k NN. Form the D by D covariance matrix $\text{cov}(\mathbf{x}_i) = \frac{1}{k} \sum_{j=1}^k (\mathbf{x}_{i_j} - \bar{\mathbf{x}}_i)(\mathbf{x}_{i_j} - \bar{\mathbf{x}}_i)^\top$, where $\bar{\mathbf{x}}_i$ is the mean of the k NN. Perform an eigenanalysis of the matrix $\text{cov}(\mathbf{x}_i)$ to obtain the d eigenvectors $\{\mathbf{u}_q \in \mathbb{R}^D\}_{q=1}^d$. The tangent coordinates of the k NN are given by the d columns of the $k \times d$ matrix V given below, where $p = 1, \dots, k$ and $q = 1, \dots, d$

$$V_{pq} = (\mathbf{x}_{i_p} - \bar{\mathbf{x}}_i)^\top \mathbf{u}_q = \langle \mathbf{x}_{i_p} - \bar{\mathbf{x}}_i, \mathbf{u}_q \rangle. \quad (6)$$

2. *Objective function*: the embedding vectors are obtained based on the null vectors of a matrix M that indicates the Hessian quadratic cost. While we refer the reader to [14] for details on the estimation of M , the basic principle is as follows. We first locally estimate a Hessian operator h^i at each point \mathbf{x}_i in the manifold in a least squares sense. In particular, consider a smooth function $f : \mathcal{M} \rightarrow \mathbb{R}$. We evaluate the function at all k NN of a point \mathbf{x}_i in the manifold and stack these entries into a vector \mathbf{f}_i . It can be shown that $h^i \mathbf{f}_i$ approximates the entries of the Hessian, whose (p, q) th entry is given by $\frac{\partial^2 f}{\partial V_p \partial V_q}$. These local estimates are then used to obtain an empirical estimate of the (i, j) th entry of M as

$$M_{i,j} = \sum_l \sum_r ((h^l)_{r,i} (h^l)_{r,j}). \quad (7)$$

The embedding coordinates are then found by selecting a basis for the space spanned by d eigenvectors of M associated with its second to $(d + 1)$ th smallest eigenvalues with the restriction that it provides an orthonormal basis to a specific fixed neighborhood \mathbf{N} . Let U denote the $n \times d$ matrix associated with the second to $(d + 1)$ th smallest eigenvectors where $U_{l,r}$ is the l th entry in the r th eigenvector of M . The embedding coordinates is obtained as $UR^{-\frac{1}{2}}$, where $R_{r,s} = \sum_{j \in \mathbf{N}} U_{j,r} U_{j,s}$.

2.2 NLDR for a Single Subspace

Consider now the application of NLDR to a single k -connected linear manifold. As the goal of NLDR is to unfold a low-dimensional manifold of a high-dimensional space into a low-dimensional linear subspace, intuitively one would expect that if

NLDR is applied to a dataset that is already a subspace of dimension d , the output representation should again be a subspace of the same dimension.

We will first illustrate that one of the NLDR algorithms, namely LLE, becomes degenerate when the data lies in a linear subspace. Proposition 1 [33] below shows that when d is known, the low-dimensional representation is indeed a subspace of dimension d , which is contained in the null space of the matrix M representing the local geometry of the manifold. However, since the vector $\mathbf{1}$ is also in $\ker(M)$, some degeneracies can show up when applying LLE to data lying in a linear subspace, as shown by the following proposition in [33].

Proposition 1 *Assume that the data points $\mathbf{x}_i \in \mathbb{R}^D$ lie in a subspace of \mathbb{R}^D of dimension $d < k - 1$. Then the dimension of the null space of M is at least $d + 1$.*

Proof Since the data lie in a subspace of \mathbb{R}^D of dimension $d < k - 1$, each point $\{\mathbf{x}_i\}$ can be reconstructed with zero error in (1), that is, for all $i = 1, \dots, n$, there are W_{ij} such that $\mathbf{x}_i = \sum_{j=1}^n W_{ij} \mathbf{x}_j$. If we let $X \in \mathbb{R}^{n \times D}$ be the matrix whose rows are the data points, then we have that $WX = X$, hence $MX = 0$. In other words, the D n -dimensional vectors formed by taking each one of the D coordinates of the n given data points are in the null space of M . Therefore, the null space of M is at least d -dimensional, because $\text{rank}(X) = d$. On the other hand, since the data points live in a subspace of dimension d , there exist a matrix $T \in \mathbb{R}^{D \times d}$, $T^\top T = I$ and vectors $\{\mathbf{y}_i\}$ such that $\mathbf{x}_i = T\mathbf{y}_i + \mathbf{m}$ and $\sum_{i=1}^n \mathbf{y}_i = 0$, where $\mathbf{m} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i \in \mathbb{R}^D$ is the mean of the data. Now, by construction, the vector of all ones $\mathbf{1}$ is also in $\ker(M)$, because $\sum_{j=1}^n W_{ij} = 1$. This implies that $X = YT^\top + \mathbf{1}\mathbf{m}^\top \Rightarrow MX = MYT^\top + M\mathbf{1}\mathbf{m}^\top = 0 \Rightarrow MYT^\top = 0 \Rightarrow MYT^\top T = 0$; hence $MY = 0$, where $Y \in \mathbb{R}^{n \times d}$ is a matrix whose rows are the $\{\mathbf{y}_i\}$ vectors. Since in addition $\sum_{i=1}^n \mathbf{y}_i = 0$, we have that $\mathbf{1}^\top Y = \mathbf{0}^\top$, hence the vector $\mathbf{1}$ is linearly independent from the columns of Y . Therefore, the null space of M is at least $(d + 1)$ -dimensional. \square

From Proposition 1, we see that if we apply LLE to data lying in a subspace of dimension d and choose the second to $(d + 1)$ th smallest eigenvectors of M for dimensionality reduction, we might not get the correct subspace reconstruction. This is because the embedding eigenvectors (the columns of Y) may be mixed with the vector $\mathbf{1}$, which is also a null vector of M and therefore, we cannot guarantee that $\mathbf{1}$ is the first smallest eigenvector given.

We will now illustrate how Proposition 1 is applicable for Laplacian eigenmaps and HLLE as well. For the Laplacian eigenmaps, it is shown in [6] that the solution that LLE finds is an approximation of the eigenfunctions of the iterated Laplace Beltrami operator \mathbf{L}^2 whereas LE attempts to find the eigenfunctions of the Laplace Beltrami operator \mathbf{L} . Note that eigenfunctions of \mathbf{L}^2 coincide with those of \mathbf{L} . Therefore, depending on the approximation that is used for the graph Laplacian, it is also possible that LE suffers from the same degeneracy. Finally, for Hessian LLE, as the entries of the Hessian approximates $\frac{\partial^2 f}{\partial V_p \partial V_q}$, it is easy to see that when the function f is linear, it becomes a null eigenfunction as well. Therefore, it is possible that the embedding vectors are mixed with the vector $\mathbf{1}$ when we have linear manifolds for LE and HLLE as well.

3 Manifold Clustering and Dimensionality Reduction Using the Euclidean Metric

This section presents an algorithm for simultaneous NLDR and manifold clustering. In Sect. 3.1, we review the algorithm for clustering nonlinear manifolds using NLDR algorithms. This algorithm is based on the fact that the null vectors of M are actually m membership vectors indicating the grouping of the data. In Sect. 3.2, we show that these clustering algorithms based on NLDR can become degenerate when applied to data lying in multiple linear subspaces. More specifically, we show that if the data live in a k -separated union of m connected manifolds, of which $m_1 \leq m$ are linear subspaces of dimensions $\{d_i\}_{i=1}^{m_1}$, then, depending on the NLDR algorithm used, the null space of M might contain m eigenvectors that give the segmentation of the data and $\sum_{i=1}^{m_1} d_i$ eigenvectors that give the embedding coordinates for the subspaces.

3.1 Manifold Clustering and Dimensionality Reduction for a k -Separated Union of k -Connected Nonlinear Manifolds

In this section, we review an extension of the NLDR algorithm for clustering a union of m k -connected manifolds under the assumption that the manifolds are k -separated, that is, *no k NN of a data point in a manifold lies in one of the other $(m - 1)$ manifolds*. The following proposition shows how NLDR (LLE, LE, and HLLE) can be extended for clustering a k -separated union of m k -connected nonlinear manifolds. The proposition follows from the block-diagonal properties of M in the presence of multiple k -separated nonlinear manifolds. Polito and Perona [33] illustrates this proposition for LLE only and make use of it to cluster different groups. Notice that, contrary to intuition, the case of nonlinear manifolds is simpler than the case of linear subspaces, as we will see in Sect. 3.2.

Proposition 2 *Let $\{\mathbf{x}_i\}_{i=1}^n$ be a set of points drawn from a k -separated union of m k -connected nonlinear manifolds of dimension $d < k - 1$. There exist m vectors $\{\mathbf{v}_j\}_{j=1}^m$ in the null space of M such that \mathbf{v}_j corresponds to the j th group of points, that is, $\mathbf{v}_{j,i} = 1$ if the i th data point is in the j th manifold ($\mathbf{x}_i \in \mathcal{M}_j$), and $\mathbf{v}_{j,i} = 0$ otherwise ($\mathbf{x}_i \notin \mathcal{M}_j$).*

Proof If the data can be partitioned into m k -connected groups, then the matrix M is block-diagonal with m blocks. This is because if points \mathbf{x}_i and \mathbf{x}_j belong to different groups, then they cannot be k NN of each other, hence $M_{ij} = 0$. Therefore, the matrix M is also block diagonal, and we can write it as $M = \text{diag}(M_j)$, where $M_j \in \mathbb{R}^{n_j \times n_j}$ is the matrix for the j th group. Now, from the properties of the local NLDR algorithms reviewed in Sect. 2, we know that each one of the m blocks of M , has the vector $\mathbf{1} \in \mathbb{R}^{n_j}$ in its null space. Therefore, there are m vectors $\{\mathbf{v}_j\}$ in

Algorithm 1: Unsupervised clustering and dimensionality reduction on nonlinear manifolds

Given data points $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{M}$

1. *nearest neighbors*: find the k NN of each data point \mathbf{x}_i
 2. *construction of M* : for each NLDR algorithm, construct the appropriate M on the entire data and compute a basis B for the null space of M
 3. *clustering*: compute the segmentation of the data by applying K -means to the rows of B
 4. *low-dimensional embedding*: apply NLDR as described in Sect. 2.1 to each group to obtain a low-dimensional embedding for each submanifold
-

$\ker(M)$, with each \mathbf{v}_j taking the values 1 and 0, indicating the group membership, as claimed. \square

Notice that when computing a basis $B \in \mathbb{R}^{n \times m}$ for $\ker(M)$, we do not necessarily obtain the set $\{\mathbf{v}_j\}_{j=1}^m$, but rather linear combinations of them, including the constant vector. Nevertheless, a generic linear combination of these membership vectors will still contain the segmentation of the data. Hence, we can cluster the data into m groups by applying a central clustering algorithm to the rows of B , for example, K -means. Therefore, this algorithm can be seen as a spectral clustering algorithm where the similarity matrix is obtained from the M matrix of NLDR. Algorithm 1 summarizes the dimensionality reduction and clustering algorithm for a union of k -connected nonlinear manifolds.

3.2 Degeneracies for a k -Separated Union of k -Connected Linear Manifolds

In this section, we illustrate the limitations of Algorithm 1 [18]. More precisely, we will show that NLDR becomes degenerate when the data points $\{\mathbf{x}_i\}_{i=1}^n$ are drawn from a union of m k -connected subspaces $\{\mathcal{M}_j\}_{j=1}^m$ of \mathbb{R}^D with dimensions $\{d_j\}_{j=1}^m$ [18]. We first define two different types of vectors, given as follows:

Definition 1 Membership vectors \mathbf{v}_j indicate the membership of each point for manifold \mathcal{M}_j . That is, $\mathbf{v}_{j,i} = 1$ if $\mathbf{x}_i \in \mathcal{M}_j$, and $\mathbf{v}_{j,i} = 0$ otherwise. Note that \mathbf{v}_j is an $n \times 1$ vector.

Definition 2 Embedding vectors \mathbf{e}_j give the embedding coordinates for each manifold \mathcal{M}_j . We define \mathbf{e}_j as the $n \times d_j$ matrix that contains the low-dimensional coordinates of each manifold. That is,

$$\mathbf{e}_j = [\mathbf{0}_{(d_j \times \sum_{i=1}^{j-1} n_i)}, Y_j^\top, \mathbf{0}_{(d_j \times \sum_{i=j+1}^m n_i)}]^\top. \quad (8)$$

From Propositions 1–2, we know that there are two types of vectors in the null space of M : the embedding vectors coming from the coordinates and the membership vectors coming from each one of the m connected components. However, it is unclear if these vectors are linearly independent, and if one can recover the segmentation of the data and a nonlinear embedding for each group from $\ker(M)$. That is because an arbitrary vector in $\ker(M)$ is a linear combination of the embedding and membership vectors. The following proposition addresses these issues in detail.

Proposition 3 *Let $\{\mathbf{x}_i\}_{i=1}^n$ be a set of points drawn from a k -separated union of m k -connected subspaces of dimensions $d_j < k - 1$, $j = 1, \dots, m$. The null space of M is of dimension at least $m + \sum_{j=1}^m d_j$ and contains orthonormal zero-padded vectors formed from the individual embedding $\{\mathbf{e}_j\}$ and membership $\{\mathbf{v}_j\}$ vectors.*

Proof From Proposition 2, we know that M is block diagonal and can be written as $M = \text{diag}(M_j)$, where $M_j \in \mathbb{R}^{n_j \times n_j}$ is the matrix for the j th group and n_j is the number of points in the j th group. From Proposition 1, we also know that the matrix M_j has $d_j + 1$ vectors in the null space: the vector of all ones and the d_j linearly independent columns of the matrix of coordinates $Y_j \in \mathbb{R}^{n_j \times d_j}$. That is $M_j [Y_j \mathbf{1}] = 0$. Therefore, the matrix $Y = \text{diag}([Y_j \mathbf{1}_{(n_j \times 1)}]) \in \mathbb{R}^{n \times (\sum_{j=1}^m d_j + m)}$ is such that $MY = 0$. Furthermore, as Y is block diagonal and $\text{rank}([Y_j \mathbf{1}]) = d_j + 1$, we have that Y is of rank $\sum_{j=1}^m d_j + m$, and so the dimension of $\ker(M)$ is at least $\sum_{j=1}^m d_j + m$. Also, the matrix of embedding vectors of the j th group $\mathbf{e}_j = [\mathbf{0}, Y_j^\top, \mathbf{0}]^\top \in \mathbb{R}^{n \times d_j}$ is orthogonal to its membership vector $\mathbf{v}_j = [\mathbf{0}, \mathbf{1}_{(1 \times n_j)}, \mathbf{0}]^\top \in \mathbb{R}^{n \times 1}$. Since \mathbf{e}_j and \mathbf{v}_j are zero-padded, they are always orthogonal to \mathbf{e}_i and \mathbf{v}_i for $i \neq j$. In addition, one can choose the embedding vectors \mathbf{e}_j to be orthogonal to each other, because the matrix M is symmetric. Therefore, we can assume that the vectors $\{\mathbf{v}_1, \mathbf{e}_1, \dots, \mathbf{v}_m, \mathbf{e}_m\}$ are orthonormal. \square

Given a set of points $\{\mathbf{x}_i \in \mathbb{R}^D\}_{i=1}^n$, it follows from the proof of Proposition 3, $\ker(M)$ contains the orthonormal set of embedding vectors \mathbf{e}_j and membership vectors \mathbf{v}_j . More precisely, when the points $\{\mathbf{x}_i\}_{i=1}^n$ are drawn from a k -separated union of m k -connected manifolds, we have,

1. for m nonlinear manifolds:

$$\{\mathbf{v}_j\}_{j=1}^m \in \ker(M) \quad \text{and} \quad \dim(\ker(M)) = m$$

2. for m linear manifolds of dimensions $\{d_j\}_{j=1}^m$:

$$\{\mathbf{v}_j\}_{j=1}^m, \{\mathbf{e}_j\}_{j=1}^m \in \ker(M) \quad \text{and} \quad \dim(\ker(M)) = m + \sum_{j=1}^m d_j$$

3. for $m - m_1$ nonlinear manifolds, and m_1 linear manifolds of dimensions $\{d_j\}_{j=1}^{m_1}$:

$$\{\mathbf{v}_j\}_{j=1}^m, \{\mathbf{e}_j\}_{j=1}^{m_1} \in \ker(M) \quad \text{and} \quad \dim(\ker(M)) = m + \sum_{j=1}^{m_1} d_j$$

Therefore, in the presence of linear manifolds, we cannot directly obtain the segmentation of the data or an embedding for each one of the subspaces from $\ker(M)$, since an arbitrary vector in $\ker(M)$ is a linear combination of both membership and embedding vectors. This is a limitation of Algorithm 1; for the rest of this chapter, we assume that the data does not lie on linear manifolds.

4 Manifold Clustering and Dimensionality Reduction Using the Riemannian Metric

The NLDR techniques presented in Sects. 2 and 3.1 are applicable only in the presence of manifolds with *unknown structure*. Every operation is approximated by the corresponding Euclidean operation as the metric is unknown. However, for Riemannian manifolds with well-studied geometries, closed-form formulae for Riemannian operations are often available. The question now is to extend NLDR techniques for Riemannian manifolds in a way that takes into consideration the appropriate Riemannian structure. In this section, we present an algorithm for clustering and dimensionality reduction on Riemannian manifolds. We first present a brief summary of the theory of Riemannian manifolds in Sect. 4.1. For a more complete description, we refer the reader to [13]. Next, in Sect. 4.2, we illustrate how to extend manifold clustering and dimensionality reduction algorithms to Riemannian manifolds by adopting the framework in [21]. This framework has been applied to motion segmentation in [21], diffusion tensor images in [19] and probability density functions in [20].

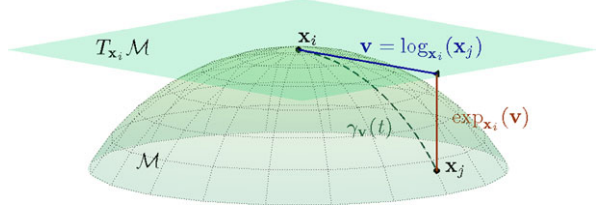
4.1 Review of Riemannian Manifolds

In this section, we will give an overview of Riemannian theory and show how the various operations such as interpolation on the manifold and computation of the mean and principal components are carried out.

A smooth manifold is a topological space that is locally diffeomorphic to a Euclidean space smooth function $\gamma(t) : \mathbb{R} \rightarrow \mathcal{M}$. Figure 1 shows an example of a two-dimensional manifold. The *tangent space* of \mathcal{M} at a point $\mathbf{x} \in \mathcal{M}$, denoted as $T_{\mathbf{x}}\mathcal{M}$, is then defined as the span of the tangent vectors for all the possible curves γ passing through \mathbf{x} . A *Riemannian metric* is a continuous collection of dot products $\langle \cdot, \cdot \rangle_{\mathbf{x}}$. Using this metric, we define the length of a curve between two points $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M}$ as

$$\mathcal{L}_a^b(\gamma) = \int_a^b \langle \dot{\gamma}(t) | \dot{\gamma}(t) \rangle_{\gamma(t)}^{\frac{1}{2}} dt, \quad (9)$$

Fig. 1 An example of a two-dimensional manifold. The tangent plane at \mathbf{x}_i , together with the exponential and logarithm maps relating \mathbf{x}_i and \mathbf{x}_j , are also shown



where $\gamma(a) = \mathbf{x}_i$ and $\gamma(b) = \mathbf{x}_j$. A curve between \mathbf{x}_i and \mathbf{x}_j with minimum length is called a geodesic. The distance between two points in the manifold is subsequently defined as the length of the geodesic curve between them,

$$\text{dist}(\mathbf{x}_i, \mathbf{x}_j) = \mathcal{L}_0^1(\gamma), \quad \gamma(0) = \mathbf{x}_i, \quad \gamma(1) = \mathbf{x}_j. \quad (10)$$

Let \mathbf{v} be a unit-length tangent vector in $T_{\mathbf{x}}\mathcal{M}$, that is, $\|\mathbf{v}\|_{\mathbf{x}} = \langle \mathbf{v}, \mathbf{v} \rangle_{\mathbf{x}}^{\frac{1}{2}} = 1$. We can then define the *exponential map* $\exp_{\mathbf{x}} : T_{\mathbf{x}}\mathcal{M} \rightarrow \mathcal{M}$, which maps each tangent vector $t\mathbf{v} \in T_{\mathbf{x}}\mathcal{M}$ to the point $\gamma(t) \in \mathcal{M}$ obtained by following the geodesic $\gamma(t)$ (parametrized with arc-length) passing through \mathbf{x} with direction \mathbf{v} for a distance t . Define the set $C_{\mathbf{x}} \subset \mathcal{M}$ as the set of all the points $\mathbf{x}_i = \exp(t_0\mathbf{v})$ such that $\gamma = \exp(t\mathbf{v})$ is a length-minimizing geodesic for $t \in [0, t_0]$. The boundary of $C_{\mathbf{x}}$ is called the *cut locus*, and, intuitively, it is the set of points for which the distance from \mathbf{x} stops to be a differentiable function of t . The exponential map is therefore invertible for all the points in the interior of $C_{\mathbf{x}}$ and we can define the *logarithm map* $\log_{\mathbf{x}} : C_{\mathbf{x}} \rightarrow T_{\mathbf{x}}\mathcal{M}$ as $\log_{\mathbf{x}} = \exp_{\mathbf{x}}^{-1}$. Note that for any $\mathbf{x}_i \in C_{\mathbf{x}}$ and $\mathbf{x}_i = \exp_{\mathbf{x}}(t\mathbf{v})$ we have that (by definition),

$$\|\log_{\mathbf{x}}(\mathbf{x}_i)\|_{\mathbf{x}} = \|t\mathbf{v}\|_{\mathbf{x}} = \text{dist}(\mathbf{x}, \mathbf{x}_i) = t. \quad (11)$$

Linear geodesic interpolation makes use of the exponential and logarithm maps and is defined as $\hat{\mathbf{x}} = \exp_{\mathbf{x}_i}(w \log_{\mathbf{x}_i}(\mathbf{x}_j))$, $w \in [0, 1]$. $\hat{\mathbf{x}}$ is the linear interpolation at $t = w$ of \mathbf{x}_i and \mathbf{x}_j . Finally, we recall that the Riemannian metric, exponential and logarithm maps depend on the point \mathbf{x} under consideration, hence the subscripts reflecting this dependency.

We will now briefly summarize how to calculate the mean and principal components of data points lying in a manifold. As defined by Fréchet in [16] and used in several recent works [15, 32], the intrinsic mean $\bar{\mathbf{x}}$ is defined as the solution to the following minimization problem

$$\bar{\mathbf{x}} = \arg \min_{\mathbf{x} \in \mathcal{M}} \sum_{i=1}^n \text{dist}(\mathbf{x}, \mathbf{x}_i)^2 = \arg \min_{\mathbf{x} \in \mathcal{M}} \sum_{i=1}^n \|\log_{\mathbf{x}}(\mathbf{x}_i)\|_{\mathbf{x}}^2. \quad (12)$$

Note that, unlike in the Euclidean case, in general there is no closed form for $\bar{\mathbf{x}}$. Moreover, there is no guarantee that $\bar{\mathbf{x}}$ exists or is unique. However, one can show the existence and uniqueness of $\bar{\mathbf{x}}$ [27] by assuming that the data lie in a small enough neighborhood, that is, the maximum distance between any \mathbf{x}_i and \mathbf{x}_j is small enough. Furthermore, $\bar{\mathbf{x}}$ can be computed as shown in Algorithm 2.

Algorithm 2: Intrinsic mean

Given data points $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{M}$, a predefined threshold ϵ , maximum number of iterations T

1. initialize $t = 1$, $\bar{\mathbf{x}}_1 = \mathbf{x}_i$ for a random i , $\mathbf{v} \neq 0 \in T_{\bar{\mathbf{x}}_1} \mathcal{M}$
2. while $t \leq T$ or $\|\mathbf{v}\|_{\bar{\mathbf{x}}} \geq \epsilon$
 - (a) compute tangent vector $\mathbf{v} = \frac{1}{n} \sum_{i=1}^n \log_{\bar{\mathbf{x}}_t}(\mathbf{x}_i)$
 - (b) set $\bar{\mathbf{x}}_{t+1} = \exp_{\bar{\mathbf{x}}_t}(\mathbf{v})$

Algorithm 3: Principal geodesic analysis

Given data points $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{M}$

1. compute intrinsic mean $\bar{\mathbf{x}}$ as in Algorithm 2
2. calculate the tangent vectors $\mathbf{v}_i = \log_{\bar{\mathbf{x}}}(\mathbf{x}_i)$ about $\bar{\mathbf{x}}$
3. construct the sample covariance matrix $\text{cov}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \mathbf{v}_i \mathbf{v}_i^\top$
4. perform eigenanalysis of the matrix $\text{cov}(\mathbf{x})$, with the eigenvectors $\{\mathbf{u}_i\}_{i=1}^d$ giving the principal directions. $\{\mathbf{u}_i\}_{i=1}^d$ forms an orthonormal basis for $T_{\bar{\mathbf{x}}} \mathcal{M}$

Table 1 Comparison of Euclidean and Riemannian operations, where $\{\mathbf{x}_i\}_{i=1}^n$ are data points

| Operation | Euclidean | Riemannian |
|--|--|---|
| Subtraction $\overrightarrow{\mathbf{x}_i \mathbf{x}_j}$ | $\mathbf{x}_j - \mathbf{x}_i$ | $\log_{\mathbf{x}_i}(\mathbf{x}_j)$ |
| Addition \mathbf{x}_j | $\mathbf{x}_i + \overrightarrow{\mathbf{x}_i \mathbf{x}_j}$ | $\exp_{\mathbf{x}_i}(\overrightarrow{\mathbf{x}_i \mathbf{x}_j})$ |
| Distance $\text{dist}(\mathbf{x}_i, \mathbf{x}_j)$ | $\ \overrightarrow{\mathbf{x}_i \mathbf{x}_j}\ = \ \mathbf{x}_j - \mathbf{x}_i\ $ | $\ \log_{\mathbf{x}_i}(\mathbf{x}_j)\ _{\mathbf{x}_i} = \sqrt{\langle \log_{\mathbf{x}_i}(\mathbf{x}_j), \log_{\mathbf{x}_i}(\mathbf{x}_j) \rangle_{\mathbf{x}_i}}$ |
| Linear interpolation $\hat{\mathbf{x}}$ | $\mathbf{x}_i + w \overrightarrow{\mathbf{x}_i \mathbf{x}_j}$ | $\exp_{\mathbf{x}_i}(w \overrightarrow{\mathbf{x}_i \mathbf{x}_j})$ |
| Mean $\bar{\mathbf{x}}$ | $\sum_{i=1}^n \overrightarrow{\mathbf{x} \mathbf{x}_i} = 0$ | $\sum_{i=1}^n \log_{\bar{\mathbf{x}}}(\mathbf{x}_i) = 0$ |
| Covariance $\text{cov}(\mathbf{x})$ | $\frac{1}{n} \sum_{i=1}^n (\overrightarrow{\mathbf{x} \mathbf{x}_i})(\overrightarrow{\mathbf{x} \mathbf{x}_i})^\top$ | $\frac{1}{n} \sum_{i=1}^n (\log_{\bar{\mathbf{x}}}(\mathbf{x}_i))(\log_{\bar{\mathbf{x}}}(\mathbf{x}_i))^\top$ |

Given $\bar{\mathbf{x}}$, the calculation of principal components on a Riemannian manifold is not as straightforward as in the Euclidean case. It involves projecting a point onto a geodesic curve, which is also defined as a minimization problem for which existence and uniqueness are not ensured [15]. Again, by making the assumptions that the data lie in a small neighborhood, the projection can be shown to be unique. In [15], it is shown that finding principal components boils down to doing PCA in the tangent vectors $\log_{\bar{\mathbf{x}}}(\mathbf{x}_i) \in T_{\bar{\mathbf{x}}} \mathcal{M}$ about the mean $\bar{\mathbf{x}}$. This algorithm, known as Principal Geodesic Analysis (PGA), is summarized in Algorithm 3. Table 1 compares the standard operations in Euclidean and Riemannian spaces.

4.2 Extending Manifold Clustering and Dimensionality Reduction to Riemannian Manifolds

In this section, we will show how to extend manifold clustering and dimensionality reduction to Riemannian manifolds. We assume here that the various Riemannian operations are known and closed-form. First of all, notice that the information about the local geometry of the manifold is essential only in the first two steps of each algorithm and therefore, modifications are made only to these two stages. The key issues are how to select the k NN and how to compute the matrix M representing the local geometry. As shown in [21], the former is straightforward, while the latter requires some thought. Given M , calculating the low-dimensional representation remains the same as in the Euclidean case. We let $X = \{\mathbf{x}_i \in \mathbb{R}^D\}_{i=1}^n$ be a set of n data points sampled from a known Riemannian manifold.

4.2.1 Selection of the Riemannian k NN

The first step of any NLDR algorithm is the computation of the k NN associated with each data point. We define the k NN of \mathbf{x}_i by incorporating the Riemannian distance, that is, *the k NN of \mathbf{x}_i are the k data points \mathbf{x}_j that minimize $\|\log_{\mathbf{x}_i}(\mathbf{x}_j)\|_{\mathbf{x}_i}$.*

4.2.2 Riemannian Calculation of M for LLE

The second step of LLE is to compute the matrix of weights $W \in \mathbb{R}^{n \times n}$. In order to do so, we will answer two main questions: (1) how does one express a point as a linear combination of its neighbors? and (2) what is the reconstruction cost? First of all, we know that from Sect. 4.1 that

$$\hat{\mathbf{x}}_{\text{Riem},i} = \exp_{\mathbf{x}_i} \left(\sum_{j=1}^n W_{ij} \log_{\mathbf{x}_i}(\mathbf{x}_j) \right), \quad (13)$$

is the geodesic linear interpolation of \mathbf{x}_i by $\{\mathbf{x}_j\}_{j=1}^n$. Now, instead of minimizing the Euclidean error, we rewrite (1) to minimize the Riemannian reconstruction error and make use of the fact that \exp and \log are inverse mappings. Therefore, we have

$$\varepsilon_{\text{Riem}}(W) = \sum_{i=1}^n \left\| \log_{\mathbf{x}_i}(\hat{\mathbf{x}}_{\text{Riem},i}) \right\|_{\mathbf{x}_i}^2 = \sum_{i=1}^n \left\| \sum_{j=1}^n W_{ij} \log_{\mathbf{x}_i}(\mathbf{x}_j) \right\|_{\mathbf{x}_i}^2, \quad (14)$$

subject to $W_{ij} = 0$ if \mathbf{x}_j is not a k NN of \mathbf{x}_i and $\sum_j W_{ij} = 1$. Using similar manipulations as in the Euclidean case, the optimal weights are obtained as in (2), with the local Gram matrix $C_i \in \mathbb{R}^{k \times k}$ defined as

$$C_i(j, l) = \langle \log_{\mathbf{x}_i}(\mathbf{x}_j), \log_{\mathbf{x}_i}(\mathbf{x}_l) \rangle_{\mathbf{x}_i}. \quad (15)$$

M is then $(I - W)^\top (I - W)$.

4.2.3 Riemannian Calculation of M for LE

Here, instead of attempting to write each data point as a linear combination of its k NN, we find a matrix of weights $W \in \mathbb{R}^{n \times n}$ whose entries W_{ij} measure the proximity between two points \mathbf{x}_i and \mathbf{x}_j as in (4). Therefore, modifying LE for Riemannian manifolds is less involved than in the case of LLE. Instead of using $\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2)$ as in (4), we construct the weight matrix W using the Riemannian distance as

$$W_{ij} = \exp\left(-\frac{\text{dist}_{\text{Riem}}(\mathbf{x}_i, \mathbf{x}_j)^2}{\sigma^2}\right) = \exp\left(-\frac{\|\log_{\mathbf{x}_i}(\mathbf{x}_j)\|_{\mathbf{x}_i}^2}{\sigma^2}\right), \quad (16)$$

subject to the constraint $W_{ij} = 0$ if \mathbf{x}_j is not a k -nearest neighbor of \mathbf{x}_i . As before, $M = D - W$ and D is a diagonal matrix, where $D_{ii} = \sum_j W_{ij}$.

4.2.4 Riemannian Calculation of M for HLLE

The second step of HLLE involves computing the tangent coordinates for each \mathbf{x}_i by applying Euclidean PCA to its neighbors. This implicitly assumes that these local points lie in a subspace. This assumption is no longer valid if \mathbf{x}_i and its k NN lie in a Riemannian manifold. From Sect. 4.1, we know that in this case, calculating the principal geodesic components and the projection coordinates is not as simple as doing Euclidean PCA. There is a need to incorporate the correct Riemannian metric, mean and covariance matrix.

Again, let $\{\mathbf{x}_{i,j}\}_{j=1}^k$ denote the set of k -nearest neighbors of \mathbf{x}_i . First, we calculate the intrinsic mean $\bar{\mathbf{x}}_i$ of the k NN (Algorithm 2). Next, we find the tangent vectors $\mathbf{v}_j = \log_{\bar{\mathbf{x}}_i}(\mathbf{x}_{i,j})$ about $\bar{\mathbf{x}}_i$ and the geodesic principal directions $\{\mathbf{u}_q\}_{q=1}^d$ using PGA (Algorithm 3). Since $\{\mathbf{u}_q \in \mathbb{R}^D\}_{q=1}^d$ is an orthonormal basis for $T_{\bar{\mathbf{x}}_i}\mathcal{M}$, we will rewrite the projection operator in (6) using the Riemannian metric. Thus, the tangent coordinates of the k NN are given by the $k \times d$ matrix V , where

$$V_{pq} = \langle \log_{\bar{\mathbf{x}}_i}(\mathbf{x}_{i,p}), \mathbf{u}_q \rangle_{\bar{\mathbf{x}}_i}, \quad p = 1, \dots, k, \quad q = 1, \dots, d. \quad (17)$$

Once the tangent coordinates are found, the estimation of the Hessian matrix M is the same as in the Euclidean case (7).

4.2.5 Calculation of the Embedding Coordinates

The last step of NLDR is to find a Euclidean low-dimensional representation of the data points. As this step is independent of the Riemannian structure, one can find the embedding coordinates as described in Sect. 2. That is, the embedding coordinates are obtained based on the d (generalized) eigenvectors of the matrix M associated with its second to $(d+1)$ th smallest (generalized) eigenvalues. Finally, notice that if the Riemannian operations are available in closed-form, then extending NLDR to Riemannian manifolds do not require significant additional computational complexity.

Algorithm 4: Unsupervised clustering and dimensionality reduction on Riemannian manifolds

Given data points $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathcal{M}$

1. *nearest neighbors*: find the k NN of each data point \mathbf{x}_i according to the Riemannian distance
 2. *construction of M* : for each NLDR algorithm, construct the appropriate M described in Sect. 4.2 on the entire data and compute a basis B for the null space of M
 3. *clustering*: compute the segmentation of the data by applying K-means to the rows of B
 4. *low-dimensional embedding*: apply NLDR as described in Sect. 4.2 to each group to obtain a low-dimensional embedding for each submanifold
-

4.2.6 Extending Manifold Clustering to Riemannian Manifolds

We will now consider the case when we have data lying in m submanifolds of a Riemannian space. Similar to what is done in the Euclidean space, we assume that the data is distributed in a k -disconnected union of m k -connected submanifolds of \mathcal{M} . Notice that just as done in the calculation of the embedding coordinate, Algorithm 1 is independent of the Riemannian structure. Therefore, it is immediate to see that they are applicable to m submanifolds, both linear and nonlinear, of a Riemannian space. Algorithm 4 summarizes the dimensionality reduction and clustering algorithm for m submanifolds of a Riemannian space.

5 Experiments

5.1 Application and Experiments on SPSPD(3) [19, 21]

This section presents an application of the theory developed in Sect. 4 to the space of 3 by 3 symmetric positive semi-definite matrices SPSPD(3). It is well known [3, 15, 32, 46] that the traditional Euclidean distance is not the most appropriate metric for SPSPD matrices as they lie on a Riemannian symmetric space. An example of such data is the well-known structure tensor found in direct 2-D motion segmentation from the image intensities *without* extracting features such as optical flow or point correspondences. Under the assumption that all surfaces are Lambertian, the optical flow (u, v) between two images of a sequence is related to the image partial derivatives $\nabla I = (I_x, I_y, I_t)$ by $I_x u + I_y v + I_t = 0 \Rightarrow \nabla I^\top(u, v, 1) = 0$, where (x, y) denotes pixel location and t denotes time. Premultiplying by ∇I gives an equation of the form $(\nabla I \nabla I^\top)(u, v, 1) = 0$. This system of linear equations involves the spatial-temporal structure tensor $(\nabla I \nabla I^\top)$. SPSPD matrices also play an important role in Diffusion Tensor Imaging (DTI). DTI is a 3-D imaging technique

that measures the diffusion of water molecules in living tissues. Water diffusion is represented mathematically with a symmetric positive semi-definite tensor field $\mathbf{T} : \mathbb{R}^3 \rightarrow \text{SPSD}(3) \subset \mathbb{R}^{3 \times 3}$ that measures the diffusion in a direction $\mathbf{d} \in \mathbb{R}^3$ as $\mathbf{d}^\top \mathbf{T} \mathbf{d}$.

The goal is to automatically segment a set of SPSPD matrices $\{\mathbf{T}_j \in \text{SPSD}(r)\}_{j=1}^n$ into different clusters, where different groups correspond to different 2-D motions in a video or to different fiber bundles in DTI. There are many possible metrics in $\text{SPSD}(r)$ [3, 15, 28, 32, 46]. Each metric is derived from different geometrical, statistical or information-theoretic considerations. The question of which one is the best metric remains an active research area. The Riemannian metric proposed in [32] $\text{dist}_{\text{Riem}}(\mathbf{T}_i, \mathbf{T}_j) = \|\log(\mathbf{T}_i^{-\frac{1}{2}} \mathbf{T}_j \mathbf{T}_i^{-\frac{1}{2}})\|_F$, where $\|\cdot\|_F$ is the Frobenius norm and $\log(\cdot)$ is the matrix logarithm, is used here. For this metric, the exponential map is defined as $\exp_{\mathbf{T}_i}(\mathbf{V}) = \mathbf{T}_i^{\frac{1}{2}} \exp(\mathbf{T}_i^{-\frac{1}{2}} \mathbf{V} \mathbf{T}_i^{-\frac{1}{2}}) \mathbf{T}_i^{\frac{1}{2}}$, where $\exp(\cdot)$ is the matrix exponential and $\mathbf{V} \in T_{\mathbf{T}_i} \text{SPSD}(r)$. The logarithm map is $\overrightarrow{\mathbf{T}_i \mathbf{T}_j} = \log_{\mathbf{T}_i}(\mathbf{T}_j) = \mathbf{T}_i^{\frac{1}{2}} \log(\mathbf{T}_i^{-\frac{1}{2}} \mathbf{T}_j \mathbf{T}_i^{-\frac{1}{2}}) \mathbf{T}_i^{\frac{1}{2}}$. The Gram matrix is $C_i(j, l) = \langle \log_{\mathbf{T}_i}(\mathbf{T}_j), \log_{\mathbf{T}_i}(\mathbf{T}_l) \rangle_{\mathbf{x}_i} = \text{trace}(\log(\mathbf{T}_i^{-\frac{1}{2}} \mathbf{T}_j \mathbf{T}_i^{-\frac{1}{2}}) \log(\mathbf{T}_i^{-\frac{1}{2}} \mathbf{T}_l \mathbf{T}_i^{-\frac{1}{2}}))$. The geodesic linear interpolation $\hat{\mathbf{T}}_{\text{Riem}, i}$ of $\{\mathbf{T}_j \in \text{SPSD}(r)\}_{j=1}^n$ about \mathbf{T}_i with weights W_{i1}, \dots, W_{in} is given by $\mathbf{T}_i^{\frac{1}{2}} \exp(\sum_{j=1}^n W_{ij} \log(\mathbf{T}_i^{-\frac{1}{2}} \mathbf{T}_j \mathbf{T}_i^{-\frac{1}{2}})) \mathbf{T}_i^{\frac{1}{2}}$.

Our algorithm is tested on 2-D motion segmentation from two consecutive frames of video sequences [21]. The spatial-temporal structure tensor is $\mathbf{T} = K * (\nabla I \nabla I^\top)$, where $*$ is the convolution operator, K is a smoothing kernel (the Gaussian kernel is commonly used), and $\nabla I = (I_x, I_y, I_t)$ is the spatial-temporal image gradient. We use the same data set as in [12]. Figure 2 shows two examples of moving patches of homogeneously textured wallpaper in which the different regions cannot be distinguished on the sole basis of appearance. This is because the input frames contain regions with the same intensity and texture with no clear edges or corners. Thus, all results are obtained exclusively from the motion information. Figure 2(a) contains two regions, the text region with the word “UCLA” and the background. In Fig. 2(c),

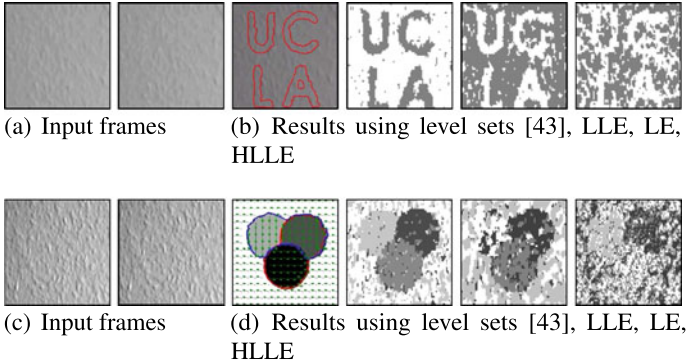


Fig. 2 2-D motion segmentation using the structure tensor [21]

there are three overlapping circles, each with its own motion, and the background. As shown in Figs. 2(b) and 2(d), LLE yields the best results among all the algebraic methods, distinguishing the text “UCLA” and the three circles. As none of the NLDR methods incorporates a smoothness constraint (as done in level set methods), it is of no surprise that the level set method produces a cleaner segmentation. Nevertheless, it is immediate that our algorithm provides a very good initialization for iterative techniques such as level sets.

The next set of experiments is done on real video sequences [21]. The first video involves a camera tracking a car going along a road, as shown in Fig. 3(a). There are three different motion groups found in the two consecutive frames. The first group contains mostly the pixels of the car, the second group contains the background pixels where the camera movement is apparent (e.g., edges and corners), and the last group contains the background pixels with the aperture problem (e.g., middle of the road). The second video of a car is taken with a stationary camera, as shown in Fig. 3(c). There are two different motion groups in this case, the first group being the car and the second group being the background. The last video, shown in Fig. 3(e), is taken from the Hamburg Taxi sequence. In this dataset, the moving taxi forms the first group and the stationary background forms the second group. From Figs. 3(b), 3(d), and 3(f), it is clear that LLE is able to segment the different groups, LE gives a reasonable segmentation but suffers from artifacts, whereas the performance of HLLE performance is poor.

Our algorithm is also tested in the segmentation of the corpus callosum and the cingulum from real DTI data [19]. The size of the entire DTI volume of the brain is

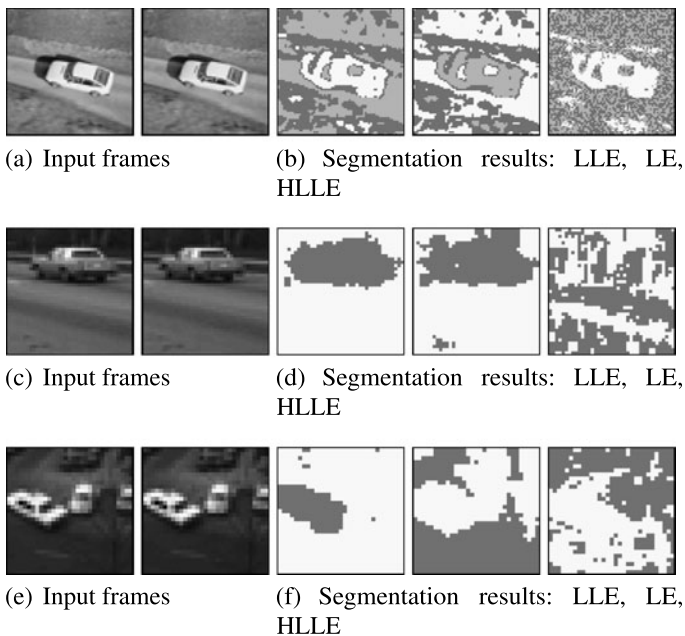


Fig. 3 2-D motion segmentation on real video sequences [21]

$128 \times 128 \times 58$ voxels and the voxel size is $2 \times 2 \times 2$ mm. From the visualization of the tensor data, we know the approximate location of each cingulum bundle in the left and right hemispheres. Hence, we reduce the input volume to the algorithm by focusing in this location. In addition, we also mask out voxels with fractional anisotropy below a threshold of 0.2 in order to separate white matter from the rest of the brain. Also, tensors at adjacent voxels within a fiber bundle are similar (similar eigenvectors and eigenvalues), while tensors at distant voxels could be very different, even if they lie in the same bundle. In order to account for this fact, the k NN $\{\mathbf{D}(x_j)\}_{j=1}^k$ of a tensor $\mathbf{D}(x_i)$ at x_i subject to $\|x_j - x_i\| \leq R$ is chosen. The value of the spatial radius is set to $R = 10$ and the number of nearest neighbors to $k = 30$.

Figure 4 shows the results of the left hemisphere. Figure 4(a) shows the sagittal slices used and the ellipsoid visualization of the tensors. The corpus callosum is the bundle with the red tensors pointing out of the plane and resembles the letter ‘C’. The cingulum, which is significantly smaller, is the bundle left to the corpus callosum with the green tensors oriented vertically. The corpus callosum and the cingulum are clustered around different centers. Figure 4(b) shows the results of LLE. The corpus callosum forms a distinct cluster (in red). Figure 4(c) shows the results of LE. Even though it appears that the cingulum forms a distinct group, the corpus callosum is merged into the same group as the tensors in the background. HLLE (not shown) failed to produce any reasonable segmentation of the fiber bundles.

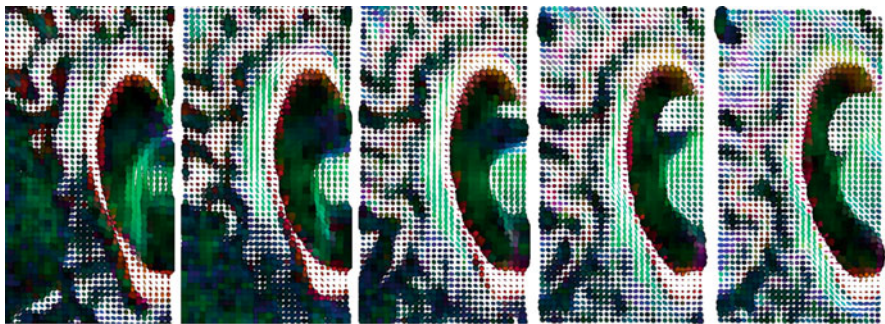
As our algorithm does not incorporate any smoothness constraint, the segmentation is noisier compared to energy minimization methods such as in [30]. However, for the segmentation of the cingulum bundle in [30], a significant effort was required to manually remove voxels in the corpus callosum before running their respective algorithms. Our algorithm, on the other hand, is automatic. Hence, an immediate use of our algorithm is that the output could be used as an automatic initialization for such algorithms.

5.2 Application and Experiments on the Space of Probability Density Functions

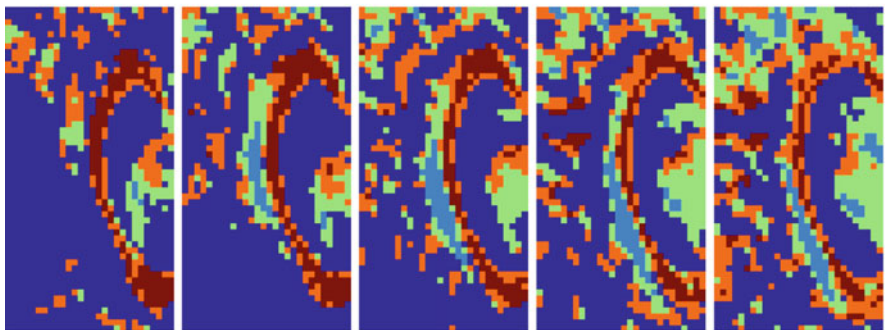
This section present an application of the theory developed in Sect. 4 to the space of probability density functions. The class of constrained non-negative continuous functions under study here is the set of pdfs defined below. Without loss of generality, we can assume that these functions are defined on the interval $[0, T]$. Therefore, the set \mathcal{P} of pdfs is given by

$$\mathcal{P} = \left\{ p : [0, T] \rightarrow \mathbb{R} | \forall s, p(s) \geq 0, \int_0^T p(s) ds = 1 \right\}. \quad (18)$$

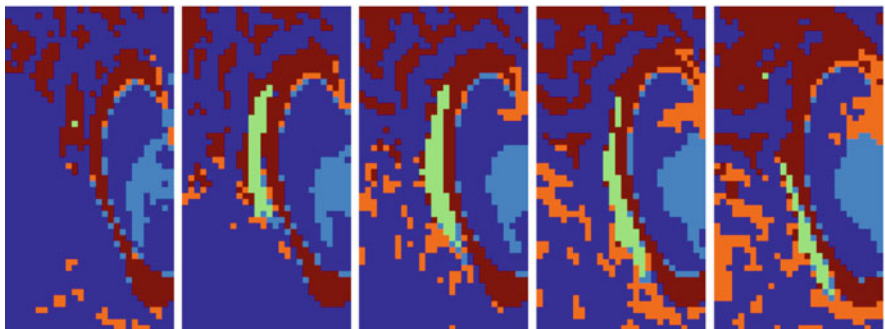
The question of how to regard the space of pdfs as a differential manifold endowed with a Riemannian metric and a family of affine connections has a long history behind it. Nevertheless, it remains an active and important research area. Treating statistical structures as geometric structures has the advantage that geometric



(a) Ellipsoid visualization of the tensors in five sagittal slices



(b) Segmentation results of the five sagittal slices using LLE



(c) Segmentation results of the five sagittal slices using LE

Fig. 4 Segmenting the corpus callosum and the cingulum [19]. The *first row* shows the visualization of the data in five sagittal slices and the tensor at each voxel is represented by an ellipsoid. The *second row* shows the clustering result given by LLE. The *third row* shows the clustering result given by LE

structures remain invariant under coordinate transforms. Rao [34] first introduces the Riemannian structure formed by the statistical manifold where each point in the manifold denotes a pdf. In addition, [34] also shows that the *Fisher–Rao* metric determines a Riemannian metric. The Fisher–Rao metric is later shown to be the

unique intrinsic metric on the statistical manifold in [9]. This study of probability and information via differential geometry is known as *information geometry*. The reader is referred to the seminal work of [2] for a complete description.

We will consider the manifold \mathcal{P} of pdfs on the interval $[0, T]$. For any point $p_i \in \mathcal{P}$, the Fisher–Rao metric is defined as

$$\langle q_j, q_k \rangle_{p_i} = \int_0^T q_j(s) q_k(s) \frac{1}{p_i(s)} ds, \quad (19)$$

where $q_j, q_k \in T_{p_i}(\mathcal{P})$ are tangent vectors and $T_{p_i}(\mathcal{P})$ is the set containing the functions tangent to \mathcal{P} at the point p_i . This representation turns out to be extremely difficult to work with as ensuring the geodesic between two elements lies on \mathcal{P} is not easy [41].

Even though the space \mathcal{P} turns out to be difficult to work with, we know that it is not the only possible representation for pdfs and in addition, we also know that the Fisher–Rao metric is the only metric that is invariant to re-parameterizations (essentially coordinate transforms) of the functions [9]. There are many different re-parameterizations of pdfs that are equivalent representations. Depending on the representation, the resulting Riemannian structure can have varying degrees of complexity and numerical techniques may be required to compute geodesics on the manifold. Therefore, the natural question to ask now is, is it possible to use a re-parameterization such that the resulting manifold is simple and the Riemannian operations are easy, preferably closed-form, to compute? Once an efficient representation is found, the corresponding Fisher–Rao metric, which depends on the tangent vector, will then be used as the Riemannian metric. In a recent work [41], it is proved that by using the square-root representation, the resulting manifold is a unit sphere in a Hilbert space with the Fisher–Rao metric being the usual \mathbb{L}^2 metric. Therefore, the various Riemannian operations such as geodesics, exponential maps, logarithmic maps are available in closed form. This is one of the most efficient representation found to date.

The square-root density function is defined as $\psi = \sqrt{p}$, where ψ is assumed to be nonnegative to ensure uniqueness. The space of such functions is defined as:

$$\Psi = \left\{ \psi : [0, T] \rightarrow \mathbb{R} \mid \forall s, \psi(s) \geq 0, \int_0^T \psi^2(s) ds = 1 \right\}. \quad (20)$$

From (20), it is easy to see that the functions ψ lie on a unit sphere. In addition, Ψ forms a convex subset of the unit sphere. The advantage of choosing the square-root density becomes immediately obvious, as many of the Riemannian expressions for the unit sphere are well-known and closed-form. By making use of the representation in (20), we can rewrite (19) and obtain the Fisher–Rao metric as

$$\langle v_j, v_k \rangle_{\psi_i} = \int_0^T v_j(s) v_k(s) ds, \quad (21)$$

where $v_j, v_k \in T_{\psi_i} \Psi$ are tangent vectors. Now, for any two functions $\psi_i, \psi_j \in \Psi$, the geodesic distance between these two points on a unit sphere is simply the angle

between them, that is,

$$\text{dist}(\psi_i, \psi_j) = \cos^{-1} \langle \psi_i, \psi_j \rangle = \cos^{-1} \left(\int_0^T \psi_i(s) \psi_j(s) ds \right), \quad (22)$$

where $\langle \cdot, \cdot \rangle$ is the normal dot product between points in the sphere under the \mathbb{L}^2 metric.

From the differential geometry of the sphere, the exponential map is defined as

$$\exp_{\psi_i}(\mathbf{v}) = \cos(\|\mathbf{v}\|_{\psi_i}) \psi_i + \sin(\|\mathbf{v}\|_{\psi_i}) \frac{\mathbf{v}}{\|\mathbf{v}\|_{\psi_i}}, \quad (23)$$

where $\mathbf{v} \in T_{\psi_i}(\Psi)$ is a tangent vector at ψ_i and $\|\mathbf{v}\|_{\psi_i} = \sqrt{\langle \mathbf{v}, \mathbf{v} \rangle_{\psi_i}} = (\int_0^T \mathbf{v}(s) \mathbf{v}(s) ds)^{\frac{1}{2}}$. In order to ensure that the exponential map is a bijective function, we restrict $\|\mathbf{v}\|_{\psi_i} \in [0, \pi]$. The logarithm map from ψ_i to ψ_j is then given by

$$\overrightarrow{\psi_i \psi_j} = \log_{\psi_i}(\psi_j) = \frac{\mathbf{u}}{(\int_0^T \mathbf{u}(s) \mathbf{u}(s) ds)^{\frac{1}{2}}} \cos^{-1} \langle \psi_i, \psi_j \rangle, \quad (24)$$

with $\mathbf{u} = \psi_j - \langle \psi_i, \psi_j \rangle \psi_i$.

We test the algorithm in the segmentation of different textures [20]. From the Columbia-Utrecht Reflectance and Texture Database (CURET) found at <http://www1.cs.columbia.edu/CAVE/software/curet/>, we obtain samples of different textures and each grayscale image contains only one texture. In order to construct a histogram that reflects the texture statistics in an image, we will calculate what is commonly known as *textons* [44]. This is done by first applying a filter bank to all images in the training set. We use the Schmid [37] filter banks shown in Fig. 5. This will provide us with a feature vector $f(x, y)$ of dimension 13 at each pixel. Next, we apply k -means to all the vectors in the entire dataset to get 30 cluster centers, also known as the textons. For each image in the dataset, we then compute a histogram that contains the number of pixels corresponding to each one of these 30 bins. This is done by assigning a pixel (x, y) to bin i if the feature vector $f(x, y)$ is closest to cluster center $i = 1, \dots, 30$, according to the Euclidean distance in \mathbb{R}^{13} .

We test our algorithm on 4 sets of data containing 2 different textures each. There are 92 images in each texture class. In these experiments, the number of nearest neighbors is set to 10. Figure 6 shows these 4 sets with a typical example of the 2 different textures and the corresponding histograms in each set. Table 2 shows the misclustering percentage of LLE and LE for each set.

Finally, we test our algorithm on a set of data containing 3 different textures. Figure 7 shows a typical example of the different textures and the corresponding histograms in each set. The error produced by LLE in clustering is 5.43% whereas LE is significantly higher at 30.07%.

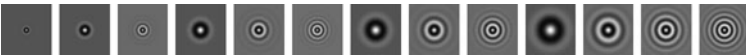


Fig. 5 Schmid filter bank that we use to generate the textons and in turn the histograms

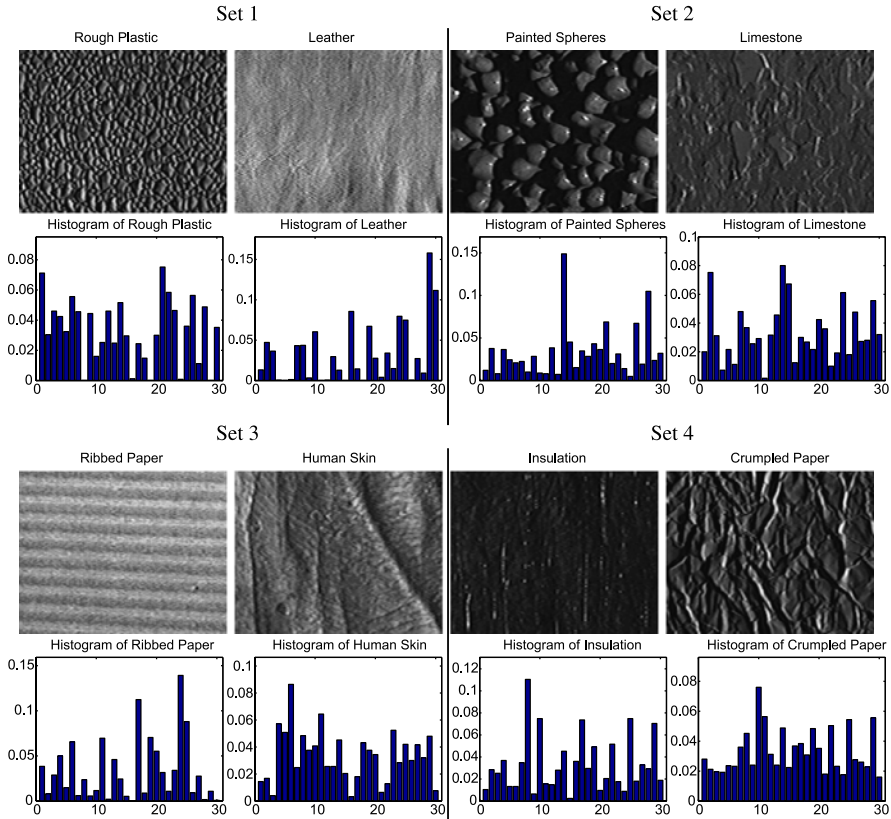


Fig. 6 Textures and corresponding histograms used in the two-class clustering experiments [20]

Table 2 Misclustering rates in % for two-class segmentation

| Algorithm | Set 1 | Set 2 | Set 3 | Set 4 |
|----------------|-------|-------|-------|-------|
| Riemannian LLE | 0 | 0 | 1.63 | 0 |
| Riemannian LE | 0 | 0 | 19.68 | 22.9 |

6 Conclusion and Open Research Problems

An algorithm for simultaneous NLDR and clustering of data sampled from multiple submanifolds of a Riemannian manifold is presented. We focused our investigation on the three NLDR algorithms, which computes a low-dimensional embedding from the eigenvectors of a matrix M that depends on the local properties of the data. It is important for the user to note that, it is possible that these algorithms become degenerate if the construction of a matrix M , which captures the local geometry of the data, is done in the presence of linear manifolds. Presently, there are several open research problems. First of all, notice that the various Riemmanian operations

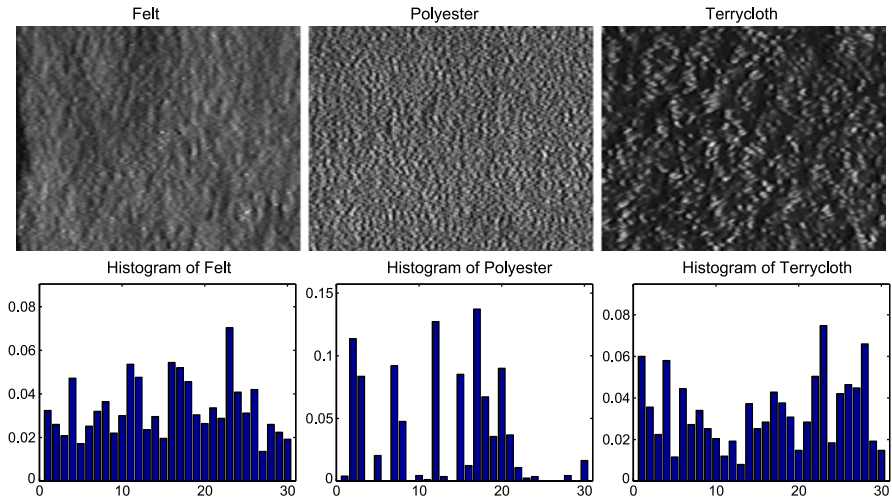


Fig. 7 Textures and corresponding histograms used in the three-class clustering experiment [20]

are assumed to be known and closed-form. This is not true for generic cases and there is a question of how one would be able to perform a similar analysis when the Riemannian operations need to be solved in an iterative manner. The next open problem that future research efforts are focusing on is to address the assumption that data lying on different manifolds do not intersect each other. Finally, there is also a need to construct a large-scale version of the algorithms presented in order to handle large datasets.

References

1. Agarwal, S., Lim, J., Zelnik-Manor, L., Perona, P., Kriegman, D., Belongie, S.: Beyond pairwise clustering. In: *Computer Vision and Pattern Recognition*, vol. 2, pp. 838–845 (2005)
2. Amari, S.: *Differential-Geometrical Methods in Statistics*. Springer, Berlin (1985)
3. Arsigny, V., Fillard, P., Pennec, X., Ayache, N.: Log-Euclidean metrics for fast and simple calculus on diffusion tensors. *Magn. Reson. Med.* **56**, 411–421 (2006)
4. Barbará, D., Chen, P.: Using the fractal dimension to cluster datasets. In: *KDD '00: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 260–264. ACM, New York (2000)
5. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. In: *Advances in Neural Information Processing Systems*, pp. 585–591. MIT Press, Cambridge (2002)
6. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.* **15**(6), 1373–1396 (2003)
7. Brand, M., Huang, K.: A unifying theorem for spectral embedding and clustering. In: *International Workshop on Artificial Intelligence and Statistics* (2003)
8. Burges, C.: Geometric methods for feature extraction and dimensional reduction—a guided tour. In: *The Data Mining and Knowledge Discovery Handbook*, pp. 59–92. Kluwer Academic, Norwell (2005)

9. Cencov, N.N.: Statistical decision rules and optimal inference. In: *Translations of Mathematical Monographs*, vol. 53. AMS, Providence (1982)
10. Chen, G., Lerman, G.: Spectral curvature clustering, SCC. *Int. J. Comput. Vis.* **81**(3), 317–330 (2009)
11. Cox, T.F., Cox, M.A.A.: *Multidimensional Scaling*. Chapman & Hall, London (1994)
12. Cremers, D., Soatto, S.: Motion competition: a variational framework for piecewise parametric motion segmentation. *Int. J. Comput. Vis.* **62**(3), 249–265 (2005)
13. do Carmo, M.P.: *Riemannian Geometry*. Birkhäuser, Boston (1992)
14. Donoho, D., Grimes, C.: Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. *Natl. Acad. Sci.* **100**(10), 5591–5596 (2003)
15. Fletcher, P.T., Joshi, S.: Riemannian geometry for the statistical analysis of diffusion tensor data. *Signal Process.* **87**(2), 250–262 (2007)
16. Frechet, M.: Les elements aleatoires de nature quelconque dans un espace distance. *Ann. Inst. Henri Poincare* **10**, 235–310 (1948)
17. Gionis, A., Hinneburg, A., Papadimitriou, S., Tsaparas, P.: Dimension induced clustering. In: *KDD '05: Proceeding of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp. 51–60. ACM, New York (2005)
18. Goh, A., Vidal, R.: Segmenting motions of different types by unsupervised manifold clustering. In: *Conference on Computer Vision and Pattern Recognition* (2007)
19. Goh, A., Vidal, R.: Segmenting fiber bundles in diffusion tensor images. In: *European Conference on Computer Vision* (2008)
20. Goh, A., Vidal, R.: Unsupervised Riemannian clustering of probability density functions. In: *European Conference on Machine Learning* (2008)
21. Goh, A., Vidal, R.: Clustering and dimensionality reduction on Riemannian manifolds. In: *Conference on Computer Vision and Pattern Recognition*, pp. 238–250 (2008)
22. Govindu, V.: A tensor decomposition for geometric grouping and segmentation. In: *Computer Vision and Pattern Recognition*, vol. 1, pp. 1150–1157 (2005)
23. Ham, J., Lee, D.D., Mika, S., Schölkopf, B.: A kernel view of the dimensionality reduction of manifolds. In: *International Conference on Machine learning*, vol. 69, p. 47 (2004)
24. Haro, G., Randall, G., Sapiro, G.: Translated poisson mixture model for stratification learning. *Int. J. Comput. Vis.* **80**, 358–374 (2008)
25. Ho, J., Yang, M.H., Lim, J., Lee, K.C., Kriegman, D.: Clustering appearances of objects under varying illumination conditions. In: *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 11–18 (2003)
26. Hotelling, H.: Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol.* **24**, 417–441 (1933)
27. Karcher, H.: Riemannian center of mass and mollifier smoothing. *Commun. Pure Appl. Math.* **30**(5), 509–541 (1977)
28. Kindlmann, G., Estepar, R.S.J., Niethammer, M., Haker, S., Westin, C.F.: Geodesic-loxodromes for diffusion tensor interpolation and difference measurement. In: *Medical Image Computing and Computer-Assisted Intervention* (2007)
29. Levina, E., Bickel, P.J.: Maximum likelihood estimation of intrinsic dimension. In: *NIPS* (2004)
30. Melonakos, J., Mohan, V., Niethammer, M., Smith, K., Kubicki, M., Tannenbaum, A.: Finsler tractography for white matter connectivity analysis of the cingulum bundle. In: *Medical Image Computing and Computer-Assisted Intervention* (2007)
31. Mordohai, P., Medioni, G.G.: Unsupervised dimensionality estimation and manifold learning in high-dimensional spaces by tensor voting. In: *International Joint Conference on Artificial Intelligence*, pp. 798–803 (2005)
32. Pennec, X., Fillard, P., Ayache, N.: A Riemannian framework for tensor computing. *Int. J. Comput. Vis.* **66**(1), 41–46 (2006)
33. Polito, M., Perona, P.: Grouping and dimensionality reduction by locally linear embedding. In: *Advances in Neural Information Processing Systems*. MIT Press, Cambridge (2002)
34. Rao, C.R.: Information and accuracy attainable in the estimation of statistical parameters. *Bull. Calcutta Math. Soc.* **37**, 81–89 (1945)

35. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**(5500), 2323–2326 (2000)
36. Roweis, S., Saul, L.: Think globally, fit locally: unsupervised learning of low dimensional manifolds. *J. Mach. Learn. Res.* **4**, 119–155 (2003)
37. Schmid, C.: Constructing models for content-based image retrieval. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2001)
38. Schölkopf, B., Smola, A.: *Learning with Kernels*. MIT Press, Cambridge (2002)
39. Sha, F., Saul, L.: Analysis and extension of spectral methods for nonlinear dimensionality reduction. In: *International Conference on Machine learning*, pp. 784–791 (2005)
40. Souvenir, R., Pless, R.: Manifold clustering. In: *IEEE International Conference on Computer Vision*, vol. I, pp. 648–653 (2005)
41. Srivastava, A., Jermyn, I., Joshi, S.: Riemannian analysis of probability density functions with applications in vision. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2007)
42. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. *Science* **290**(5500), 2319–2323 (2000)
43. Tipping, M., Bishop, C.: Mixtures of probabilistic principal component analyzers. *Neural Comput.* **11**(2), 443–482 (1999)
44. Varma, M., Zisserman, A.: A statistical approach to texture classification from single images. *Int. J. Comput. Vis.* **62**(1–2), 61–81 (2005)
45. Vidal, R., Ma, Y., Sastry, S.: Generalized principal component analysis, GPCA. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(12), 1–15 (2005)
46. Wang, Z., Vemuri, B.: An affine invariant tensor dissimilarity measure and its applications to tensor-valued image segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 228–233 (2004)
47. Weinberger, K.Q., Saul, L.: Unsupervised learning of image manifolds by semidefinite programming. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 988–955 (2004)
48. Wright, J., Ma, Y.: Dense error correction via l_1 -minimization. *IEEE Trans. Inf. Theory* **56**(7), 3540–3560 (2010). doi:[10.1109/TIT.2010.2048473](https://doi.org/10.1109/TIT.2010.2048473)
49. Yan, J., Pollefeys, M.: A factorization approach to articulated motion recovery. In: *IEEE Conference on Computer Vision and Pattern Recognition*, vol. II, pp. 815–821 (2005)
50. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM J. Sci. Comput.* **26**(1), 313–338 (2005)

Machine Learning for Vision-Based Motion Analysis
Theory and Techniques

Wang, L.; Zhao, G.; Cheng, L.; Pietikäinen, M. (Eds.)

2011, XIV, 372 p., Hardcover

ISBN: 978-0-85729-056-4