Geometric Data Analysis Based on Manifold Learning with Applications for Image Understanding

Gastão Florêncio Miranda Junior
Department of Mathematics
Federal University of Sergipe
Aracaju, Brazil
Email: gastao@mat.ufs.br

Carlos Eduardo Thomaz
Department of Electrical Engineering
FEI
São Bernardo do Campo, Brazil
Email: cet@fei.edu.br

Gilson Antonio Giraldi
Depart. of Math. and Comput. Methods
National Laboratory for
Scientific Computing (LNCC)
Petrópolis, Brazil
Email: gilson@lncc.br

***Abstract*—Nowadays, pattern recognition, computer vision, signal processing and medical image analysis, require the managing of large amount of multidimensional image databases, possibly sampled from nonlinear manifolds. The complex tasks involved in the analysis of such massive data lead to a strong demand for nonlinear methods for dimensionality reduction to achieve efficient representation for information extraction. In this avenue, manifold learning has been applied to embed nonlinear image data in lower dimensional spaces for subsequent analysis. The result allows a geometric interpretation of image spaces with relevant consequences for data topology, computation of image similarity, discriminant analysis/classification tasks and, more recently, for deep learning issues. In this paper, we firstly review Riemannian manifolds that compose the mathematical background in this field. Such background offers the support to set up a data model that embeds usual linear subspace learning and discriminant analysis results in local structures built from samples drawn from some unknown distribution. Afterwards, we discuss topological issues in data preparation for manifold learning algorithms as well as the determination of manifold dimension. Then, we survey dimensionality reduction techniques with particular attention to Riemannian manifold learning. Besides, we discuss the application of concepts in discrete and polyhedral geometry for synthesis and data clustering over the recovered Riemannian manifold with emphasis in face images in the computational experiments. Next, we discuss promising perspectives of manifold learning and related topics for image analysis, classification and relationships with deep learning methods. Specifically, we discuss the application of foliation theory, discriminant analysis and kernel methods in curved spaces. Besides, we take differential geometry in manifolds as a paradigm to discuss deep generative models and metric learning algorithms.**

***Keywords*-manifold learning; statistical learning; Riemannian manifolds; image analysis; deep learning**

I. Introduction

Many areas such as computer vision, signal processing and medical image analysis require the managing of data sets with a large number of features or dimensions. Therefore, dimensionality reduction may be necessary in order to discard redundancy and reduce the computational cost of further operations [1]–[3].

We may distinguish two major classes of dimensionality reduction methods: linear and nonlinear. The former includes the classical principal component analysis (PCA), factor analysis (FA) [3], multidimensional scaling (MDS) [4], [5] and projection pursuit (PP) [3], [6]. Linear techniques seek for new variables that obey some optimization criterium and can be expressed as linear combination of the original ones. That's why they fail if the input data has curved or nonlinear structures.

Nonlinear dimensionality reduction methods include kernel approaches, like kernel PCA (KPCA), kernel LDA (KLDA) and kernel Fisher discriminant analysis (KFD). These techniques map the original input data into a feature space by a (global) nonlinear mapping, where inner products in the feature space can be computed by a kernel function in the input space without explicitly knowing the nonlinear mapping [7]–[9].

A more complete scenario in nonlinear dimensionality reduction is the one that encompasses samples from different classes obtained from an unknown data manifold. We can use manifold learning techniques to estimate the intrinsic manifold topology and geometry in order to address the problem of dimensionality reduction. Also, we can estimate local tangent spaces using linear subspace learning techniques. From such viewpoint emerges a data model that embeds usual linear dimensionality reduction and discriminant analysis results in local structures built from samples drawn from some unknown distribution. In this way, we recognize manifold learning as a background to image analysis techniques based on the geometric structure of high-dimensional image databases [10], [11]. The so called *geometric data analysis* is a combination of differentiable manifold elements, data representation techniques, machine and statistical learning methods, for extracting meaningful information from image spaces [1], [2], [12], [13].

From a theoretical viewpoint, manifold learning is based on the assumption that the database samples (or their features) lie on a low-dimensional manifold $\mathcal{M}$ embedded in a high-dimensional space [14]. In the specific case of Riemannian manifold learning (RML) techniques [15]–[17], there is also the assumption that the low-dimensional manifold is a Riemannian one; that is, it is equipped with an inner product

that varies smoothly from point to point [18], [19].

Therefore, we need to learn the underlying intrinsic manifold geometry in order to address the problem of dimensionality reduction. Thus, instead of seeking for an optimum linear subspace, like performed for linear techniques [3], the manifold learning methods try to discover an embedding procedure that describes the intrinsic similarities of the data [20]. In order to implement this solution, manifold learning approaches take the samples of a database $\mathcal{D} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\} \subset \mathbb{R}^D$ and perform the following steps [11], [17], [21]: (a) Recover the data topology; (b) Determination of the manifold dimension d; (c) Construction of a neighborhood system; (d) Computing the embedding or local parameterizations associated to the neighborhood system. The former is a global map $f : \mathcal{M} \to \mathbb{R}^d$ while the latter is a family of local coordinate systems $\{(U_\alpha, \varphi_\alpha)\}_{\alpha \in I}$, where I is an index set, $U_\alpha \subset \mathbb{R}^d$ and $\varphi_\alpha : U_\alpha \to \mathcal{M}$.

Each one of the above steps has specific issues that have been addressed in different ways along the literature. So, we analyse the problems related to data topology, and manifold dimension estimation [2], [22], [23]. Then, we review some traditional algorithms in manifold learning (Local Tangent Space Alignment (LTSA), Locally Linear Embedding (LLE) and Isomap). Besides, we discuss the local Riemannian manifold learning (LRML) technique in order to present specific issues in distance preservation in the lower dimensional data representation. We give special attention to application of concepts in discrete and polyhedral geometry for synthesis and data clustering over the manifold [17], [24]. In the experimental results we explore the LRML framework for face image synthesis and analysis. Lastly, we discuss opened issues exploring foliation theory and the topological structure of the face image space. Also, the relationship between manifold and deep learning starts to be investigated to identify the factors that contribute to the success of deep hierarchical representations of the data [25]. We focus on generative adversarial networks and deep metric learning [26], [27]. Besides, discriminant analysis and kernel methods in manifolds will be discussed.

This paper is organized as follows. Section II gives the geometric background. Then, section III offers a geometric viewing of linear and nonlinear dimensionality reduction methods. Section IV discusses concerns about data topology and dimensionality estimation in manifold learning. Next, sections V and VI survey linear and manifold learning frameworks for dimensionality reduction. Afterwards, in section VII we embed the concepts of synthesis and clustering in a discrete geometry framework. The section VIII shows the experimental results using LRML technique. In section IX we discuss promising perspectives for manifold learning in image analysis and relationships with deep learning methods. Finally, we end with the conclusions in section X.

II. Differentiable Manifolds Elements

In this paper, the normal uppercase symbols represent matrices, data sets and subspaces (P, U, D, S, etc.); the bold lowercase symbols represent data points and vectors, such as $\mathbf{p}$, $\mathbf{x}$, $\mathbf{y}$; and the normal Greek lowercase symbols represent scalar numbers (λ, α, etc.).

A differentiable manifold of dimension d is a set, denoted in Figure 1 as $\mathcal{M}^d$ and a family of one-to-one functions $\{\varphi_\alpha\}_{\alpha \in I}$, with I an index set, $\varphi_\alpha : U_\alpha \subset \mathbb{R}^d \to \mathcal{M}^d$ where U_α is an open set of $\mathbb{R}^d$, such that [28]:

1) $\cup_{\alpha \in I} \varphi_\alpha (U_\alpha) = \mathcal{M}^d$.

2) For every $\alpha, \beta \in I$, with $\varphi_\alpha (U_\alpha) \cap \varphi_\beta (U_\beta) = W \neq \emptyset$, the sets $\varphi_\alpha^{-1}(W)$ and $\varphi_\beta^{-1}(W)$ are open sets in $\mathbb{R}^d$ and the chart transition $\varphi_\beta^{-1} \circ \varphi_\alpha : \varphi_\alpha^{-1}(W) \to \varphi_\beta^{-1}(W)$ are differentiable functions.

3) The family $\{(U_\alpha, \varphi_\alpha)\}$ is maximal respect to properties (1) and (2).

The properties (1) and (2) define the differential structure of $\mathcal{M}^d$. They allow to generate a natural topology over $\mathcal{M}^d$: a set $A \subset \mathcal{M}^d$ is an open set of $\mathcal{M}^d$ if $\varphi_\alpha^{-1}(A \cap \varphi_\alpha (U_\alpha))$ is an open set of $\mathbb{R}^d$, $\forall \alpha$.

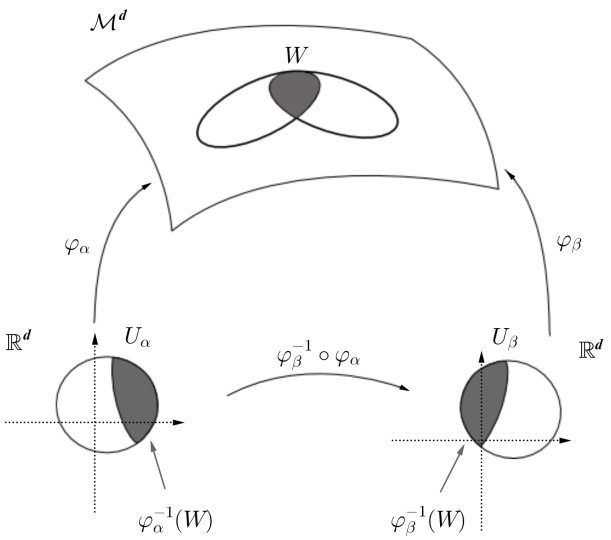


Figure 1. Coordinates change and differentiable manifold elements.

Let $\mathbf{p} \in \varphi_\alpha (U_\alpha)$ and $\varphi_\alpha^{-1}(\mathbf{p}) = (x_1(\mathbf{p}), ..., x_n(\mathbf{p}))$. So, $\varphi_\alpha (U_\alpha)$ is called a coordinate neighborhood and the pair $(U_\alpha, \varphi_\alpha)$ a local parametrization or system of coordinates for $\mathcal{M}^d$ in $\mathbf{p}$. The Figure 1 pictures the main elements of items (1)-(3) representing also the change of coordinate systems in the item (2). If $\varphi_\beta^{-1} \circ \varphi_\alpha \in C^k$, with $k \geq 1$, then we say that $\mathcal{M}^d$ is a C^k-differentiable manifold, or simply C^k-manifold. If $k = \infty$, $\mathcal{M}^d$ is called a smooth manifold. Besides, we say that $\mathcal{N}$ is a submanifold of $\mathcal{M}^d$ if $\mathcal{N} \subset \mathcal{M}^d$ and $\mathcal{N}$ is also a differentiable manifold.

Let $\mathcal{M}^d$ be a C^k-manifold of dimension d with local coordinates $\varphi : U \subset \mathbb{R}^d \to \mathcal{M}^d$, at a point $\mathbf{p} = \varphi(\mathbf{x})$.

A tangent vector $\mathbf{v}$ to $\mathcal{M}^d$ at $\mathbf{p}$ can be expressed in the local coordinates $\mathbf{x} = (x_1, ..., x_n)$ as:

$$\mathbf{v} = \sum_{i=1}^{d} \left(v_i \frac{\partial}{\partial x_i} \right). \tag{1}$$

where the vectors

$$B = \left\{ \frac{\partial}{\partial x_1}, ..., \frac{\partial}{\partial x_m} \right\}, \tag{2}$$

are defined by the local coordinates.

The set of all tangent vectors to $\mathcal{M}^d$ at $\mathbf{p}$ is called the tangent space to $\mathcal{M}^d$ at $\mathbf{p}$, and is denoted by $T_{\mathbf{p}}\left(\mathcal{M}^d\right)$. The vectors in the set (2) determine an natural basis for $T_{\mathbf{p}}\left(\mathcal{M}^d\right)$. The collection of all tangent spaces to $\mathcal{M}^d$ is the tangent bundle of the differentiable manifold $\mathcal{M}^d$:

$$TM = \bigcup_{\mathbf{p} \in \mathcal{M}^d} T_{\mathbf{p}}\left(\mathcal{M}^d\right). \tag{3}$$

A Riemannian manifold is a manifold $\mathcal{M}^d$ equipped with an inner product in each point $\mathbf{p}$ (bilinear, symmetric and positive definite form in the tangent space $T_{\mathbf{p}}\left(\mathcal{M}^d\right)$) that varies smoothly from point to point.

A geodesic in a Riemannian manifold $\mathcal{M}^d$ is a differentiable curve $\alpha : I \subset \mathbb{R} \rightarrow \mathcal{M}^d$ that is the shortest path between any two points $\mathbf{p}_1 = \alpha(t_1)$ and $\mathbf{p}_2 = \alpha(t_2)$ [29]. With this concept, we can define the geodesic distance between the points $\mathbf{p}_1$ and $\mathbf{p}_2$ as:

$$d_{\mathcal{M}^d}(\boldsymbol{p}_1, \boldsymbol{p}_2) = \int_{t_1}^{t_2} \sqrt{\left\langle \frac{d\alpha}{dt}, \frac{d\alpha}{dt} \right\rangle} dt. \tag{4}$$

We denote by $\alpha(s, \mathbf{q}, \frac{\mathbf{v}}{\|\mathbf{v}\|})$ the geodesic, parameterized by the arc length s, that pass to $\mathbf{q}$ at $s = 0$ with unitary tangent vector $\alpha\prime(0) = \frac{\mathbf{v}}{\|\mathbf{v}\|}$. Existence and uniqueness for geodesics can be demonstrated in a Riemannian manifold which allows to define the exponential map as follows.

Definition 1. Let the subset $\mathcal{U} \subset TM$, such that $\mathcal{U} = \{(\mathbf{q}, \mathbf{v});\ \mathbf{q} \in \mathcal{M}^d,\ \mathbf{v} \in T_{\mathbf{q}}(\mathcal{M}^d),\ ||\mathbf{v}|| < \epsilon\}$. Then, the function:

$$\begin{array}{rccl} \exp: & \mathcal{U} & \rightarrow & \mathcal{M}^d \\ & (\mathbf{q}, \mathbf{v}) & \mapsto & \exp(\mathbf{q}, \mathbf{v}) = \alpha\left(\|\mathbf{v}\|, \mathbf{q}, \frac{\mathbf{v}}{\|\mathbf{v}\|}\right) \end{array}$$

is well defined and is called exponential map in $\mathcal{U}$.

According to the definition, the geodesic distance from the point $\mathbf{q}$ to the point $\mathbf{z} = \alpha\left(\|\mathbf{v}\|, \mathbf{q}, \frac{\mathbf{v}}{\|\mathbf{v}\|}\right)$, that is, the arc length from $\mathbf{q}$ to $\mathbf{z}$, is exactly $d_{\mathcal{M}^d}(\mathbf{q}, \mathbf{z}) = \|\mathbf{v}\|$. We also use the notation $\exp(\mathbf{q}, \mathbf{v}) = \exp_{\mathbf{q}}(\mathbf{v})$ for the image of a vector $\mathbf{v} \in T_{\mathbf{q}}\left(\mathcal{M}^d\right)$ by the exponential map. The Gauss Lema assures that the exponential map is a local isometry; in other words, it is a diffeomorphism that preserves the inner product in $\mathcal{M}^d$ (a differentiable map f is a diffeomorphism if it is one-to-one and onto, and if the inverse f^{-1} is also differentiable). Therefore, we can define its inverse, called logmap.

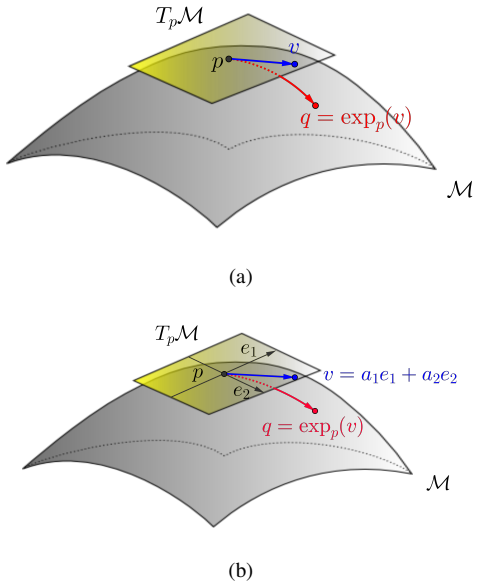


Figure 2. (a) Exponential map. (b) Riemann normal coordinates for a 2-dimensional manifold.

Once the exponential map is a local diffeomorphism at $T_{\mathbf{p}}(\mathcal{M}^d)$, $\forall \mathbf{p} \in \mathcal{M}^d$, if we map $T_{\mathbf{p}}(\mathcal{M}^d)$ with $\mathbb{R}^d$ the exponential map can be used to define a system of local coordinates for $\mathcal{M}^d$ in $\mathbf{p}$ as follows.

Definition 2. Let $\mathbf{p} \in \mathcal{M}^d$, and a orthonormal basis $\{\mathbf{e}_i\}$ in $T_{\mathbf{p}}(\mathcal{M}^d)$. If $\mathbf{q} = \exp_{\mathbf{p}}(\sum_i u_i \mathbf{e}_i)$ then we call $(u_1, u_2, \ldots, u_d)$ the Riemann normal coordinates of $\mathbf{q}$.

Conversely, once the exponential map is a diffeomorphism, its inverse (logmap) is well defined. So, if $logmap(\mathbf{q}) = \mathbf{v} \in T_{\mathbf{p}}(\mathcal{M}^d)$ then we can project $\mathbf{v}$ in $\{\mathbf{e}_i\}$ to get the Riemann normal coordinates. The Figure 2 illustrates these concepts for 2-dimensional manifold (a regular surface in $\mathbb{R}^3$). The Figure 2.(a) describes the exponential map computed at a pair $(\mathbf{p}, \mathbf{v})$ given the point $\mathbf{q} = \exp_{\mathbf{p}}(\mathbf{v})$ and the Figure 2.(b) the associated system of local coordinates in which the point $\mathbf{q} \in \mathcal{M}^d$ has Riemann normal coordinates (a_1, a_2).

Let $\mathcal{M}^d$ and $\mathcal{N}^k$ be submanifolds of a manifold $\mathcal{Y}$. We say that $\mathcal{M}^d$ and $\mathcal{N}^k$ are transverse, if at every point $\mathbf{p} \in \mathcal{M}^d \cap \mathcal{N}^k$ we verify the property:

$$T_{\mathbf{p}}\left(\mathcal{M}^d\right) + T_{\mathbf{p}}\left(\mathcal{N}^k\right) = T_{\mathbf{p}}\left(\mathcal{Y}^d\right), \tag{5}$$

where '+' indicates sum o vector spaces.

III. Geometry and Dimensionality Reduction

In general, the dimension of the original data space is very large which requires some dimensionality reduction

technique in order to discard redundancies. So, let an input database:

$$\mathcal{D} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_N\} \subset \mathbb{R}^D. \quad (6)$$

The key assumption in manifold learning methods is that the points in $\mathcal{D}$ are samples of a lower-dimensional manifold $\mathcal{M}^d$, with $d < D$, embedded in the high-dimensional space $\mathbb{R}^D$. For instance, in the case of Figure 3 the curve represents an one-dimensional manifold embedded in a two-dimensional space.

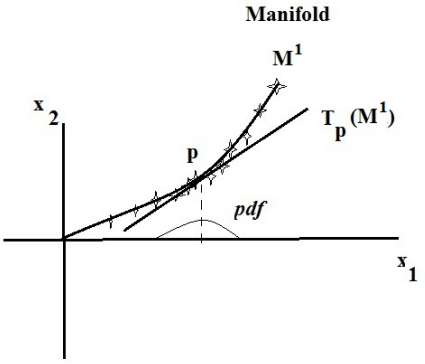


Figure 3. Data model elements: Original coordinate system (x_1, x_2), manifold ($\mathcal{M}^1$), tangent space at a point $\mathbf{p} \in \mathcal{M}^1$, probability density function (*pdf*).

Behind manifold learning techniques there is a data model which main elements are pictured on Figure 3. In this figure, the corresponding geometry is represented by a smooth curve, which is a differentiable manifold of dimension $d = 1$. Given a point $\mathbf{p} \in \mathcal{M}^1$, the tangent space $T_{\mathbf{p}}\left(\mathcal{M}^1\right)$ is the Euclidean space that spans all the vectors that are tangent to $\mathcal{M}^1$ in $\mathbf{p}$. Data points appear according to some probability density function (*pdf*), which is depicted bellow the tangent space in the Figure 3. In this paper, we are concerning with the problem of how to recover the underlying manifold from the samples.

Once $\mathcal{M}^d \subset \mathbb{R}^D$ the Whitney Theorem [30] assures that $D \geq 2d+1$. Therefore, if we compute an one-to-one smooth map $\psi : \mathcal{M}^d \to \mathbb{R}^s$ that preserves the differential structure of $\mathcal{M}^d$, called here embedding, such that $(2d+1) \leq s < D$ then we perform dimensionality reduction in the sense that the embedding f allows to represent each data point using less coordinates than the original data representation. This process is pictured in Figure 4.(a).

On the other hand, we could try to estimate the dimension d of the data manifold and, instead of computing the global embedding, we could calculate local parameterizations $\varphi_\alpha : U_\alpha \subset \mathbb{R}^d \to \mathcal{M}$ in order to recover the differentiable structure of the manifold, defined by properties (1)-(2) of section II, and represented in Figure 1.

Therefore, to address the manifold learning problem we can distinguish two approaches: (a) Learning methods that compute an embedding $\psi : \mathcal{M}^d \to \mathbb{R}^s$; (b) Seek for a compact data representation by recovering the differentiable structure behind the support manifold. Techniques like LLE and Isomap [31], [32] follow the first approach while LRML implements the second one [17].

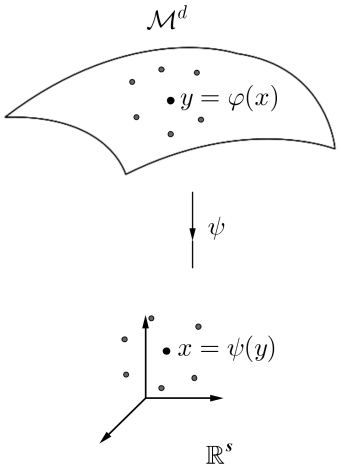


Figure 4. Data samples over the manifold $\mathcal{M}^d \subset \mathbb{R}^D$ and the embedding in the $\mathbb{R}^s$.

Besides, linear dimensionality methods, like PCA, can be used to estimate the manifold dimension and, consequently, compose the whole framework behind manifold learning. Moreover, nonlinear dimensionality reduction methods based on kernel approaches can be analysed using differentiable manifold elements [33]. Such relationships between dimensionality reduction methods motivates the taxonomy shown in Figure 5.

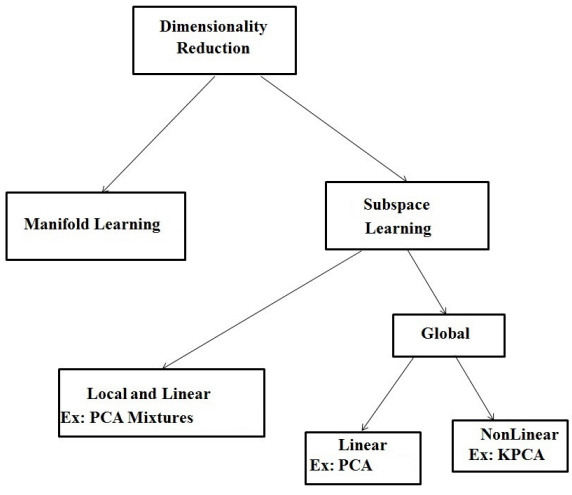


Figure 5. Taxonomy for dimensionality reduction methods: manifold learning, subspace learning and subclasses.

IV. Manifold Learning Issues

In most of manifold learning methods, there are two free parameters: the neighborhood size and the intrinsic dimension of the high dimensional data set. The former is fundamental to define the manifold topology. For instance, the Figure 6 shows an example that illustrates the problem when choosing a too large neighborhood size: point $\mathbf{x}_i$ will be connected with point $\mathbf{c}$ generating paths that do not belong to the data manifold (short-circuits). Such problem is even more important in sparse data sets, as pictured in Figure 6. The generation of the local parameterizations in definition of differentiable manifold (section II) undergoes also strong influence of the neighborhood size parameter. The intrinsic dimension can provide insights about the complexity of the model needed to represent the data, as well as the actual degrees of freedom involved which may be different from the dimensionality of the input space. Next, we provide some solutions for these issues.

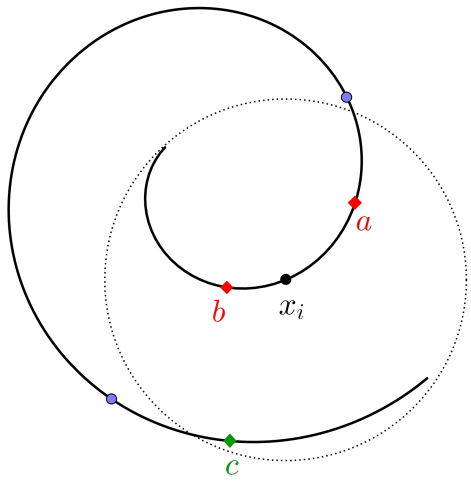


Figure 6. Sparse sampling and a short-circuit point in green.

A. Recovering the data topology

Among the possibilities to yield the data topology [34], [35], we will describe the solution presented in the RML technique [15]. The RML has as input the database $\mathcal{D} = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\} \subset \mathbb{R}^D$, the number of neighbors for each point $\mathbf{p}_i \in \mathcal{D}$ and a parameter ρ used for the selection of these neighbors, both chosen by trial and error. Initially, the Dijkstra's algorithm is used to compute the minimum paths on the complete graph $G(\mathcal{D})$ whose vertices are the elements of $\mathcal{D}$.

1 For each element $\mathbf{p}_i \in \mathcal{D}$ find the set $KNN(\mathbf{p}_i)$ composed by the $K \leq N$ nearest neighbors of $\mathbf{p}_i$ in the graph $G(\mathcal{D})$.

2 Determine the set of *visible* neighbors of $\mathbf{p}_i$ defined by:

$$VN(\mathbf{p}_i) = \{\mathbf{p} \in KNN(\mathbf{p}_i); \langle \mathbf{p}_i - \mathbf{p}, \mathbf{p}_i - \mathbf{q} \rangle \geq 0, \quad \forall \mathbf{q} \in KNN(\mathbf{p}_i)\}. \tag{7}$$

3 Obtain the *safe* neighborhood denoted by $SN(\mathbf{p}_i)$ as follows: Firstly, sort the vectors $\mathbf{p}_j - \mathbf{p}_i$, $\mathbf{p}_j \in VN(\mathbf{p}_i)$ in the ascending order of lengths to obtain the set $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_K\}$. Next, apply PCA to compute the local intrinsic dimension d_j of the subspace generated by the first j $(1 \leq j \leq K)$ elements of $\{\mathbf{e}_1, \ldots, \mathbf{e}_K\}$. If $d_j > d_{j-1}$, calculate the jump of increased length $\|\mathbf{e}_j\| - \|\mathbf{e}_{j-1}\|$. If the maximal jump is larger than a threshold given by $\rho\, \mu(\|\mathbf{e}_j\|)$, where $\rho > 0$ and $\mu(\|\mathbf{e}_j\|)$ is the average size of vectors $\{\mathbf{e}_1, \ldots, \mathbf{e}_K\}$, then $\mathbf{p}_j$ is removed.

B. Building Neighborhood System

In this section we build a suitable cover for the database that is composed of subsets $\mathcal{D}_i \subset \mathcal{D}$ such that $\mathcal{D} = \cup_{i=1}\mathcal{D}_i$. For this task we revise the solution presented in [17]. We consider the subgraph $\widetilde{G}$ of G, whose edges connect only safe neighbors determined in section IV-A, and use Dijkstra's algorithm to compute shortest paths on the graph $\widetilde{G}$ which, given two safe points $\mathbf{p}_1$ and $\mathbf{p}_2$, may be considered approximations of the geodesic distances between them, given by expression (4). We denote by $d_{\widetilde{G}}(\mathbf{p}_1, \mathbf{p}_2)$, the shortest distance between the two points on the graph $\widetilde{G}$.

1 Choose $\mathbf{p} \in \mathcal{D}$ and set $W \leftarrow \mathcal{D}$ and $i \leftarrow 1$.

2 Main Loop: while $W \neq \emptyset$:

3 $\mathbf{q}_i \leftarrow \mathbf{p}$,

4 Let $V(\mathbf{q}_i)$ the set of K nearest neighbors of $\mathbf{q}_i$ in $\widetilde{G}$.

5 Find the data point $\mathbf{q}'$ that is farthest from $\mathbf{q}_i$ in the set $V(\mathbf{q}_i)$ and compute $r_i = d_{\widetilde{G}}(\mathbf{q}_i, \mathbf{q}')$, the geodesic distance from $\mathbf{q}_i$ to q'. Take the safe points $\mathbf{s}$ such that $d(\mathbf{q}_i, \mathbf{s}) \leq \alpha r_i$ where $\alpha > 1.0$. Let $\mathcal{D}_i$ the obtained set.

6 Find $\mathbf{p}$ that is farthest from $\mathbf{p}_i$ in $\mathcal{D}_i$,

7 If $W \neq W - \mathcal{D}_i$ then $W \leftarrow W - \mathcal{D}_i$. Otherwise choose $\mathbf{p} \in W$,

8 $i \leftarrow i + 1$,

The output of the above algorithm is given by the points $\mathbf{q}_i \in \mathcal{D}$, called base points, and the sets $\mathcal{D}_i$, $V(\mathbf{q}_i)$, $i = 1, \cdots, L$, where L is the execution number of the main loop.

C. Manifold Dimension Estimation

The data manifold dimension is directly related to the concept of intrinsic dimension of a data set, which is the minimum number of coordinates that is necessary in order to account for all the information in the data [36]. In this paper, the data set is modeled by the manifold geometry which is supposed to be a connected topological space locally homeomorphic to Euclidean d-space. The number d is the

manifold dimension (section II) which can be also defined as intrinsic dimension of data set [36].

The use of more dimensions than strictly necessary leads to several problems in terms of space needed to store the data as well as performance of algorithms. In fact, the computational complexity of pattern recognition procedures depends on the dimension of the feature vectors, so a reduction of the dimension can result in reduced computation time. In this scenario, classification of different sample groups must be performed over the manifold by computing separating submanifolds [37]. Hence, the suitable determination of the manifold dimension leads to improved classification performance.

Following [36], we may distinguish two categories of techniques for estimating the manifold dimension. The first one is local, using the information contained in sample neighborhoods, avoiding the projection of the whole data set onto a lower-dimensional space. For instance, at the end of the steps described on section IV-A, we can get a dimension d_i for every $SN(p_i)$ by computing the PCA (section V-A next) and keep the d_i eigenvectors with eigenvalues larger than a predefined tolerance. Then, and we can estimate the manifold dimension through the average of the d_i's. Regarding the data model pictured in Figure 3, in this process we are determining the dimension of the tangent space $T_{\mathbf{p}}\left(\mathcal{M}^d\right)$. However, the sets $SN(p_i)$ define a topology of the data set, which allows to define a neighborhood system for the target manifold, as we shall see next.

In the second class (global), the data set is unfolded in the d-dimensional space [38]. Traditional methods in this category include PCA, that works well if the manifold curvature is small. Global method based on minimal spanning trees of geodesic graphs are also reported in the literature [36].

V. Linear Dimensionality Reduction

These methods can be also classified as subspace learning approaches in the sense that the output linear space has an optimum subspace for compact data representation. Among the techniques in this category [4], [5], we describe bellow the PCA and MDS because the former is commonly used to estimate the manifold dimension (as commented in section IV-C) while the latter is applied in the ISOMAP to compute a distance preserving embedding for dimensionality reduction.

A. Principal Component Analysis (PCA)

Given the database in expression (6) the PCA works as follows:

1) Compute the global mean:

$$\overline{\mathbf{p}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{p}_i, \tag{8}$$

2) Compute the set $\{(\mathbf{p}_1 - \overline{\mathbf{p}}), (\mathbf{p}_2 - \overline{\mathbf{p}}) \cdots, (\mathbf{p}_N - \overline{\mathbf{p}})\}$, and the Covariance Matrix:

$$S = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{p}_i - \overline{\mathbf{p}}) \cdot (\mathbf{p}_i - \overline{\mathbf{p}})^T, \tag{9}$$

3) Compute the eigenvectors $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_D\}$ of S and select the d ones which correspond to the largest eigenvalues as the principal direction.

Considering Figure 3, we shall observe that, the step (3) will select d directions, say $\mathbf{v}_1, \cdots, \mathbf{v}_d$, such that $span\{\mathbf{v}_1, \cdots, \mathbf{v}_d\} \approx T_{\overline{\mathbf{p}}}\left(\mathcal{M}^d\right)$. Therefore, if the manifold curvature is small everywhere, we can represent $\mathcal{M}^d$ as a linear d dimensional space:

$$\mathcal{M}^d = \left\{\overline{\mathbf{p}} + \mathbf{w} \in \mathbb{R}^D;\ \mathbf{w} \in span\{\mathbf{v}_1, \cdots, \mathbf{v}_d\}\right\}. \tag{10}$$

B. Multidimensional Scaling (MDS)

Let the distance matrix $A = \{d_{ij}\}$, where:

$$d_{ij} = d(\mathbf{p}_i, \mathbf{p}_j) = ||\mathbf{p}_i - \mathbf{p}_j||, \tag{11}$$

with $||\cdot|| \equiv ||\cdot||_2$ along this paper, and $\mathbf{p}_i, \mathbf{p}_j \in \mathcal{D}$ defined in expression (6). Therefore:

$$d_{ij}^2 = ||\mathbf{p}_i||^2 + ||\mathbf{p}_j||^2 - 2\mathbf{p}_i^T\mathbf{p}_j. \tag{12}$$

Let the matrices $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$ given by:

$$a_{ij} = -\frac{1}{2}d_{ij}^2, \tag{13}$$

$$b_{ij} = \mathbf{p}_i^T\mathbf{p}_j = -\frac{1}{2}\left(d_{ij}^2 - ||\mathbf{p}_i||^2 - ||\mathbf{p}_j||^2\right), \tag{14}$$

The multidimensional scaling seeks for a low-dimensional orthogonal projection $\mathbf{y}_i = P^T\mathbf{p}_i$, where $P \in \mathbb{R}^{D\times s}$ that solves the problem [11]:

$$\min_P \sum_i \sum_j \left(d(\mathbf{p}_i, \mathbf{p}_j) - d\left(P^T\mathbf{p}_i, P^T\mathbf{p}_j\right)\right)^2. \tag{15}$$

So, following [11], [39], we form the matrix:

$$B = HAH, \tag{16}$$

where $A = \{(-1/2)d_{ij}^2\}$, $H = I_N - N^{-1}J_N$ is a centering matrix, J_N is the matrix of all ones and I_N is the identity matrix, all of them with dimension $N \times N$. Then, we find a matrix B^* that minimizes:

$$trace\left\{(B - B^*)^2\right\} = \sum_i \sum_j \left(b_{ij} - b_{ij}^*\right)^2. \tag{17}$$

If $\{\lambda_k\}$ and $\{\lambda_k^*\}$ are the eigenvalues of B and B^*, respectively, then the minimum of the objective function (17) is given by $\sum_{k=1}^{N}(\lambda_k - \lambda_k^*)^2$, where $\lambda_k^* = \max(\lambda_k, 0)$, for $k = 1, 2, \ldots, N$. By the spectral theorem we know that we can factor B as:

$$B = V \Lambda V^T, \tag{18}$$

where $\Lambda = diag\{\lambda_1, \lambda_2, \ldots, \lambda_N\}$ is the diagonal matrix of the eigenvalues of B, and $V = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N)$ is the matrix whose columns are the eigenvectors of B. If B is nonnegative-definite with $s < D$ non-null eigenvalues, then expression (18) becomes:

$$B = V_s \Lambda_s V_s^T = \left(V_s \Lambda_s^{1/2}\right)\left(\Lambda_s^{1/2} V_s\right)^T = YY^T, \tag{19}$$

where $V_s = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_s)$, $\Lambda_s = diag\{\lambda_1, \lambda_2, \ldots, \lambda_s\}$, and:

$$Y = V_s \Lambda_s^{1/2} = \left(\sqrt{\lambda_1}\mathbf{v}_1, \sqrt{\lambda_2}\mathbf{v}_2, \ldots, \sqrt{\lambda_s}\mathbf{v}_s\right). \tag{20}$$

The new coordinates are the columns $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N \in \mathbb{R}^s$ of $Y^T \in \mathbb{R}^{s \times N}$ and we can show that:

$$||\mathbf{y}_i - \mathbf{y}_j||^2 = (\mathbf{y}_i - \mathbf{y}_j)^T (\mathbf{y}_i - \mathbf{y}_j) = d_{ij}^2,$$

where d_{ij} is given by expression (11).

VI. Manifold Learning Methods

If the curvature cannot be discarded, as observed in Figure 3, than the expression (10) gives a poor representation of $\mathcal{M}^d$ if globally applied. Manifold learning methods address such limitation of linear techniques by embedding linear subspace learning as local structures in the data geometry.

A. Local Tangent Space Alignment (LTSA)

We assume that the target differentiable manifold $\mathcal{M}^d$ is globally parameterized through an unknown function:

$$f : U \subset \mathbb{R}^d \to \mathcal{M}^d, \tag{21}$$

with $d < D$ and $\mathcal{M}^d \subset \mathbb{R}^D$.

Given the subset $\mathcal{D} \subset \mathcal{M}$, where $\mathcal{D}$ is defined in expression (6), we want to estimate $\tau_i \in \mathbb{R}^d$, such that:

$$f(\tau_i) = \mathbf{p}_i, i = 1, 2, \cdots, N, \tag{22}$$

without explicitly constructing f.

Let:

$$f(\overline{\tau}) = f(\tau) + J(\tau)(\overline{\tau} - \tau) + O\left(\|\overline{\tau} - \tau\|^2\right) \tag{23}$$

If we know an unitary matrix $Q_{\overline{\tau}}$ forming an orthonormal basis for the tangent space $T_{\overline{\tau}}$ we can write:

$$J(\tau)(\overline{\tau} - \tau) = Q_{\overline{\tau}} \theta_{\overline{\tau}}^* \tag{24}$$

Therefore:

$$\theta_{\overline{\tau}}^* = Q_{\overline{\tau}}^T J(\tau)(\overline{\tau} - \tau) \equiv P_{\overline{\tau}}(\overline{\tau} - \tau)$$

But:

$$J(\tau)(\overline{\tau} - \tau) = [f(\overline{\tau}) - f(\tau)] + O\left(\|\overline{\tau} - \tau\|^2\right)$$

So, we can think about an approximation for $\theta_{\overline{\tau}}^*$ given by:

$$\theta_{\overline{\tau}} = Q_{\overline{\tau}}^T [f(\overline{\tau}) - f(\tau)] = \theta_{\overline{\tau}}^* + O\left(\|\overline{\tau} - \tau\|^2\right).$$

So, we are looking for $\overline{\tau}$ and $P_{\overline{\tau}}$ such that:

$$\int d\tau \int_{\Omega(\tau)} \|P_{\overline{\tau}}(\overline{\tau} - \tau) - \theta_{\overline{\tau}}\| \, d\overline{\tau} \approx 0 \tag{25}$$

If $J(\tau)$ is of full column rank, the matrix $P_{\overline{\tau}}$ is non-singular and then:

$$\overline{\tau} - \tau \approx P_{\overline{\tau}}^{-1} \theta_{\overline{\tau}} \equiv L_{\overline{\tau}} \theta_{\overline{\tau}}$$

So, we can introduce this result in expression (25) and say that we are looking for $\overline{\tau}$ and an affine transformation $L_{\overline{\tau}}$ such that:

$$\int d\tau \int_{\Omega(\tau)} \|\overline{\tau} - \tau - L_{\overline{\tau}} \theta_{\overline{\tau}}\| \, d\overline{\tau} \approx 0 \tag{26}$$

The main point here is how to translate the above (continuous) development in a model driven by the data. In the local tangent space alignment (LTSA) method the solution is based on the following steps [40]: (a) for each $\mathbf{p}_i$ of the database take the K-nearest neighbors $KNN(\mathbf{p}_i)$ and form the matrix $X_i = [\mathbf{p}_{i_1}, \mathbf{p}_{i_2}, \cdots, \mathbf{p}_{i_K}]$; (b) For each local neighborhood $KNN(\mathbf{p}_i)$ compute a d-dimensional affine approximation by solving the optimization problem:

$$min_{\mathbf{p}, \theta_j, Q_i} \sum_{j=1}^{k} \left\|\mathbf{p}_{i_j} - \left(\mathbf{p} + Q_i \theta_j^i\right)\right\|_2^2 \tag{27}$$

The solution of this problem is $\mathbf{p} = \overline{p}_i + Q_i \theta_j^i$, where $\overline{\mathbf{p}}_i$ is the mean of the elements of $KNN(\mathbf{p}_i)$ and Q_i is obtained using the d largest singular values of the matrix $X_i - \overline{\mathbf{p}}_i \mathbf{e}^T$, where $\mathbf{e}$ is a m-dimensional column vector of all ones. The matrix Q_i corresponds to the matrix $Q_{\overline{\tau}}$ in the continuous formulation (expression (24)) and will be the input for the next stage of the LTSA method in order to solve a discrete version of the problem (26) for computing the global coordinates $\tau_i, i = 1, 2, \cdots, N$ [40]. This is performed by written the reconstruction error

$$\tau_{i_j} = \overline{\tau}_i + L_i \theta_j^i + \varepsilon_j^i, \tag{28}$$

with $j = 1, 2 \ldots, K$ and $i = 1, 2, \ldots, N$, where L_i is a local affine transformation. If we form the matrices $T_i = [\tau_{i_1}, \ldots, \tau_{i_K}]$ and $E_i = \left[\varepsilon_1^i, \ldots, \varepsilon_K^i\right]$, and $\Theta_i = \left[\theta_1^i, \ldots, \theta_K^i\right]$ then we can assemble the reconstruction errors in the matrix form:

$$E_i = T_i \left(I - \frac{1}{K} \mathbf{e}\mathbf{e}^T\right) - L_i \Theta_i, \tag{29}$$

where $\mathbf{e}$ is an N-dimensional column vector of all ones. The optimal alignment matrix L_i that minimizes the Frobenius norm [41] of the local reconstruction $||E_i||_F$ is [40]:

$$L_i = T_i \left(I - \frac{1}{K} \mathbf{e}\mathbf{e}^T \right) \Theta_i^+, \tag{30}$$

where Θ_i^+ denotes the Moor-Penrose generalized inverse of Θ_i^+ [41]. The last step of LTSA algorithm is to find T_i that minimizes the global reconstruction error:

$$\sum_{i=1}^{N} ||E_i||_F.$$

B. Locally Linear Embedding (LLE)

The LLE method [31] is a manifold learning technique that works through scheme of Figure 7.

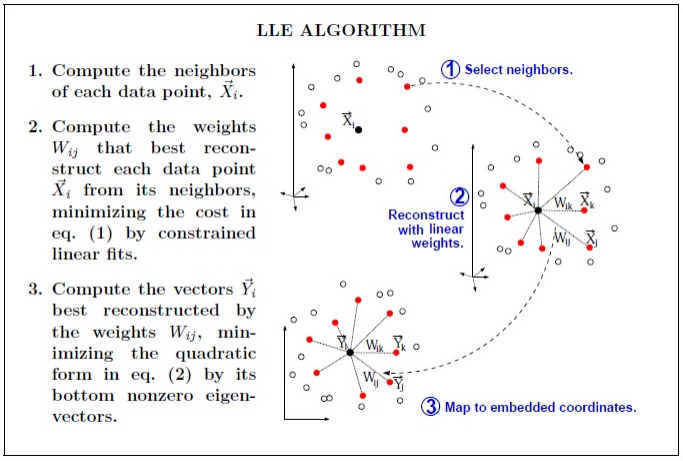


Figure 7. Basic steps of Locally Linear Embedding (LLE) algorithm (Reprinted from [31]).

In this method, for each data point $\mathbf{p}_i \in \mathbb{R}^D$ it is computed the KNN neighbors, represented with red color in the Figure 7. Then, a weighting vector $w_{i,j}$ is computed for each $\mathbf{p}_i$ by solving the optimization problem:

$$\min_{(w_{i1}, w_{i2}, \cdots, w_{iK})} \left\| \mathbf{p}_i - \sum_{j=1}^{K(i)} w_{ij} \mathbf{p}_j \right\|^2, \tag{31}$$
$$subject \quad to: \sum_{j=1}^{K(i)} w_{ij} = 1.$$

where $K(i) = KNN(\mathbf{p}_i)$.

Expression (31) implies that we are looking for a point, closer to the point $\mathbf{p}_i$, in the convex hull of the polygon with vertices in the $KNN(\mathbf{p}_i)$ set. Finally, if we assume that the underlying data geometry can be embedded in $\mathbb{R}^s$, with $s < D$, then we can assign a feature vector $\mathbf{y}_i \in \mathbb{R}^s$ to each $\mathbf{p}_i$ by solving the expression:

$$\min_{\mathbf{y}_i} \left\| \mathbf{y}_i - \sum_{j=1}^{K(i)} w_{ij} \mathbf{y}_j \right\|^2, \tag{32}$$

Observe that in this case we are not computing a compact representation to the tangent space. Instead, the low dimensional output $\mathbf{y}_i$ represents the s coordinates on the manifold respect to the target space $\mathbb{R}^s$, as represented in Figure 4.

C. Isometric Feature Mapping

This approach builds on classical MDS but seeks an isometric embedding; that is, a map that preserves the geodesic manifold distances between all pairs of data points. In this way, the isometric feature mapping (ISOMAP) seeks an embedding of the data points $\mathcal{D}$ in $\mathbb{R}^s$ that preserves the intrinsic manifold geometry, estimated through the geodesic distances. So, the ISOMAP algorithm is composed by three steps, as follows.

(1) For each $\mathbf{p}_i \in \mathcal{D}$ compute the K nearest neighbors in $\mathcal{D}$, denoted by $KNN(\mathbf{p}_i)$;

(2) Geodesic Distances Computation: Build a graph $\mathcal{G} = \mathcal{G}(\mathcal{D}, \mathcal{E})$, where the vertices are the points of $\mathcal{D}$ and matrix $\mathcal{E} = \{e_{ij}\}$ indicates neighborhood relationships between points in the sense that, $e_{ij} = 1$ if $\mathbf{p}_j \in KNN(\mathbf{p}_i)$, and $e_{ij} = 0$, otherwise. Next, we estimated the geodesic distances by computing paths of minimum lengths on the graph $\mathcal{G}$ using the Dijkstra algorithm [42]. The result is the matrix $D_{\mathcal{G}} = \{d_{\mathcal{G}}^2(i,j)\}$, where $d_{\mathcal{G}}(i,j)$ is the length of the shortest path, in the graph $\mathcal{G}$, between the points $\mathbf{p}_i, \mathbf{p}_j \in \mathcal{D}$.

(3) Embedding Construction through MDS: We apply the MDS Algorithm (section V-B) on the graph $\mathcal{G}$ by replacing the matrix A by $A_{\mathcal{G}} = (-1/2)D_{\mathcal{G}}$ in expression (17) to obtain a representation of the data in the reduced dimension s, as shown in Figure 4, preserving the geodesic distances.

D. Local Riemannian Manifold Learning (LRML)

In the LRML method, it is used the RML algorithm [15] to determine the topology of the data (selection of neighborhoods), perform the dimensionality reduction through a local adaptation of the fast RML algorithm [16], and synthesize new points in the original data space using barycentric coordinates.

1) Local Normal Coordinates : The sets $\mathcal{D}_i$, obtained in section IV-B, provide a cover for $\mathcal{D}$ that together with the manifold dimension d and the sets $V(\mathbf{p}_i)$ (section IV-B), are used as input to the local version of the fast RML, given below:

1 Take the set $V(\mathbf{p}_i)$ and apply PCA to calculate an orthonormal basis $\{\mathbf{e}_j^i\}_{j=1}^d$ for the tangent space. The points $\mathbf{p} \in V(\mathbf{p}_i)$ are projected onto the $T_{\mathbf{p}_i}\mathcal{M}^d$, and represented as $\mathbf{y} = (y_1, \cdots, y_d)^T \in \mathbb{R}^d$ in the PCA coordinate system. Proceed similarly to $\mathbf{p} \in \mathcal{D}_i - V(\mathbf{p}_i)$ to obtain $\mathbf{x} = (x_1, \cdots, x_d)^T \in \mathbb{R}^d$. Let Y_i e X_i the obtained coordinate sets.

2 Consider the square of the geodesic distance function $\mathbf{d}^2_{\mathcal{M}^d}(\mathbf{x},\mathbf{y})$. Compute its bilinear last square approximation $f_i : X_i \times Y_i \to \mathbb{R}$, where $f_i(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T A_i(\mathbf{x}-\mathbf{y})$, with $A_i \in \mathbb{R}^{d\times d}$ a symmetric and positive defined matrix.

3 Given $\mathbf{p} \in \mathcal{D}_i$, estimate the direction of the geodesic passing through $\mathbf{p}$ and $\mathbf{p}_i$ by computing $\mathbf{g} = \sum_{j=1}^{d} g^j \mathbf{e}^i_j = \nabla_{\mathbf{y}} f_i(\mathbf{x},\mathbf{y})\big|_{\mathbf{y}=\mathbf{p}_i}$.

4 Calculate the normal Riemann coordinates $\mathbf{z}$ for $\mathbf{p} \in \mathcal{D}_i$ through: $\mathbf{z} = d_{\mathcal{M}^d}(\mathbf{p},\mathbf{p}_i)\dfrac{\mathbf{g}}{\sqrt{\langle \mathbf{g},\mathbf{g}\rangle}}$, where $d_{\mathcal{M}^d}(\mathbf{p},\mathbf{p}_i)$ is approximated through the minimum path in the graph $\widetilde{G}$ (section IV-B) linking the points $\mathbf{p}$ and $\mathbf{p}_i$.

As a consequence of the step 4, we get $\mathbf{z}_i = \mathbf{0}$ as the normal coordinate vector for the point $\mathbf{p}_i$. At the end of this process we obtain the set $\widetilde{\mathcal{D}} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N\}$ where $\mathbf{z}_j$ is the (local) normal coordinates of $\mathbf{p}_j \in \mathcal{D}$. Besides, we get a lower dimensional data representation that preserves geodesic distances because $\langle \mathbf{z}_i, \mathbf{z}_j\rangle = (d_{\mathcal{M}^d}(\mathbf{p}_i,\mathbf{p}_j))^2$.

This is an important property because the distance between points (images) is directly linked to notion of similarity between images. Therefore, it is important to preserve geodesic distances in the final representation. The Figure 8 summarizes the main elements of the LRML and the obtained manifold parametrization by exponential map.

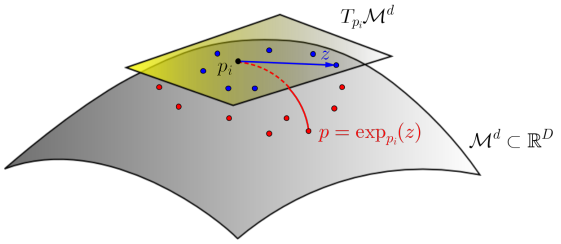


Figure 8. Exponential map from $T_{\mathbf{p}_i}(\mathcal{M}^d)$ to $\mathcal{M}^d \subset \mathbb{R}^D$, with $\langle \mathbf{z},\mathbf{z}_i\rangle = (d_{\mathcal{M}^d}(\mathbf{p},\mathbf{p}_i))^2$, para $\mathbf{z} \in T_{\mathbf{p}_i}(\mathcal{M})$ and $\exp_{\mathbf{p}_i}(\mathbf{z}) = \mathbf{p} \in \mathcal{M}^d$.

2) Composition of Local Parameterizations : Let us take a look to the process for Riemann local normal coordinates computation, given by Definitions 1 and 2 of section II to explain a key limitation of LRML for pattern recognition applications. The LRML generates a decomposition of the database into a neighborhood system $\{\mathcal{D}_i, \quad i = 1,2,\ldots,L\}$ and builds a local parametrization for each neighborhood $\varphi_i : U_i \subset \mathbb{R}^d \to \mathcal{M}$, where φ_i is given by the local exponential map. So, a point $\mathbf{z} = (u_1,u_2,\ldots,u_d)^T \in U_i$ gives the Riemann normal coordinates of its image $\mathbf{p} = \varphi_i(\mathbf{z})$. In fact, by returning to Figures 2, $U_i \subset T_{\mathbf{p}}(\mathcal{M}^d)$ where $\mathcal{M}^d$ is the manifold holding the data and $\mathbf{p}$ is common point between the manifold and the tangent plane used for the normal coordinates computation. Therefore, $\varphi_i(0,0,\ldots,0) = \mathbf{p}$ because the geodesic distance $d_{\mathcal{M}^d}(\mathbf{p},\mathbf{p}) = 0$. So, $(0,0,\ldots,0) \in U_i$, $i = 1,2,\cdots,L$, which implies that $\cap_{i=1}^{L} U_i = \{\mathbf{0}\}$, a fact already pointed out in [17]. So, the obtained parametrization can not be used as a feature space, for instance, without ambiguities.

To address this limitation, in this section, we adapt the technique proposed in [43] to compute rigid-body transformations to properly position and orient the local coordinate systems to get a final global parametrization. Firstly, we take the sets $\mathcal{D}_i$ and build disjoint clusters X_i as follows: $X_1 = \mathcal{D}_1$ and $X_{i+1} = \mathcal{D}_{i+1} - \cup_{j=1}^{i} X_j$, $i = 1,2,3,\ldots,L-1$. The Figure 9 pictures the result of this iterative process when considering the sets $\mathcal{D}_1$, $\mathcal{D}_2$ and $\mathcal{D}_3$, colored with red, green and blue, respectively. At the initial step of the iterative process we get $X_1 = \mathcal{D}_1$, shown in red. Then, the next iteration will generate the set X_2 through the difference $\mathcal{D}_{1+1} - \cup_{j=1}^{1} X_j = \mathcal{D}_2 - X_1$, which gives the green set on the right side of the Figure 9. Analogously, the procedure will generate the set $X_3 = \mathcal{D}_3 - (X_1 \cup X_2)$, in blue color. We can verify that $X_1 \cap X_2 \cap X_3 = \emptyset$, which is the desired property.

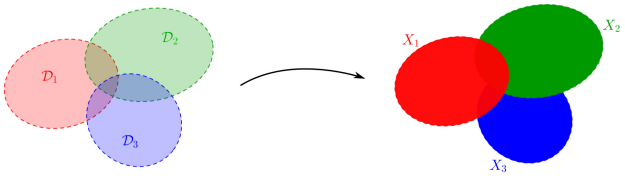


Figure 9. Example for construction of the disjoint sets X_1, X_2 and X_3.

Next, we estimate the base points set $BP = \{\mathbf{q}_i \in X_i, \quad i = 1,2,\ldots,L\}$, which is computed by an approximated version of the traditional minimax algorithm to calculate the center of each X_i. Then, we find the nearest data pair $\mathbf{nx}^i_j \in X_i$ and $\mathbf{nx}^j_i \in X_j$ (see Figure 10).

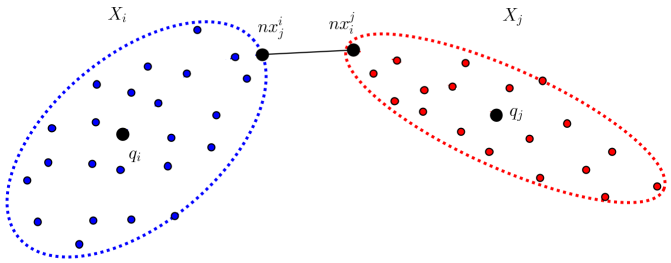


Figure 10. Elements of the composition method.

Each set X_i has a landmark set $NX_i = \{\mathbf{nx}^i_{i_1}, \mathbf{nx}^i_{i_2}, \ldots, \mathbf{nx}^i_{i_L}\}$ with local reduced representations $Y_i = \{\mathbf{y}^i_1, \mathbf{y}^i_2, \cdots, \mathbf{y}^i_{l_i}\}$ and $NY_i = \{\mathbf{ny}^i_{i_1}, \mathbf{ny}^i_{i_2}, \ldots, \mathbf{ny}^i_{i_L}\}$, respectively, both computed by LRML. The base points set BP has also a local reduced LRML representation denoted by

$\overline{BP} = \{\overline{\mathbf{q}}_1, \overline{\mathbf{q}}_2, \cdots, \overline{\mathbf{q}}_L\}$. Moreover, we can compute the (global) low dimensional representation of NX_i as well as of the base points in the set BP through any dimensionality reduction method that preserves geodesic distances. The obtained sets, named $SY_i = \{\mathbf{sy}^i_{i_1}, \mathbf{sy}^i_{i_2}, \ldots, \mathbf{sy}^i_{i_L}\}$ and $CY = \{\mathbf{cy}_1, \mathbf{cy}_2 \ldots, \mathbf{cy}_L\}$ are used as the *skeleton* of the representational space in order to compute the rotation matrix A^i to properly reorient the local parametrization Y_i. However, instead of computing principal axes like performed in [43], we achieve a more robust implementation by solving the following least square problem:

$$A^i = \arg\min_A \sum_{k=1}^{L} \left\| A\left(\mathbf{ny}^i_{i_k} - \overline{\mathbf{q}}_i\right) - \left(\mathbf{sy}^i_{i_k} - \mathbf{cy}_i\right)\right\|^2, \tag{33}$$

where $i = 1, 2, \ldots, L,$.

Therefore, we can align the local representation Y_i with the global skeleton $SY_i \cup CY$ by computing the set:

$$RY_i = \left\{\mathbf{ry}^i_j = A^i \times \mathbf{y}^i_j, \quad j = 1, \ldots, l_i\right\}. \tag{34}$$

We must translate each RY^i to its proper location given by lower dimensional representation of the corresponding base point:

$$TY_i = \left\{\mathbf{ty}^i_j = \mathbf{ry}^i_j + \mathbf{cy}_i, \quad j = 1, \ldots, l_i\right\}. \tag{35}$$

Finally, the composition:

$$TY = \cup_{i=1}^{L} TY_i, \tag{36}$$

gives global low dimensional representation of the database in the representational space.

VII. Discrete Methods for Syntheses and Data Analysis

After performing the dimensionality reduction, we need a technique to compute the synthesis of new data from new points in the representational space. Specifically, given a new point $\tilde{\mathbf{ty}} \in \mathbb{R}^d$, $\tilde{\mathbf{ty}} \notin TY$, how to compute the corresponding image $\mathbf{q} \in \mathcal{M}^d$ that is immersed in $\mathbb{R}^D$? In [21], this problem is addressed using barycentric coordinates. In this section we immerse this idea in the framework of discrete and polyhedral geometry in order to efficiently use barycentric coordinates and to incorporate piecewise linear methods for data analysis in the reduced space. Let's look at some fundamental concepts initially [44].

Definition 3. Given a set $\mathcal{D} = \{\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_m\} \subset \mathbb{R}^{N+1}$ we define the convex hull $(co\,(\mathcal{D}))$, the affine hull $(aff\,(\mathcal{D}))$ and the tangent space $(tng\,(\mathcal{D}))$, respectively, as follows:

$co\,(\mathcal{D}) = \{\mathbf{v} = \sum_{i=0}^{m} \alpha_i \mathbf{p}_i \mid \alpha_i \geq 0, \sum_{i=1}^{m} \alpha_i = 1\}$,

$aff\,(\mathcal{D}) = \{\mathbf{v} = \sum_{i=0}^{m} \alpha_i \mathbf{p}_i \mid \sum_{i=1}^{m} \alpha_i = 1\}$,

$tng\,(\mathcal{D}) = \{\mathbf{v} = \mathbf{w}_1 - \mathbf{w}_2 \mid, \mathbf{w}_1, \mathbf{w}_2 \in aff\,(\mathcal{D})\}$.

If the vectors $\mathbf{p}_1 - \mathbf{p}_0, \mathbf{p}_2 - \mathbf{p}_0, ..., \mathbf{p}_m - \mathbf{p}_0$ are linearly independent in $\mathbb{R}^{N+1}$ then points in $\mathcal{D}$ are called affinely independent. In this case, the set $co\,(\mathcal{D})$ is called a m-simplex.

If $\mathbf{u} \in \mathbb{R}^{N+1}$, $\mathbf{u} \neq 0$ and $\gamma \in \mathbb{R}$, then the set $\left\{\mathbf{x} \in \mathbb{R}^{N+1} \mid \mathbf{u}^T\mathbf{x} \geq \gamma\right\}$ is named half-space and its boundary $\left\{\mathbf{x} \in E \mid \mathbf{u}^T\mathbf{x} = \gamma\right\}$ a hyperplane. A finite intersection of half-spaces is a convex polyhedral or a cell [44]. Hence, cells are closed convex sets that may be bounded or unbounded. A bounded cell is called polytope. It has a finite number of vertices, say $\mathbf{p}_0$, $\mathbf{p}_1$,...,$\mathbf{p}_m$, and will be denoted by $[\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_m]$. The dimension of a cell σ is identified with the dimension of its tangent space. If that dimension is m, we say that the cell is m-dimensional or simply that we have a m-cell.

Definition 4. A $face$ τ of a cell σ is a convex subset $\tau \subset \sigma$ such that for all $\mathbf{x}, \mathbf{y} \in \sigma$ and $\lambda \in \mathbb{R}$ the following property holds: $\mathbf{x}, \mathbf{y} \in \sigma$, $0 < \lambda < 1$, $(1-\lambda)\,\mathbf{x} + \lambda\mathbf{y} \in \tau \Longrightarrow \mathbf{x}, \mathbf{y} \in \tau$.

A face τ of dimension k is called a k-face. For instance, a $0-$face is a vertex and a $1-$face is an edge of σ. If $\dim(\tau) = \dim(\sigma) - 1$, then τ is called a facet of σ. All other faces of σ are called proper faces. A point $\mathbf{v} \in \sigma - \{\tau \mid \tau$ is a k-face of $\sigma\}$ is an interior point of σ.

Definition 5. Let $\widehat{M}$ a family of $(N+1) - cells$. For $0 \leq k \leq N+1$, we define the sets:

$$\widehat{M}^{\,k} = \left\{\tau \mid \tau \text{ is a k-face of some cell} \quad \sigma \in \widehat{M}\right\}$$

and:

$$\left|\widehat{M}\right| = \bigcup_{\sigma \in \widehat{M}} \sigma.$$

Definition 6. We call $\widehat{M}$ in definition 5 a piecewise linear manifold of dimension $N+1$ if and only if the following conditions hold:

(1) The intersection $\sigma_1 \cap \sigma_2$ of two cells $\sigma_1, \sigma_2 \in \widehat{M}$ is empty or a common face of both cells;

(2) A facet $\tau \in \widehat{M}^{\,N}$ is common to at most two cells of $\widehat{M}$;

(3) The family $\widehat{M}$ is locally finite, that is, any compact subset of $\left|\widehat{M}\right|$ meets only finitely many cells of $\widehat{M}$.

A triangulation Γ is a special kind of piecewise linear manifold in which the cells are simplices of $\mathbb{R}^{N+1}$. A fundamental point for our work is to build a triangulation from a set $\mathcal{D}$ of points in $\mathbb{R}^{N+1}$ such that the $0-$faces of the triangles are points in $\mathcal{D}$. An efficient algorithm can be designed to perform this task by using the pivoting process.

Definition 7: Let $\sigma = [\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_N, \mathbf{p}_{N+1}]$ be a $(N+1)$-simplex with vertices in $\mathcal{D}$ and $\tau = [\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_{i-1}, \mathbf{p}_{i+1}, ..., \mathbf{p}_{N+1}]$ the facet of σ lying opposite the vertex $\mathbf{p}_i$. The process of pivoting consists of finding a point $\widetilde{\mathbf{p}}_i \in \mathcal{D}$, with $\widetilde{\mathbf{p}}_i \neq \mathbf{p}_i$, in order to build a simplex $\widetilde{\sigma} = [\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_{i-1}, \widetilde{\mathbf{p}}_i, \mathbf{p}_{i+1}, ..., \mathbf{p}_{N+1}]$ such that $\sigma \cap \widetilde{\sigma} = \tau$.

In the above process, we must be careful because it may exist a point $\mathbf{p} \in \mathcal{D} - \{\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_{i-1}, \widetilde{\mathbf{p}}_i, \mathbf{p}_{i+1}, ..., \mathbf{p}_{N+2}\}$ such that $\mathbf{p} \in \widetilde{\sigma}$. Besides, for each new simplex we must have the properties (1) and (2) above satisfied. The following algorithm summarizes the whole process:

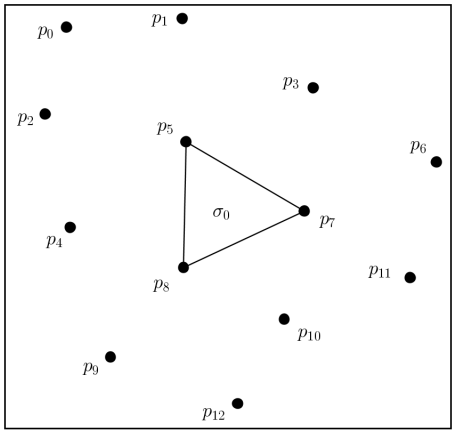
Algorithm 1: Building a triangulation from a finite set of points

- **0** Input: Set $\mathcal{D} = \{\mathbf{p}_0, ..., \mathbf{p}_m\} \subset \mathbb{R}^{N+1}$, with $m \geq N+1$
- **1** Output: Triangulation Γ with vertices in $\mathcal{D}$.
- **2** Initialization
  - **2.1**. Find a simplex $\sigma \subset \mathbb{R}^{N+1}$ with vertices in $\mathcal{D}$;
  - **2.2**. Actual list of simplices: $\sum = \{\sigma\}$;
  - **2.3**. $V(\sigma) =$ set of vertices of σ;
  - **2.4**. Point to simplex hash table: $H(\mathbf{p}) = \{\sigma\}$ if $\mathbf{p} \in V(\sigma)$; $H(\mathbf{p}) = \emptyset$ otherwise.
- **3** While $V(\sigma) \neq \emptyset$ for some $\sigma \in \sum$,
  - **3.1**. get $\sigma = \left[\mathbf{p}_{i_0}, ..., \mathbf{p}_{i_{N+1}}\right] \in \sum$ such that $V(\sigma) \neq \emptyset$;
  - **3.2**. get $\mathbf{p} \in V(\sigma)$;
  - **3.3**. $(\widetilde{\mathbf{p}}, \widetilde{\sigma}) = Pivoting(\mathbf{p}, \sigma)$;
  - **3.4**. drop $\mathbf{p}$ from $V(\sigma)$;
  - **3.5**. if $\widetilde{\sigma} \in \sum$ then
    - **3.5.1**. go to **3.**
  - **3.6**. else
    - **(a)**. $\sum \leftarrow \sum \cup \{\widetilde{\sigma}\}$;
    - **(b)**. Update hash table: $H(\mathbf{p}) \leftarrow H(\mathbf{p}) \cup \{\widetilde{\sigma}\}$ if $\mathbf{p} \in \widetilde{\sigma}$
    - **(c)**. $V(\widetilde{\sigma}) \leftarrow V(\widetilde{\sigma}) - \{\widetilde{\mathbf{p}}\}$;
  - **3.7**. Go to **3.**

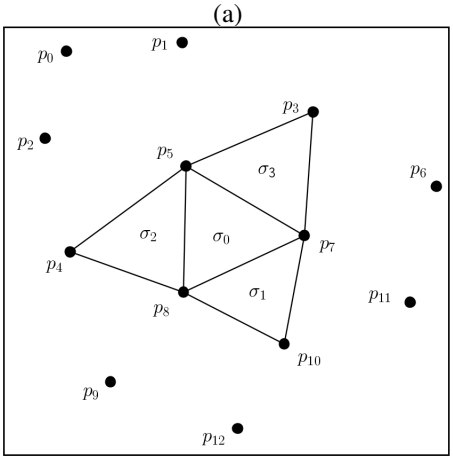
Procedure: Obtain $\widetilde{\sigma}$ from σ by pivoting the vertex $\mathbf{p}$ into $\widetilde{\mathbf{p}}$:

- $(\widetilde{\mathbf{p}}, \widetilde{\sigma}) = Pivoting(\mathbf{p}, \sigma)$
- Take the facet with vertex set $F = \left\{\mathbf{p}_{i_0}, ..., \mathbf{p}_{i_{N+1}}\right\} - \{\mathbf{p}\}$;
- Find the set of simplices $\triangle = \{\rho;\ \rho \in H(\mathbf{v})\ for\ \mathbf{v} \in F\}$;
- Consider the hyperplane defined by $\mathbf{u} \in \mathbb{R}^{N+1}$ and $\lambda \in \mathbb{R}$, such that $\mathbf{u}^T \cdot \mathbf{x} - \lambda = 0, \forall \mathbf{x} \in F$ and $\mathbf{u}^T \cdot \mathbf{p} - \lambda < 0$,
- Delete from $\triangle$ any simplex ρ that satisfies $\mathbf{v} \in \rho \Longrightarrow \mathbf{u}^T \cdot \mathbf{v} - \lambda \leq 0$;
- If $\triangle = \emptyset$ then
  - find a point $v \in \mathcal{D}$ such that the simplex $\rho = [F \cup \{\mathbf{v}\}]$ satisfies properties (1)-(2) and $\rho \cap \mathcal{D} = F \cup \{\mathbf{v}\}$;
  - $\widetilde{\sigma} \leftarrow \rho$ and $\widetilde{\mathbf{p}} \leftarrow \mathbf{v}$;
  - return $(\widetilde{\mathbf{p}}, \widetilde{\sigma})$.
- Let $\mathbf{v} \in \mathcal{D} \cap \left(\underset{\rho \in \triangle}{\cup} \rho\right)$ such that $\mathbf{u}^T \cdot \mathbf{v} - \lambda > 0$. If any new facet $\tau \subset [F \cup \{\mathbf{v}\}]$ satisfies $\tau \cap \rho \subset \mathcal{D}$ for any$\rho \in \triangle$ then set $\widetilde{\sigma} \leftarrow [F \cup \{\mathbf{v}\}]$ and $\widetilde{\mathbf{p}} \leftarrow \mathbf{v}$;
- return $(\widetilde{\mathbf{p}}, \widetilde{\sigma})$.

The Figure 11 pictures the initial steps of the evolution of the algorithm. Firstly, the initial simplex σ_0, shown in Figure 11.(a) is obtained at the initialization by exhaustive search. Then, the execution enters the mail loop of the algorithm generating by pivoting the simplices σ_1, σ_2 and σ_3 in the first, second and third iterations, respectively.



(a)



(b)

Figure 11. Evolution of Algorithm 1 for data set $\{\mathbf{p}_0, \mathbf{p}_1, ..., \mathbf{p}_{12}\}$: (a) Initial simplex. (b) Simplices generated after three iterations of the main loop.

It can be shown that the triangulation Γ generated by the Algorithm 1 has the property $\underset{\sigma \in \Gamma}{\cup} \sigma = co(\mathcal{D})$. This triangulation can be used to represent a manifold defined by the *Implicit Function Theorem* [44] through a piecewise linear manifold with some degree of precision. For our interest, the piecewise linear manifolds will be defined through a Characteristic Function $\chi : \mathcal{D} \rightarrow \{-1, 1\}$ that may represent the labels in a classification task. For such a function, the following definitions will be useful.

Definition 8. We call an $1 - face\ \tau$ of a triangle σ *completely labeled* respected to $\chi : \mathcal{D} \rightarrow \{-1, 1\}$ if this function changes its signal in τ.

Definition 9. A triangle (or face) σ in $\mathbb{R}^{N+1}$ is called

transverse with respect to χ if it contains a completely labeled $1-face$.

Given a k-face $\tau \in \Gamma$, let us define an affine map that coincides with χ on the vertices of τ as follows:

$$\chi_\tau : \tau \to \mathbb{R};$$

$$\chi_\tau(\mathbf{v}) = \sum_{i=1}^{k+1} \alpha_i \chi(\mathbf{v}_i),\ \mathbf{v} = \sum_{i=1}^{k+1} \alpha_i \mathbf{v}_i, \tag{37}$$

We can straightforward use this definition to extend the characteristic function to the set $|\Gamma|$:

$$\chi_\Gamma : |\Gamma| \to \mathbb{R}; \quad \chi_\Gamma(\mathbf{v}) = \chi_\sigma(\mathbf{v}), \tag{38}$$

for $\mathbf{v} \in \sigma$ and $\sigma \in \Gamma$.

Finally, we can state a fundamental theorem for our work.

Theorem 1. If σ is a triangle of a triangulation Γ of $\mathbb{R}^{N+1}$ which has a non-empty intersection with $\chi_\Gamma^{-1}(0)$, then $M_\sigma = \sigma \cap \chi_\Gamma^{-1}(0)$ is a $N-dimensional$ polytope, and the family:

$$M_\Gamma = \left\{ M_\sigma \mid \sigma \in \Gamma,\ \sigma \cap \chi_\Gamma^{-1}(0) \neq \emptyset \right\},$$

is a piecewise linear manifold of dimension N.

Dem. See [44].

The following algorithm describes the fundamental steps for obtaining the piecewise linear manifold approximating $\chi_\Gamma^{-1}(0)$. The heart of the procedure is a continuation methodology that keeps track of all vertices of each transverse simplex σ which remain to be checked in order to find all possible new transverse simplices by pivoting. The step-by-step of the procedure marches through the triangulation generated in the Algorithm 1. Therefore, it is not necessary to apply any consistency test during its execution [44].

Algorithm 2: Piecewise Linear Manifold Generation Algorithm

- **0** Input: Set $\mathcal{D} = \{\mathbf{p}_0, ..., \mathbf{p}_m\} \subset \mathbb{R}^{N+1}$; Triangulation Γ and Characteristic function $\chi : \mathcal{D} \to \{-1, 1\}$.
- **1** Output: piecewise linear manifold M_Γ approximating $\chi_\Gamma^{-1}(0)$.
- **2** Initialization
  - **2.1**. Find a transverse simplex $\sigma \subset \Gamma$;
  - **2.2**. Actual list of transverse simplices: $\sum = \{\sigma\}$;
  - **2.3**. $V(\sigma)$ = set of vertices of σ;
  - **2.4**. Actual list of manifold patches: $M_\Gamma = \sigma \cap \chi_\Gamma^{-1}(0)$
- **3** While $V(\sigma) \neq \emptyset$ for some $\sigma \in \sum$,
  - **3.1**. get $\sigma = \left[\mathbf{p}_{i_0}, ..., \mathbf{p}_{i_{N+1}}\right] \in \sum$ such that $V(\sigma) \neq \emptyset$;
  - **3.2**. get $\mathbf{p} \in V(\sigma)$ such that the facet with vertices in $F = \left\{\mathbf{p}_{i_0}, ..., \mathbf{p}_{i_{N+1}}\right\} - \{\mathbf{p}\}$ is transverse;
  - **3.3**. $(\widetilde{\mathbf{p}}, \widetilde{\sigma}) = Pivoting\_Transverse(\mathbf{p}, \sigma)$;
  - **3.4**. drop $\mathbf{p}$ from $V(\sigma)$;
  - **3.5**. if $\widetilde{\sigma}$ is not transverse then
    - **3.5.2**. go to **3**.
  - **3.6**. else
    - if $\widetilde{\sigma} \in \sum$ then drop $\widetilde{\mathbf{p}}$ from $V(\widetilde{\sigma})$;
    - else
      - $\sum \leftarrow \sum \cup \{\widetilde{\sigma}\}$;
      - $V(\widetilde{\sigma})$= set of vertices of $\widetilde{\sigma}$;
      - drop $\widetilde{\mathbf{p}}$ from $V(\widetilde{\sigma})$.
      - $M_\Gamma \leftarrow \widetilde{\sigma} \cap \chi_\Gamma^{-1}(0)$
  - **3.7**. Go to **3**.

Procedure: Obtain $\widetilde{\sigma}$ from σ by pivoting the vertex p into $\widetilde{\mathbf{p}}$:

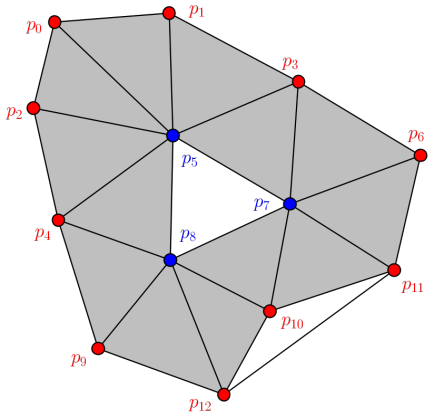
- $(\widetilde{\mathbf{p}}, \widetilde{\sigma}) = Pivoting\_Transverse(\mathbf{p}, \sigma)$
- Take the facet with vertex set $F = \left\{\mathbf{p}_{i_0}, ..., \mathbf{p}_{i_{N+1}}\right\} - \{\mathbf{p}\}$;
- Find the transverse simplex $\widetilde{\sigma}$ that satisfies $\widetilde{\sigma} \cap \sigma \supset F$;
- Find $\{\widetilde{\mathbf{p}}\} = V(\widetilde{\sigma}) - F$
- return $(\widetilde{\mathbf{p}}, \widetilde{\sigma})$.

The Figure 12 helps to understand the main stages of Algorithm 2. The Figure 12.(a) shows a set of points with *blue* and *red* labels, as well as the set of transverse triangles in gray level. Each transverse triangle holds a patch of the one dimensional piecewise linear manifold (a polygonal line in this case) which composes the output shown in Figure 12.(b).

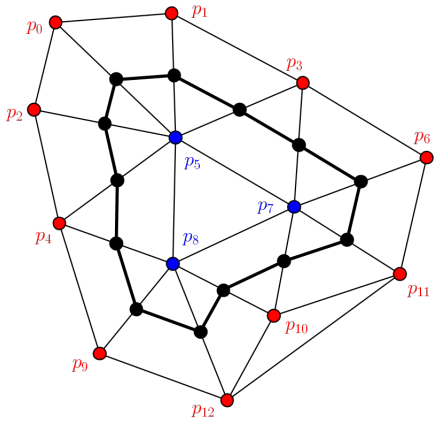
In this paper, we apply the above framework for synthesis and data analysis. Thus, let us return to the set TY in the representational space, defined by expression (36). Let us take the triangulation Γ of $co(TY)$ obtained by Algorithm 1 with input $\mathcal{D} = TY$. Given a new $\tilde{ty} \in co(TY) - TY$, the goal of synthesis is to find its image $\mathbf{q} \in \mathcal{M}^d \subset \mathbb{R}^D$. For this, we firstly find a d-dimensional simplex $[\tilde{\mathbf{ty}}_0, \tilde{\mathbf{ty}}_1, \ldots, \tilde{\mathbf{ty}}_d] \in \Gamma$ containing $\tilde{\mathbf{ty}}$. In this case, the equation $\tilde{\mathbf{ty}} = \sum_{j=0}^{d} \alpha_j \tilde{\mathbf{ty}}_j$, with the constraint $\sum_{j=0}^{d} \alpha_j = 1$, has unique solution $(\alpha_0, \alpha_1, \cdots, \alpha_d)$. Since $\tilde{\mathbf{ty}}_i \in TY$, the simplex in $\mathbb{R}^d$ has a unique counterpart $[\tilde{\mathbf{p}}_0, \tilde{\mathbf{p}}_1, \cdots, \tilde{\mathbf{p}}_d]$ with vertices $\tilde{\mathbf{p}}_i \in \mathcal{D}$, the original database, which allows to write $\mathbf{q} = \sum_{j=0}^{d} \alpha_j \tilde{\mathbf{p}}_j$ as an linear estimate (synthesis) for the new sample $\mathbf{q}$ in the data space.

Starting from the simplex $[\tilde{\mathbf{ty}}_0, \tilde{\mathbf{ty}}_1, \ldots, \tilde{\mathbf{ty}}_d]$ a more general polytope σ can be generated and the technique described in [45] can be used to compute barycentric coordinates of $\tilde{\mathbf{ty}}$ with respect to σ. With such generalization we can smooth local variations of data samples and consequently increase the quality of the interpolation process.

The LRML together with the discrete geometry framework of this section can be used to analyse two class, say $C_1, C_2 \subset \mathbb{R}^{N+1}$, databases as follows: (a) Perform dimensionality reduction using LRML; (b) Compute the set TY in the representational space by using expressions (34)-(36); (c) Build a triangulation Γ using the Algorithm 1;



(a)



(b)

Figure 12. (a) Set of transverse triangles obtained by Algorithm 2 evolution. (b) One dimensional piecewise linear manifold (polygonal line in black).

(d) Define the characteristic function $\chi : \mathcal{D} \to \{-1, 1\}$ by setting C_1 samples with label $+1$ and the C_2 samples with label -1; (e) Apply the Algorithm 2 in order to get the N dimensional piecewise linear approximation of the manifold $\chi_\Gamma^{-1}(0)$, where χ_Γ is given by expression (38). The topology and geometry of the obtained piecewise linear manifold gives information about data distribution in the image space, like clusters and separation surfaces.

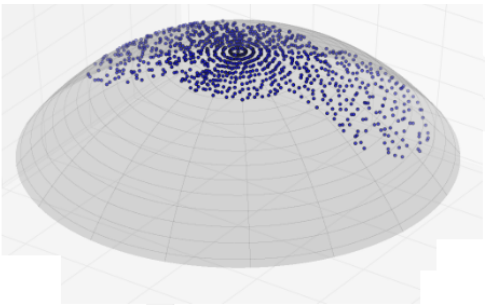
VIII. Computational Experiments

In this section we tested the LRML framework (section VI-D) and the syntheses technique of VII on synthetic data sets composed by sphere points as well as face images of FEI data base, maintained by the Department of Electrical Engineering of FEI, São Paulo, Brazil.

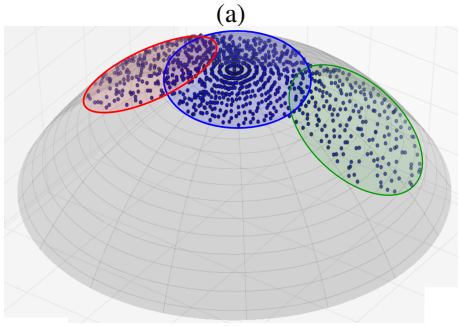
For performance comparisons we consider the traditional ISOMAP, LLE, and LTSA, summarized in section VI, as well as the fast RML [16]. The goal of this section is to check the LRML ability to preserve geodesic distances, the efficiency of the composition technique to get a representational space and the capabilities of the polyhedral geometry framework of section VII for synthesis and data analysis.

A. Sampling of Sphere

In this experiment we consider a synthetic 3-D data set $\mathcal{D}$, pictured on Figure 13.(a), uniformly generated from the unitary sphere. The objective of this experiment is to map data samples, originally embedded in a 3D space, onto a 2D plane. Therefore, this synthetic data provides a standard benchmark to evaluate the efficiency of the algorithms, because both input and output data can be easily visualized.



(a)



(b)

Figure 13. (a) Point samples over the sphere. (b) Neighborhoods of the data cover: $\mathcal{D}_1$ in red color, $\mathcal{D}_2$ in blue and $\mathcal{D}_3$ in green.

The raw data set $\mathcal{D}$ is processed by the procedure of section IV, with parameters $K = 40$, $\rho = 1.5$, and tolerance 10^{-5} (see section IV-C). The output is the topology of the data set given by the safe neighborhoods $SN(\mathbf{p}_i)$, $i = 1, 2, ..., 1086$. Then, the original data set is partitioned into 3 subsets $\mathcal{D}_1$, $\mathcal{D}_2$ and $\mathcal{D}_3$, by using the algorithm of section IV-B. The obtained neighborhoods, highlighted on Figure 13.(b), together with the manifold dimension $d = 2$, are used in the algorithm of section VI-D1 in order to compute the local normal coordinates. In this case, we did not apply the composition process of section VI-D2.

The Table I allows to compare the performance of the LRML algorithm against three traditional competing ones (ISOMAP, LLE, and LTSA) on the synthetic data using the following parameters: $K = 40$ and dimension $d = 2$. Table I reports the mean μ and standard deviation σ for the error associated to each obtained map. This error distribution for

Cluster	μ_{LLE}	μ_{LTSA}	μ_{ISOMAP}	μ_{LRML}
1	0.45490	0.45441	0.00411	0.00944
2	0.34195	0.34216	0.00184	0.00128
3	0.43712	0.43496	0.00696	0.00546
Cluster	σ_{LLE}	σ_{LTSA}	σ_{ISOMAP}	σ_{LRML}
1	0.22508	0.22487	0.00382	0.00590
2	0.18500	0.18510	0.00128	0.00127
3	0.20838	0.20721	0.00521	0.00307

Table I
MEAN ERROR (μ) AND STANDARD (σ) DEVIATION CORRESPONDING TO THE GEODESIC DISTANCES FOR THE CONSIDERED EMBEDDINGS.

Method	LLE	LTSA	ISOMAP	LRML+Comp.
Mean Error	0.67248	0.67224	0.00382	0.00840
Variance	0.39558	0.39530	0.00327	0.00780

Table II
COMPARING TRADITIONAL EMBEDDINGS AND THE OBTAINED REPRESENTATIONAL SPACE (LRML PLUS COMPOSITION).

each technique is computed as follows: (i) take a subset $\mathcal{D}_i$; (ii) for each pair of points $\mathbf{p}_i, \mathbf{p}_j \in \mathcal{D}_i$ compute the geodesic distance $\mathbf{d}_{\mathcal{M}^2}(\mathbf{p}_i, \mathbf{p}_j)$ and the corresponding distance in the reduced space $\mathbf{d}(\mathbf{z}_i, \mathbf{z}_j) = ||\mathbf{z}_i - \mathbf{z}_j||$; (iii) compute the matrix $C(i,j) = |\mathbf{d}_{\mathcal{M}^2}(\mathbf{p}_i, \mathbf{p}_j) - \mathbf{d}(\mathbf{z}_i, \mathbf{z}_j)|$; (iv) calculate the mean μ and the standard deviation σ of the elements of the matrix $\{C(i,j)\}$.

Table I shows that LRML and ISOMAP perform better then the competing ones in every case. The standard deviation for both LRML and ISOMAP are of order 10^{-3} which allows to compare their results through the reported mean error. The Table I shows that for the neighborhoods $\mathcal{D}_2$ and $\mathcal{D}_3$ the LRML performs better than the ISOMAP. However, things change for the neighborhood $\mathcal{D}_1$ and LRML achieves a mean error larger than the mean error of the LRML.

In the next example, we take the result of LRML and apply the composition procedure of section VI-D2 in order to get the representational space. The Figure 14 shows the obtained result in the reduced space as well as the global embedding generated by ISOMAP, LLE and LTSA. By visual inspection it is difficult to get some conclusion. So, we compute the global error distribution (steps (i)-(iv) above with $\mathcal{D}_i$ replaced by $\mathcal{D}$ and $\mathbf{d}(\mathbf{z}_i, \mathbf{z}_j)$ by $\mathbf{d}(\mathbf{ty}_i, \mathbf{ty}_j)$) which is reported on Table II. The LRML is outperformed only by the ISOMAP although the mean error and standard deviation of both methods remains in the same order of magnitude (10^{-3}).

Now, we exemplify the triangulation generated by the Algorithm 1 and piecewise linear manifolds obtained with the Algorithm 2. The Figure 15 shows the triangulation, in the representational space, obtained from the set of points pictured on Figure 14.(a). A visual analysis shows that the triangulation verifies the properties (1)-(3) of Definition 6. However, there are triangles that are not well-shaped (internal angles out of the range $[45^o, 90^o]$). We shall take

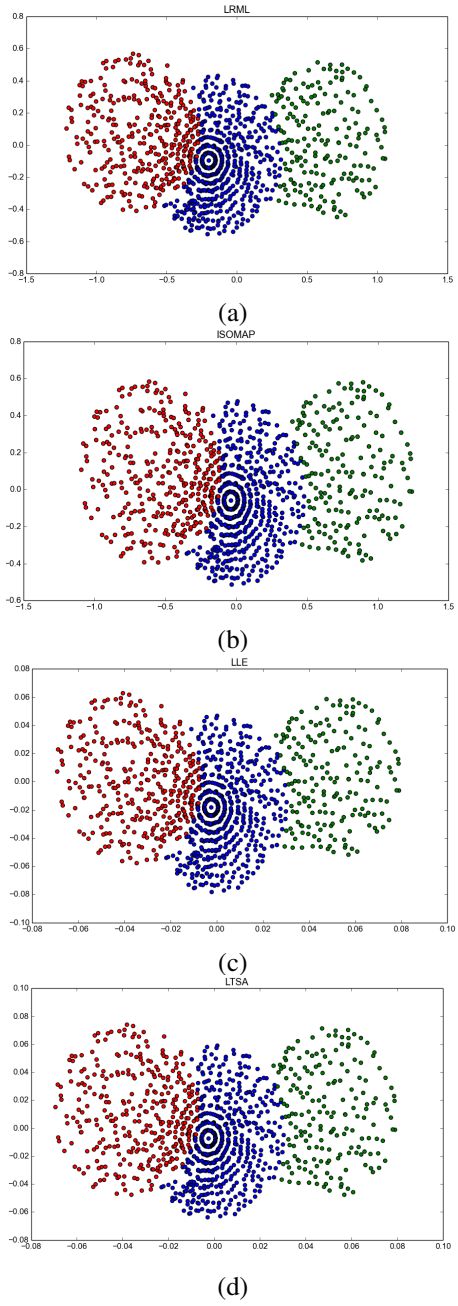


Figure 14. Comparing manifold learning methods applied to the sphere samples: (a) LRML plus composition process. (b) ISOMAP. (c) LLE. (d) LTSA.

into account that the Algorithm 1 did not perform any procedure to check the quality of the triangles. We avoid the computational cost of such step. For instance, the computational complexity to build a Delaunay triangulation from the points $\{\mathbf{ty}_1, \mathbf{ty}_2, \cdots, \mathbf{ty}_N\} \subset \mathbb{R}^d$ is $O(N^{\lceil d/2 \rceil})$ if $d \geq 3$, which is prohibitive for higher dimensions [46]. Besides, once the region $\bigcup_{\sigma \in \Gamma} \sigma$ is the convex hull of the input points we notice degenerate triangles nearby the boundary of the triangulation.

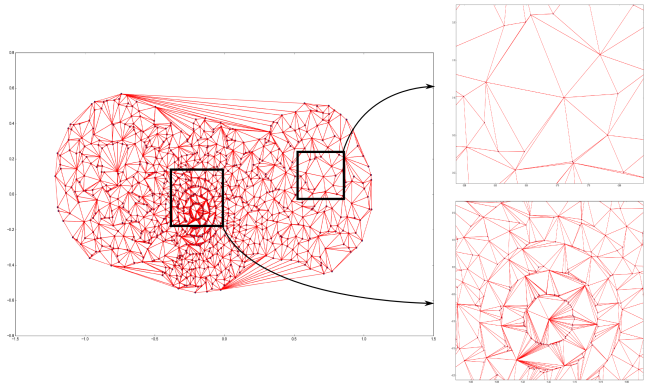


Figure 15. Triangulation obtained with Algorithm 1 from the points in Figure 14.(a), with zoom in some regions.

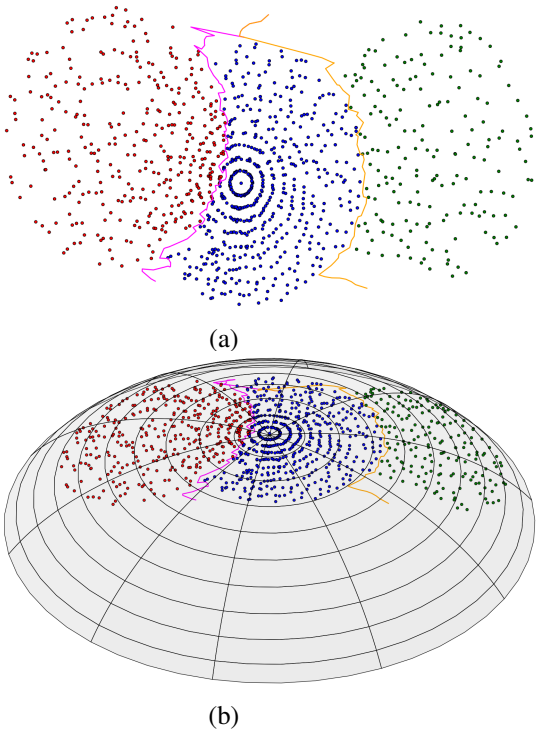


Figure 16. (a) Piecewise linear manifolds that separates clusters. (b) Exponential map image (over the sphere) of the piecewise linear manifolds.

The Figure 16.(a) pictures the one-dimensional piecewise linear manifolds that separate the set TY_1 from TY_2 and the set TY_2 from TY_3 in the representational space. We avoid to show the triangulation in order to simplify the visualization. The former is generated by the following steps: take the triangulation of the Figure 15 and define the characteristic function $\chi : S \rightarrow \{-1, 1\}$ by setting the samples (vertices) from the set TY_1 with label $+1$ and the remaining samples with label -1; apply the Algorithm 2 in order to extract the one-dimensional piecewise linear manifold. An analogous procedure is performed to build the other piecewise linear manifold. In this case, due to the low dimensions involved we can also visualize the image of the linear manifold under the exponential map in the original data space, as we can see in Figure 16.(b). Such result can be applied for classifying unlabeled samples based on the nearest projection distances from the samples in the piecewise linear manifolds. A similar approach is performed in [47] but without the dimensionality reduction step and the global triangulation generation.

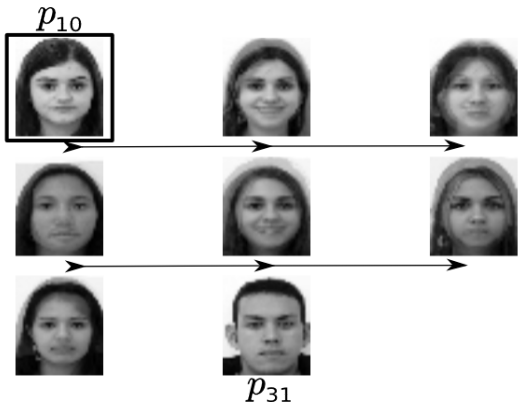
B. Real-World Data Set

In this section, the FEI database is used to illustrate the behavior of our algorithm. In this data set the number of subjects is equal to 200 (100 men and 100 women) and each subject has two frontal images (one with a neutral or non-smiling expression and the other with a smiling facial expression), performing a total of 400 images for the experiments. All faces are mainly represented by subjects between 19 and 40 years old with distinct appearance, hairstyle and adorns. The images are well scaled and aligned. These features make the FEI database very attractive for testing dimensionality reduction techniques. For memory requirements, we convert each pose to gray scale before computations and reduce the original spatial resolution (260×360) to 35×40.

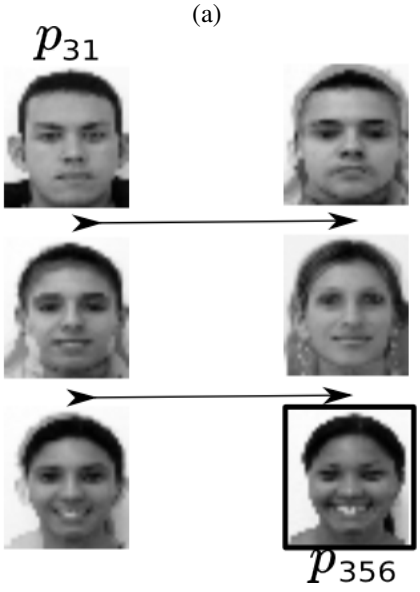
We execute the procedures of section IV to compute the topology of the data set and estimate the manifold dimension d. So, firstly, for each element $\mathbf{p}_i \in \mathcal{D}$ we find $K = 30$ nearest neighbors in the graph $G(\mathcal{D})$. Then, we determine the set of visible neighbors for $\mathbf{p}_i$, defined by expression (7). Next, we obtain the safe neighborhoods $SN(\mathbf{p}_i)$, using the parameters $\rho = 1.0$. The sets $SN(\mathbf{p}_i)$ define a topology of the data set, which allows to build the neighborhood system $\{\mathcal{D}_i, \quad i = 1, 2, \cdots, 11\}$, obtained with the procedure of section IV-B. The manifold dimension was estimated as $d = 12$ and is obtained by computing the dimension d_i for each $SN(\mathbf{p}_i)$, $i = 1, \cdots, 400$, through PCA, and averaging the obtained values. We apply tolerance 10^{-12} for the eigenvalues (see section IV-C). However, we decided to reduce the dimension to $d = 4$ (tolerance 0.01) in order to analyse the result considering a lower dimensional space.

We take two images, $\mathbf{p}_{10}$ and $\mathbf{p}_{356}$, highlighted in Figure 17 with continuum squares. Then, we consider a path linking these images and synthesize intermediate points along it.

Figures 17.(a) and 17.(b) allow to check the quality of the images generated by the synthesis process (section VII). In these figures, all images are synthetic ones, with exception of $\mathbf{p}_{10}$ and $\mathbf{p}_{356}$. Although the presence of some artifacts, we consider the image quality pretty good. We verify a gradual evolution from image $\mathbf{p}_{10}$ to $\mathbf{p}_{31}$ in Figure 17.(a). The same can be realized in Figure 17.(b) but in this case, from image $\mathbf{p}_{31}$ to $\mathbf{p}_{356}$.



(a)



(b)

Figure 17. (a) Sequence of images generated by our synthesis technique along the path connecting the images $\mathbf{p}_{10}$ to $\mathbf{p}_{31}$, indicated with continuum squares. (b) Analogous result but using images $\mathbf{p}_{31}$ and $\mathbf{p}_{356}$ of the data set.

IX. Perspectives

A. Foliations and Face Image Space

Roughly speaking, a foliation of dimension k of a differentiable manifold $\mathcal{M}^d$ is a decomposition of $\mathcal{M}^d$ into a disjoint family $\mathbb{F} = \{L_\alpha; \quad \alpha \in I\}$ of submanifolds, named leafs, each one with dimension k, as shown in Figure 18.

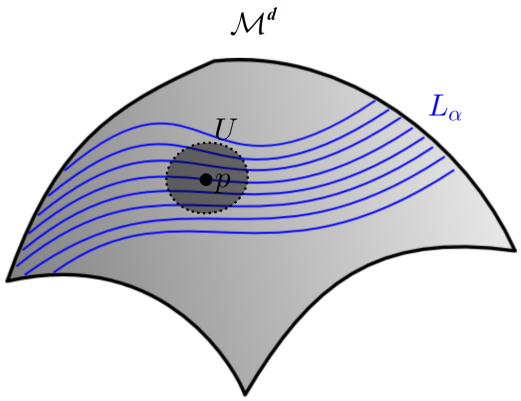
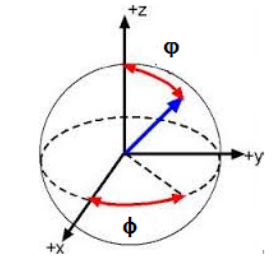
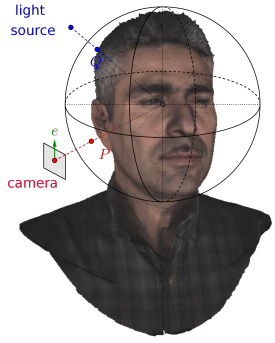


Figure 18. Foliation $\mathbb{F}$ with submanifolds L_α as leafs.

Our purpose in this subject is to apply the foliation theory to analyze the space of human face images. Specifically, let us take an acquisition set up which can be described by the geometric imaging model of Figure 19.



(a)



(b)

Figure 19. (a) Spherical coordinates to parameterize S^2. (b) Geometric elements of the face imaging model (Source [17]).

In this model the human head is placed in the center of a sphere $S = \{(x, y, z) \in \mathbb{R}^3; x = r\sin(\varphi)\cos(\phi), y = r\sin(\varphi)\sin(\phi), z = r\cos(\varphi)\}$, with $r > 0$, $0 \leq \phi < \pi; 0 \leq \varphi \leq \pi$, pictured on Figure 19.(a). We suppose that the human face is fixed and so we obtain different poses by moving the camera but its optical axis is set to pass through the sphere center (see Figure 19.(b)). The projection (camera) plane is supposed to be perpendicular to the optical axis. All the other geometric parameters (planar rotation angle of the camera, focal length of the camera, light position, etc.) and

physical parameters related to the illumination model are supposed to be fixed. Accordingly, the degrees of freedom are $(r, \phi, \varphi) \in \mathbb{R}^3$, and we can define a function:

$$f: \quad U \subset \mathbb{R}^3 \quad \rightarrow \quad \mathcal{M}, \tag{39}$$

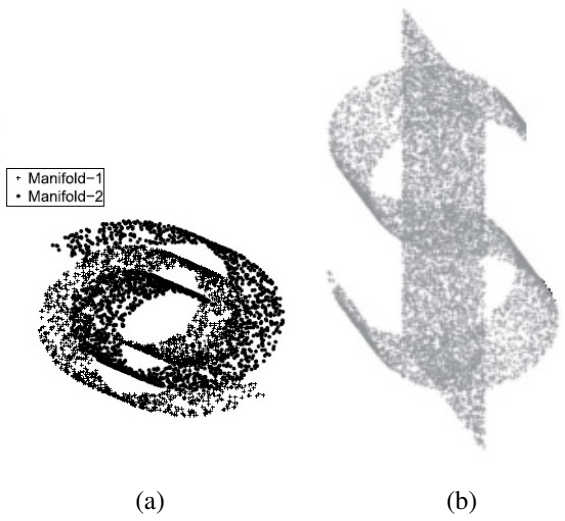
$$U = \{(r, \phi, \varphi)\,;\; r > 0, 0 \leq \phi, \varphi < \pi\}\,,$$

$$(r, \phi, \varphi) \xrightarrow{f} I\,(r, \phi, \varphi) \in \mathbb{R}^{N_1 \times N_2},$$

which, given the point $(r, \phi, \varphi) \in U$, calculates a digital image $I\,(r, \phi, \varphi)$ with resolution $N_1 \times N_2$. In this case, the manifold $\mathcal{M}$ is a set of digital images, which has dimension $d = 3$ and it is embedded in a space of dimension $N_1 \times N_2$ by f. This simple model is enough to show that for face images the idea that the input data resides on a low-dimensional manifold embedded in an environment space isomorphic to $\mathbb{R}^{N_1 \times N_2}$ makes sense, as observed in [17]. More specifically, if the function f in equation 39 is differentiable then the generated image space consists of a differentiable manifold with dimension $d = 3$. It is a known result of the theory of foliations that every compact and oriented manifold of dimension $d = 3$ admits a foliation with dimension $k = 2$. If an analogous result is valid for the manifold generated by the acquisition model of Figure 19, then we will have in hand a resource to organize the topology of the face image space: we have 'building blocks' of the face space formed by three-dimensional manifolds that hold the different poses for each person. These blocks can be decomposed according to the cited theorem. Does the space so generated admit a foliation? Also, does the space of expression have a similar organization?

Foliations can be also used to generalize the concept of manifold entanglement used in [25] to investigated the factors that contribute to the success of deep hierarchical representations of the data. Transversality (end of section II) is also useful in this context. To explain this, let us suppose that the data points are samples from two swiss rolls entangled within each other as shown in Figure 20.(a). In this case, the learning process should recognize and separate individual data manifolds for respective classes. Besides, the transition between the two manifolds will recover a foliation that could reveal structural changes when navigating between the classes. Also, the two manifolds could be transverse likewise in Figure 20.b.

A complete synthesis process in such case will generate samples belonging to two transverse foliations. In [25], authors propose a few quantities for measuring the geometric configurations represented in Figure 20 and to validate the hypothesis that deep architectures are able to compute the embedding of Figure 4, a process named flattening in [25]. However, we are still in the beginning of this research in order to translate deep learning machinery in manifold geometry elements.



(a) (b)

Figure 20. (a) Two entangled swiss rolls representing different data classes (source [25])) . (b) Transverse data sets (source [25]).

B. Manifold Learning and Deep Hierarchical Structures

Considering the data model of Figure 3 we can say that a promising learning technique must discover rich models that represent the probability distributions over the data and/or the geometry that is implicity in the data samples. We expect that deep learning approaches can fulfill these requirements and generate hierarchical models to describe the data.

In this section, we focus on generative adversarial network (GAN) models and deep metric learning [26], [27] because the former is directly related to the synthesis problem while the latter offers promising perspectives for RML techniques.

GANs are models that take a training set, consisting of samples drawn from an unknown distribution, and learns to synthesize new data samples [26]. However, GANs are not based in manifold learning. Instead, in such models a generative model G captures the data distribution, and a discriminative model D estimates the probability that a sample came from the training data rather than G.

The basic GAN model incorporates a generator's distribution p_g over data $\mathbf{x} \in \mathbb{R}^D$, a prior $p_{\mathbf{z}}$ on input noise variables $\mathbf{z} \in \mathbb{R}^d$, and represents the mapping to data space as $G\,(\mathbf{z}, \theta_g)$, where G is a differentiable function and θ_g is an array of parameters. In this scenario, the discriminator is a function $D\,(\mathbf{x}, \theta_d)$, parameterized by θ_d, that represents the probability that $\mathbf{x}$ came from the data rather than p_g. In practice, both G and D models are implemented as multilayer perceptrons with θ_g and θ_d being the network parameters. Along the training process we try to solve the problem:

$$\min_G \max_D V\,(G, D)\,, \tag{40}$$

where:

$$V(G,D) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left[\log\left(D\left(\mathbf{x}\right)\right)\right]$$

$$+\mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}\left[\log\left(1-D\left(G\left(\mathbf{z}\right)\right)\right)\right]. \quad (41)$$

Putting the expression (41) explicitly gives:

$$V(G,D) = \int_{\mathbf{x}} p_{data}\left(\mathbf{x}\right)\log\left(D\left(\mathbf{x}\right)\right)d\mathbf{x}$$

$$+\int_{\mathbf{z}} p_{\mathbf{z}}\left(\mathbf{z}\right)\log\left(1-D\left(G\left(\mathbf{z}\right)\right)\right)d\mathbf{z}. \quad (42)$$

It can be shown that the solution of the problem (40) is achieved if and only if $p_g = p_{data}$.

From the viewpoint of manifold learning, we shall notice that the generator is a function $G : \mathbb{R}^d \longrightarrow \mathcal{M}^d$ that can be seen as a global parametrization of the data manifold and the computation of G allows the execution of a synthesis procedure.

However, we know from the differential geometry literature that manifold parameterizations may be only locally defined. The unity sphere is the most known example of this fact. What are the implications of this fact for GANs? Besides, does G preserve data topology? A partial solution for this question would be to impose the following constraint to GAN model:

$$d_{\mathcal{M}^d}\left(G\left(\mathbf{z}_1\right), G\left(\mathbf{z}_2\right)\right) = ||\mathbf{z}_1 - \mathbf{z}_2||. \quad (43)$$

This constraint can be satisfied if the generator G computes the exponential map function (see Figure 2). In this case, the space $\mathbb{R}^d$, named latent space in GAN literature, would be a tangent space, as represented in the Figure 2. The distance $d_{\mathcal{M}^d}$ over the manifold can be approximated using shortest paths in the graph on the data samples, as performed elsewhere (see section VI-C, for instance). The open point is to set a multilayer perceptron that computes the exponential and logmap functions, defined in section II.

The metric concept is important because it can be seen as a similarity measure between data points. However, which metric makes sense? This question points towards deep metric learning techniques, like that one presented in [48], which considers the metric:

$$d_f\left(\mathbf{p}_1, \mathbf{p}_2\right) = ||f\left(\mathbf{p}_1\right) - f\left(\mathbf{p}_1\right)||, \quad (44)$$

where f is a nonlinear mapping, computed by a deep neural network, with discriminative properties. We can go a step further and use a similar methodology to seek for a mapping f such that $d_{\mathcal{M}^d}\left(\mathbf{p}_1, \mathbf{p}_2\right) = d_f\left(\mathbf{p}_1, \mathbf{p}_2\right)$. New issues related to the network architecture must be addressed to complete such task.

C. Discriminant Analysis and Manifolds

Let a differentiable curve $\alpha : (-\xi, \xi) \subset \mathbb{R} \rightarrow \mathcal{M}^d$ passing through two points $\mathbf{p}_i = \alpha\left(t_i\right)$ and $\mathbf{p}_j = \alpha\left(t_j\right)$, a differentiable function $f : \mathcal{M}^d \rightarrow \mathbb{R}$, and a local coordinate (U, φ) such that $\alpha\left(-\xi, \xi\right) \subset \varphi\left(U\right)$. So, the representations of f and α in the local parametrization $\varphi : U \rightarrow \mathcal{M}^d$ are:

$$f \circ \varphi\left(\mathbf{x}\right) = f\left(x_1, ..., x_n\right), \quad \mathbf{x} = \left(x_1, ..., x_n\right) \in U, \quad (45)$$

$$\varphi^{-1} \circ \alpha\left(t\right) = \left(x_1\left(t\right), ..., x_n\left(t\right)\right) \in U. \quad (46)$$

Therefore, considering the restriction of f to α and applying the chain rule we get:

$$\frac{d}{dt}\left(f \circ \alpha\right) = \frac{d}{dt} f\left(x_1\left(t\right), ..., x_n\left(t\right)\right) = \sum_{i=1}^{n} \frac{dx_i\left(t\right)}{dt}\frac{\partial f}{\partial x_i}. \quad (47)$$

Hence, if $\mathbf{p}_j$ and $\mathbf{p}_i$ are close each other in the manifold, we can write:

$$f\left(\mathbf{p}_j\right) \approx f\left(\mathbf{p}_i\right) + \frac{d}{dt} f\left(x_1\left(t\right), ..., x_n\left(t\right)\right)|_{t_i} dt, \quad (48)$$

which renders:

$$f\left(\mathbf{p}_j\right) \approx f\left(\mathbf{p}_i\right) + \left(\frac{\partial f}{\partial x_1}, ..., \frac{\partial f}{\partial x_d}\right)\left(dx_1, ..., dx_d\right)^T. \quad (49)$$

If we approximate $\left(dx_1, ..., dx_d\right)^T = P_{\mathbf{p}_i}^T\left(\mathbf{p}_j - \mathbf{p}_i\right)$, which is a projection of $\left(\mathbf{p}_j - \mathbf{p}_i\right)$ in the tangent space $T_{\mathbf{p}_i}\left(\mathcal{M}^d\right)$, we obtain:

$$f\left(\mathbf{p}_j\right) \approx f\left(\mathbf{p}_i\right) + \mathbf{v}_{\mathbf{p}_i}^T P_{\mathbf{p}_i}^T\left(\mathbf{p}_j - \mathbf{p}_i\right), \quad (50)$$

where $\mathbf{v}_{\mathbf{p}_i}^T = \left(\frac{\partial f}{\partial x_1}, ..., \frac{\partial f}{\partial x_d}\right)|_{\mathbf{p}_i}$.

Expression (50) is the key of the work presented in [49]. The function f serves as a connection between the data representation and discriminant analysis in $\mathcal{M}^d$. Specifically, from equation (50) we can write:

$$f\left(\mathbf{p}_i\right) \approx f\left(\mathbf{p}_j\right) + \mathbf{v}_{\mathbf{p}_j}^T P_{\mathbf{p}_j}^T\left(\mathbf{p}_i - \mathbf{p}_j\right). \quad (51)$$

The supervised methodology presented in [49] supposes that data in different classes, with labels in the set $\{1, 2, \ldots, C\}$, are generated from different manifolds. Provided that the labeled data set:

$$X = \left\{\left(\mathbf{p}_i, y_i\right); \mathbf{p}_i \in \mathbb{R}^D, y_i \in \{1, 2, \ldots, C\}, 1 \leq i \leq N\right\} \quad (52)$$

has been partitioned into L disjoint patches X_i, following the LDA philosophy [5], the within-class data structure over the manifold is obtained by constructing the within-class graph $G = \{X, W\}$ where W is a characteristic matrix compute as follows: if $\mathbf{p}_i$ belongs to the k-nearest neighbors of $\mathbf{p}_j$ and $y_i = y_j$ then $W_{ij} = W_{ji} = 1$. Otherwise, it is assumed that there is no edge connecting the points and $W_{ij} = 0$.

Besides, let $\pi_i \in \{1, 2, \ldots, L\}$ an index indicating the patch X_i to which $\mathbf{p}_i$ belongs to. In this way, we shall perform the changes $\mathbf{v}_{\mathbf{p}_i}^T \rightarrow \mathbf{v}_{\pi_i}^T$ and $P_{\mathbf{p}_i}^T \rightarrow P_{\pi_i}^T$, analogous for $\mathbf{p}_j$ in expressions (50)-(51). Then, if nearby data points $\mathbf{p}_i$ and $\mathbf{p}_j$ belong to the same class π_j, we can measure their similarity $s(\mathbf{p}_i, \mathbf{p}_j)$ as follows:

$$s(\mathbf{p}_i, \mathbf{p}_j) = \left[f(\mathbf{p}_i) - f(\mathbf{p}_j) - \mathbf{v}_{\pi_j}^T P_{\pi_j}^T (\mathbf{p}_i - \mathbf{p}_j) \right]^2. \tag{53}$$

From expressions (50) and (51), the similarity between nearby tangent spaces can be computed by:

$$s_T(\pi_i, \pi_j) = \mathbf{v}_{\pi_j}^T P_{\pi_j}^T - \mathbf{v}_{\pi_i}^T P_{\pi_i}^T, \tag{54}$$

and, consequently, if $s_T(\pi_i, \pi_j) \approx 0$ we get:

$$\mathbf{v}_{\pi_i} \approx P_{\pi_i}^T P_{\pi_j} \mathbf{v}_{\pi_j} \Longrightarrow ||\mathbf{v}_{\pi_i} \approx P_{\pi_i}^T P_{\pi_j} \mathbf{v}_{\pi_j}||_2^2 \approx 0 \tag{55}$$

if the tangent spaces $T_{\pi_i}\left(\mathcal{M}^d\right)$ and $T_{\pi_j}\left(\mathcal{M}^d\right)$ are similar. If we suppose that f in equation (53) is given by a linear function $f(\mathbf{p}) = \mathbf{t}^T \mathbf{p}$ then, putting all the above elements and results together we can build the following criterion to gather within-class information based on the manifold structure:

$$\min_{\mathbf{t}, \mathbf{v}} \sum_{i,j} W_{ij} \Theta_{ij}, \tag{56}$$

where:

$$\Theta_{ij} = \left(\mathbf{t}^T \mathbf{p}_i - \mathbf{t}^T \mathbf{p}_j - \mathbf{v}_{\pi_j}^T P_{\pi_j}^T (\mathbf{p}_i - \mathbf{p}_j) \right)^2$$

$$+\gamma ||\mathbf{v}_{\pi_i} - P_{\pi_i}^T P_{\pi_j} \mathbf{v}_{\pi_j}||_2^2$$

where γ is a parameter to balance the influence of between expressions (53) and (55). We can say that when solving (56) we are looking for a projection direction $\mathbf{t}$ that minimizes the within-class separability but steered by the manifold geometry implicit in the products $\mathbf{v}_{\pi_j}^T P_{\pi_j}^T$.

The model for the between-class information is based on a graph $\check{G} = \left\{X, \check{W}\right\}$, construct as follows:

$$\check{W}_{ij} = \frac{1}{N},\ y_i \neq y_j,$$

and:

$$\check{W}_{ij} = A_{ij} \left(\frac{1}{N} - \frac{1}{N_c} \right),\ y_i = y_j,$$

where:

$$A_{ij} = \exp\left(-\frac{||\mathbf{p}_i - \mathbf{p}_j||_2^2}{\sigma_i \sigma_j} \right),$$

if $\mathbf{p}_i \in KNN(\mathbf{p}_j)$ or $\mathbf{p}_j \in KNN(\mathbf{p}_i)$; and, $A_{ij} = 0$, otherwise.

Then, in [49] it is proposed the following objective function to separate nearby between-class data points:

$$\max_{\mathbf{t}} \sum_{i,j} \check{W}_{ij} \left(\mathbf{t}^{\mathbf{T}} \mathbf{p}_i - \mathbf{t}^T \mathbf{p}_j \right)^2. \tag{57}$$

In [49] it is shown that we can solve the problems (56) and (57) simultaneously in order to get the solution $\mathbf{t} \in \mathbb{R}^d$ and $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_C \in \mathbb{R}^d$.

In this formulation, it is supposed that samples from different classes belong to different manifolds that could be submanifolds of the data space. Could it make sense to suppose that these submanifolds are members of a foliation, with dimension d, of the $\mathbb{R}^D$ or some suitable subset? The geometry and topology of such foliation could be incorporated in the above formulation? On the other hand, if we suppose that the different classes lye over a single manifold $\mathcal{M}^d$, then we can formulate the classification problem using separating submanifolds $\mathcal{N} \subset \mathcal{M}^d$. The next section deals with such scenario.

D. Classification and Kernels in Manifolds

Classification of different sample groups over the manifold can be achieved by defining kernels on manifolds and computing separating hypersurfaces (submanifolds) through the kernel trick.

The foundations of kernel methods belongs to the reproducing kernel Hilbert spaces and Mercer theory. A remarkable results in this area is the Mercer theorem, which we summarized bellow [50]. So, let the space $\mathbb{R}^n$ and μ a finite measure in $\mathbb{R}^n$. We define also the function spaces $L_2(\mathbb{R}^n) = \left\{ f : \mathbb{R}^n \longrightarrow \mathbb{R};\ |f|^2\ is\ \mu - integrable \right\}$ and $L_\infty(\mathbb{R}^n) = \{ f : \mathbb{R}^n \longrightarrow \mathbb{R};\ \exists K > 0,\ |f(\mathbf{x})| \leq K \}$.

Mercer Theorem: Suppose $k : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}$ is a continuous symmetric positive definite function (kernel) such that $k \in L_\infty(\mathbb{R}^n \times \mathbb{R}^n)$. Under certain conditions, the integral operator $T_k : L_2(\mathbb{R}^n) \longrightarrow L_2(\mathbb{R}^n)$:

$$(T_k f)(\mathbf{x}) = \int_{\mathbb{R}} k(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) d\mu(\mathbf{y}), \tag{58}$$

has a set of normalized eigenfunctions $\psi_j : \mathbb{R}^n \longrightarrow \mathbb{R}$, with associated eigenvalues $\lambda_j > 0$, sorted in non-increasing order, such that: $k(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{n_F} \lambda_j \psi_j(\mathbf{x}) \psi_j(\mathbf{y})$. Either $n_F \in \mathbb{N}$ or $n_F = \infty$.

With this result, the kernel support vector machines (KSVM) generalizes their linear versions through the kernel function k which allows to write the hypersurface that separates positive from negative samples in the input space as [50]:

$$F(\mathbf{p}) \equiv \sum_{i=1}^{N} y_i \alpha_i k(\mathbf{p}_i, \mathbf{p}) + \widetilde{b} = 0, \tag{59}$$

where $\alpha_i \geq 0$, $i = 1, 2, \cdot, \cdot, \cdot, N$, are Lagrange multipliers in the quadratic optimization problem behind KSVM technique [50]. The samples $\mathbf{p}_i$ with $\alpha_i \neq 0$ are named

support vectors. If $n_F \in \mathbb{N}$ then, there exists a map $\Phi(\mathbf{p}) \equiv (z_1(\mathbf{p}), z_2(\mathbf{p}), z_3(\mathbf{p}), \ldots, z_{n_F}(\mathbf{p}))$ such that the separating surface in the feature space $\mathbb{R}^{n_F}$ is given by:

$$\sum_{r=1}^{n_F} \omega_r z_r(\mathbf{p}) + \widetilde{b} = 0 \tag{60}$$

where $\omega_r = \sum_{i=1}^{M} y_i \alpha_i z_r(\mathbf{p}_i)$.

All the above machinery remains at hand if we can define kernels $k : \mathcal{M}^d \times \mathcal{M}^d \longrightarrow \mathbb{R}$ on manifolds. In this way, the work [37] defines positive definite kernels on manifolds, based on Gaussian radial basis function, that permit to embed a given Riemannian manifold in a high dimensional reproducing kernel Hilbert space. The main consequence of this development is analogous to the one that emerges from expression (60): it makes possible to utilize algorithms developed for linear spaces (support vector machines, for instance) on data manifolds.

X. Conclusion

This paper covers the main topics involved in the application of manifold learning methods for image analysis: (a) Recover the data topology; (b) Determination of the manifold dimension d; (c) Construction of a neighborhood system; (d) Computing the embedding or local parameterizations associated to the neighborhood system; (d) Given a point in the representational space, compute its image in the data manifold (synthesis).

We review traditional techniques in this field, with particular attention to LRML method and the application of concepts in discrete and polyhedral geometry for synthesis and data clustering over the manifold. Also, we present practical examples using $3D$ data points as well as image analysis and synthesis through LRML applied to FEI database. The material helps to understand advanced concepts, advantages and drawbacks of nonlinear dimensionality reduction based on manifold learning. Besides, the perspectives in the field, including topological structure of image spaces and application of manifold learning concepts for deep learning analysis and understanding, opens future directions in manifold learning and related topics.

References

[1] J. A. Lee and M. Verleysen, *Nonlinear Dimensionality Reduction*, 1st ed. Springer Publishing Company, Incorporated, 2007.

[2] J. Wang, *Geometric Structure of High-Dimensional Data and Dimensionality Reduction*. Springer Berlin Heidelberg, 2012.

[3] J. P. Cunningham and Z. Ghahramani, "Linear dimensionality reduction: Survey, insights, and generalizations," *Journal of Machine Learning Research*, vol. 16, pp. 2859–2900, 2015.

[4] D. Engel, L. Hüttenberger, and B. Hamann, "A Survey of Dimension Reduction Methods for High-dimensional Data Analysis and Visualization," in *Proceedings of IRTG 1131 Workshop 2011*, vol. 27. Germany: Schloss Dagstuhl, 2012, pp. 135–149.

[5] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning," *Springer*, 2001.

[6] H. Safavi and C.-I. Chang, "Projection pursuit-based dimensionality reduction," *Proc. SPIE*, vol. 6966, pp. 69 661H–69 661H–11, 2008.

[7] G. Baudat and F. Anouar, "Generalized discriminant analysis using a kernel approach," *Neural Computation*, vol. 12, no. 10, pp. 2385–2404, 2000.

[8] C. H. Park and H. Park, "Nonlinear discriminant analysis using kernel functions and the generalized singular value decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 27, no. 1, pp. 87–102, 2005.

[9] B. Scholkopf, A. Smola, and K.-R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[10] A. Kuleshov and A. Bernstein, *Proc. 10th International Conference, St. Petersburg, Russia, July 21-24*, 2014, ch. Manifold Learning in Data Mining Tasks, pp. 119–133.

[11] Y. Ma and Y. Fu, *Manifold Learning Theory and Applications*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 2011.

[12] A. J. Izenman, *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning*, 1st ed. Springer Publishing Company, Incorporated, 2008.

[13] E. C. Kitani, C. E. Thomaz, and G. A. Giraldi, "Geometric Elements of Manifold Learning," National Laboratory for Scientific Computing and FEI Universitary Center, Tech. Rep., 2011.

[14] Y. Goldberg, A. Zakai, D. Kushnir, and Y. Ritov, "Manifold learning: The price of normalization," *J. Mach. Learn. Res.*, vol. 9, pp. 1909–1939, June 2008.

[15] T. Lin and H. Zha, "Riemannian Manifold Learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 796–809, 2008.

[16] A. Brun, C. Westin, M. Herberthson, and H. Knutsson, "Fast manifold learning based on riemannian normal coordinates," in *Image Analysis*, ser. LNCS. Springer, 2005, vol. 3540, pp. 920–929.

[17] G. F. M. Jr., G. A. Giraldi, C. E. Thomaz, and D. Millan, "Composition of local normal coordinates and polyhedral geometry in riemannian manifold learning," *IJNCR*, vol. 5, no. 2, pp. 37–68, 2015.

[18] S. N. B. Dubrovin, A. Fomenko, *Modern Geometry: Methods and Applications*. Springer,Verlag, 1990.

[19] M. P. Carmo, *Geometria Riemanniana*. Livros Técnicos e Científicos Editora S.A., 1979.

[20] K. Sun and S. Marchand-Maillet, "An information geometry of statistical manifold learning," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 2014, pp. 1–9.

[21] G. F. M. Jr., G. A. Giraldi, C. E. Thomaz, and R. D. Millan., "Aprendizagem e síntese de variedades via coordenadas normais de riemann locais e baricentricas," in *Proc. of the ENIAC*, Fortaleza, Ceará, Brazil, 20th-24th October 2013.

[22] J. Wang, *Local Tangent Space Alignment*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 221–234.

[23] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[24] A. Samat, P. Gamba, S. Liu, P. Du, and J. Abuduwaili, "Jointly informative and manifold structure representative sampling based active learning for remote sensing image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 11, pp. 6803–6817, Nov 2016.

[25] P. P. Brahma, D. Wu, and Y. She, "Why deep learning works: A manifold disentanglement perspective," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 10, pp. 1997–2008, Oct 2016.

[26] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680.

[27] J. Hu, J. Lu, Y. P. Tan, and J. Zhou, "Deep transfer metric learning," *IEEE Transactions on Image Processing*, vol. 25, no. 12, pp. 5576–5588, Dec 2016.

[28] B. Dubrovin, A. Fomenko, and S. Novikov, "Modern geometry: Methods and applications," 1992.

[29] M. do Carmo, *Geometria Riemanniana*. Projeto Euclides. Instituto de Matemática Pura e Aplicada, 1988.

[30] E. Lima, *Variedades diferenciáveis*. Monografias de matemática. Instituto Matemática Puro e Aplicada, Conselho Nacional de Pesquisas, 1973.

[31] L. K. Saul and S. T. Roweis, "Think globally, fit locally: Unsupervised learning of low dimensional manifolds," *J. Mach. Learn. Res.*, vol. 4, pp. 119–155, December 2003.

[32] J. Tenenbaum, V. Silva, and J. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[33] E. Kitani, C. E. Thomaz, and G. A. Giraldi, "Geometric elements of manifold learning," Laboratrio Nacional de Computao Cientfica, Tech. Rep., Relatório Técnico No. 5/2011.

[34] L. Shi, Q. Yang, Y. Xu, and P. He, "A model of selecting the parameters based on the variance of distance ratios for manifold learning algorithms," in *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 2, Aug 2009, pp. 507–512.

[35] L. Feng, C. Gao, T. Sun, and H. Wu, "A neighborhood selection algorithm for manifold learning," in *2010 International Conference On Computer Design and Applications*, vol. 2, June 2010.

[36] J. He, L. Ding, L. Jiang, Z. Li, and Q. Hu, "Intrinsic dimensionality estimation based on manifold assumption," *Journal of Visual Communication and Image Representation*, vol. 25, no. 5, pp. 740 – 747, 2014.

[37] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M. Harandi, "Kernel methods on riemannian manifolds with gaussian rbf kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 12, pp. 2464–2477, Dec 2015.

[38] F. Camastra, "Data dimensionality estimation methods: a survey," *Pattern Recognition*, vol. 36, no. 12, pp. 2945 – 2954, 2003.

[39] T. Cox and M. Cox, *Multidimensional Scaling*. Chapman and Hall, 1994.

[40] Z. Zhang and H. Zha, "Principal manifolds and nonlinear dimension reduction via local tangent space alignment," *SIAM Journal of Scientific Computing*, vol. 26, pp. 313–338, 2002.

[41] G. Golub and C. Van Loan, *Matrix Computations*, ser. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.

[42] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, no. 1, p. 269271, 1959.

[43] D. Meng, Y. Leung, T. Fung, and Z. Xu, "Nonlinear dimensionality reduction of data lying on the multicluster manifold," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Trans. on*, vol. 38, no. 4, pp. 1111–1122, Aug 2008.

[44] E. L. Allgower and K. Georg, *Numerical Continuation Methods: An Introduction*. Springer-Verlag Berlin Heidelberg, 1990.

[45] J. D. Warren, "Barycentric coordinates for convex polytopes." *Adv. Comput. Math.*, vol. 6, no. 1, pp. 97–108, 1996.

[46] J.-D. Boissonnat, O. Devillers, and S. Hornus, "Incremental construction of the delaunay triangulation and the delaunay graph in medium dimension," in *Proceedings of the Twenty-fifth Annual Symposium on Computational Geometry*, ser. SCG '09. New York, NY, USA: ACM, 2009, pp. 208–216.

[47] J. Zhang, Z. Xie, and S. Z. Li, "Prime discriminant simplicial complex." *IEEE Trans. Neural Netw. Learning Syst.*, vol. 24, no. 1, pp. 133–144, 2013.

[48] J. Lu, J. Hu, and Y. P. Tan, "Discriminative deep metric learning for face and kinship verification," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4269–4282, Sept 2017.

[49] Y. Zhou and S. Sun, "Manifold partition discriminant analysis," *IEEE Transactions on Cybernetics*, vol. 47, no. 4, pp. 830–840, April 2017.

[50] V. N. Vapnik, "Statistical learning theory," *John Wiley & Sons, INC.*, 1998.