Noname manuscript No.
(will be inserted by the editor)

Nonlinear Dimensionality Reduction by Topologically Constrained Isometric Embedding

Guy Rosman · Michael M. Bronstein · Alexander M. Bronstein · Ron Kimmel

Received: date / Accepted: date

Abstract Many manifold learning procedures try to embed a given feature data into a flat space of low dimensionality while preserving as much as possible the metric in the natural feature space. The embedding process usually relies on distances between neighboring features, mainly since distances between features that are far apart from each other often provide an unreliable estimation of the true distance on the feature manifold due to its non-convexity. Distortions resulting from using long geodesics indiscriminantly lead to a known limitation of the Isomap algorithm when used to map non-convex manifolds. Presented is a framework for nonlinear dimensionality reduction that uses both local and global distances in order to learn the intrinsic geometry of flat manifolds with boundaries. The resulting algorithm filters out potentially problematic distances between distant feature points based on the properties of the geodesics connecting those points and their relative distance to the boundary of the feature manifold, thus avoiding an inherent limitation of the Isomap algorithm. Since the proposed algorithm matches non-local structures, it is robust to strong noise. We show experimental results demonstrating the advantages of the proposed approach over conventional dimensionality reduction techniques, both global and local in nature.

This research was partly supported by United States-Israel Binational Science Foundation grant No. 2004274, by the Israel Science Foundation (grant no. 623/08) by the Ministry of Science grant No. 3-3414, and by the Elias Fund for Medical Research.

Guy Rosman · Michael M. Bronstein · Alexander M. Bronstein · Ron Kimmel
The Department of Computer Science, Technion – Israel Institute of Technology, Haifa, 32000
Tel.: +972-4-8294895
Fax: +972-4-8293900
E-mail: rosman,mbron,bron,ron@cs.technion.ac.il

Keywords Dimensionality Reduction · Isomap · Differential Geometry · Machine Learning · Image Manifolds

1 Introduction

Analysis of high-dimensional data is encountered in numerous pattern recognition applications. In many cases, it appears that just a small number of dimensions is needed to explain the high-dimensional data.

For example (Tenenbaum et al 2000), consider a large set of images with an underlying parameter space of a small dimension. One example for such a manifold is the set of all images of an object sampled at certain poses, after being centered. This manifold is represented by vectors of a high dimension (e.g., the column stacked images), but is of a much lower intrinsic dimensionality – the Euler angles representing the pose of the object, are one such possible parametrization.

Dimensionality reduction methods such as *principal component analysis* (PCA, see Duda et al 2000) and *multidimensional scaling* (MDS, see Borg and Groenen 1997) are often used to obtain a low dimensional representation of the data, which is a commonly used preprocessing stage in pattern recognition.

The principal components analysis algorithm linearly projects the points to a low dimensional space by minimizing the least square fitting error. Multidimensional scaling algorithms minimize the error in the pairwise distances between data points, and are intimately related to PCA (e.g., see Borg and Groenen 1997; Williams 2002).

While methods such as PCA assume the existence of a linear map between the data points and the parametrization space, such a map often does not exist. Ap-

plying linear dimensionality reduction methods to data therefore results in a distorted representation.

Nonlinear dimensionality reduction (NLDR) methods attempt to describe a given high-dimensional set of points as a low dimensional manifold by means of a nonlinear map preserving certain properties of the data. This kind of analysis has applications in numerous fields, such as color perception, pathology tissue analysis (Coifman et al 2005), enhancement of MRI images (Diaz and Arencibia 2003), shape recognition (Elad and Kimmel 2003), face recognition (Bronstein et al 2005), motion understanding (Pless 2003), and biochemistry (Keller et al 2005), to mention a few.

As the input data, we assume to be given N points in the M-dimensional Euclidean space, $\{\mathbf{z}_i\}_{i=1}^N \subset \mathbb{R}^M$. The points constitute vertices of a proximity graph with the set of edges E; the points $\mathbf{z}_i, \mathbf{z}_j$ are neighbors if $(i,j) \in E$. The data points are further assumed to be samples of an m-dimensional manifold $\mathcal{M} \subset \mathbb{R}^M$, where typically $m \ll M$. This manifold together with the geodesic metric $d_{\mathcal{M}}$ defined on $\mathcal{M}$ form a metric space. The manifold is represented by a *parametrization domain* $\mathcal{C}$ using the smooth bijective map $\varphi : \mathcal{C} \subset \mathbb{R}^m \to \mathcal{M}$.

The goal of NLDR methods is to uncover the parametrization of $\mathcal{M}$. More precisely, we are looking for a set of points $\{\mathbf{x}_i \approx \varphi^{-1}(\mathbf{z}_i)\}_{i=1}^N \subset \mathcal{C} \subset \mathbb{R}^m$ parameterizing the data. We will try to compute the $N \times m$ matrix $\mathbf{X}$ representing the coordinates of the points in the parametrization domain.

Many NLDR techniques attempt to find an m-dimensional representation for the data, while preserving *local* properties. For example, the *locally linear embedding* (LLE) algorithm (Roweis and Saul 2000) tries to preserve the representation of each data point as a linear combination of its neighbors. The *Laplacian eigenmap* algorithm (Belkin and Niyogi 2002) uses the Laplacian operator for selecting low dimensionality coordinate functions based on its eigenfunctions. The resulting coordinate system maps neighboring points in $\mathcal{M}$ to neighboring points in $\mathbb{R}^m$. The *diffusion map* (Coifman et al 2005) generalize this framework in the context of analysis of diffusion processes, making it more robust to non-uniform sampling density. The *Hessian locally linear embedding* (HLLE, Grimes and Donoho 2003) tries to use the proximity graph for finding coordinate functions that have a minimal response to the Hessian operator of the surface, obtaining a truly locally linear mapping.

Another class of algorithms preserves *global* properties, like the *geodesic distances* $d_{\mathcal{M}}(z_i, z_j)$, approximated as shortest paths on the proximity graph. The geodesic distance $d_{\mathcal{M}}(z_i, z_j)$ is defined as

$$d_{\mathcal{M}}(z_i, z_j) = \min_{c'(z_i, z_j)} l(c'(z_i, z_j)),$$

where $l(c'(z_i, z_j))$ denotes the length of the curve $c'(z_i, z_j)$, and minimization over all curves $c'(z_i, z_j)$ connecting z_i and z_j is obtained by a geodesic

$$c_{\mathcal{M}}(z_i, z_j) = \operatorname*{argmin}_{c'(z_i, z_j)} l(c'(z_i, z_j)).$$

The *Semidefinite embedding* (Weinberger and Saul 2004) algorithm maximizes the variance in the data set while keeping the local distances unchanged, thereby approximately preserving geodesic distances in the manifold. The problem is formulated and solved as a semidefinite programming (SDP, Ben-Tal and Nemirovski 2001) problem, under constraints reflecting invariance to translation and local isometry of the manifold to Euclidean space. Yet, the computational cost for solving an SDP problem is $O(N^6)$ (see Ben-Tal and Nemirovski 2001, for details), which is prohibitive even in medium-scale problems. Attempts to overcome it by using *landmarks* (Weinberger et al 2005) still incur high computational complexity.

Brand (2005) describes an algorithm which utilizes global distances in order to numerically stabilize and robustify local embedding techniques, but the approach presented is still mostly local in nature, assuming that a few randomly selected longer acting connections are enough to prevent the weaknesses of local techniques while still keeping their attractive computational time.

Finally, the *Isomap* algorithm (Schwartz et al 1989; Tenenbaum et al 2000) considers both local and *global* invariants – the lengths of geodesics between points on the manifold. Short geodesic distances are assumed to be equal to Euclidean distances, and longer ones are approximated as shortest paths length on the proximity graph, using standard graph search methods like Dijkstra's algorithm (Dijkstra 1959; Cormen et al 1990). The resulting distance measure, $\delta_{ij} = \delta(\mathbf{z}_i, \mathbf{z}_j)$, approximates $d_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j)$ under certain assumptions, as shown by Bernstein et al (2001). Isomap then uses multidimensional scaling, attempting to find an m-dimensional Euclidean representation of the data, such that the Euclidean distances between points are as close as possible to the corresponding geodesic ones. For example, using the L_2 criterion (referred to as *stress*),

$$\mathbf{X}^* = \operatorname*{argmin}_{\mathbf{X} \in \mathbb{R}^{N \times m}} \sum_{i<j} \left(d_{ij}(\mathbf{X}) - \delta_{ij}\right)^2,$$

where $d_{ij}(\mathbf{X}) = \|\mathbf{x}_i - \mathbf{x}_j\|_2$ is the Euclidean distance between points $\mathbf{x}_i$ and $\mathbf{x}_j$ in $\mathbb{R}^m$. Instead of Dijkstra's algorithm, higher accuracy algorithms such as fast marching methods (Kimmel and Sethian 1998) can be used

when dealing with surfaces, resulting in a more accurate mapping (Zigelman et al 2002; Elad and Kimmel 2003).

The main advantage of Isomap is that it uses global geometric invariants, which are relatively less sensitive to measurement noise, compared to local ones. Yet, its underlying assumption is that $\mathcal{M}$ is *isometric* to $\mathcal{C} \subset \mathbb{R}^m$ with the *geodesic metric* $d_{\mathcal{C}}$, induced by the Riemannian structure of $\mathcal{C}$. This metric is different in general from the metric of $\mathbb{R}^m$ restricted to $\mathcal{C}$, referred to as the *restricted* metric and denoted by $d_{\mathbb{R}^m}|_{\mathcal{C}}$. That is, Isomap assumes $\delta(\mathbf{z}_i, \mathbf{z}_j) = d_{\mathbb{R}^m}(\mathbf{x}_i, \mathbf{x}_j)$ for all $i, j = 1, ..., N$. If $\mathcal{C}$ is convex, the restricted metric $d_{\mathbb{R}^m}|_{\mathcal{C}}$ coincides with the geodesic metric $d_{\mathcal{C}}$ and Isomap succeeds in recovering the parametrization of $\mathcal{M}$. Otherwise, $\mathcal{C}$ has no longer the Euclidean geometry and no isometric map of the dataset to $\mathbb{R}^m$ can be found. The convexity assumption of $\mathcal{C}$ appears to be too restrictive, as many data manifolds have complicated topology. Indeed, Grimes and Donoho (2002) showed examples of data for which $\mathcal{C}$ is not convex, and pointed out that Isomap fails in such cases. Lack of convexity may stem from the structure of the data themselves or from incomplete measurements. In any case, non-convex data is sufficiently common so as to hinder the use of the Isomap algorithm. A note about intrinsic convexity of the manifold: If an isometric embedding of the manifold into Euclidean space results in a convex region, another isometric embedding cannot result in a non-convex region. This result is implicit in the discussion made by Grimes and Donoho (2002), and can be shown by noting that between each two points in R^m exists a single curve with the same length as the line connecting them.

1.1 Contribution

In this paper, we claim that even when violating the convexity assumption, one could still use non-local distances in order to stabilize and robustify the flattening procedure. We do that by detecting and ignoring geodesic distances which may be inconsistent with the underlying convexity assumption.

Our approach, hereinafter referred to as the *topologically constrained isometric embedding* (TCIE), allows handling flat data manifolds with arbitrary boundaries and "holes" that may often occur when sampling natural phenomena. A rough sketch of the approach has been presented in a conference paper (Rosman et al 2006). Here, we provide a better picture of it, in terms of the algorithms used, proofs of their validity, and comparison to existing techniques. We further present an additional example for which a ground truth parameter space is known and where our algorithm manages to handle such non-convex data. We note that while our approach bears some resemblance to methods for robustifying Isomap against topological noise such as the approach presented by Choi and Choi (2007), which deals with pointwise noise and its effect on the topology of the embedded manifold, our paper tackles a more fundamental limitation of the Isomap algorithm apparent even in an ideal noise-less setting.

The rest of this paper is organized as follows. In Section 2, we introduce the algorithm and prove that it rejects inconsistent geodesics. Section 3 discusses the numerical implementation of the algorithm and suggests ways to speed up its convergence. In Section 4, we demonstrate our approach on synthetic data. Proofs of supporting propositions are presented in the appendix.

2 Topologically Constrained Isometric Embedding

In order to construct an isometric embedding, the Isomap algorithm assumes that $\mathcal{C}$ is a convex subset of $\mathbb{R}^m$, and relies on the assumption of an isometry between $(\mathcal{C}, d_{\mathcal{C}})$ and $\mathcal{M}$ in order to find the map from $\mathcal{M}$ to the metric space $(\mathcal{C}, d_{\mathcal{C}})$ by means of MDS (the stress in the solution will be zero). This assumption is valid because $d_{\mathcal{C}} = d_{\mathbb{R}^m}|_{\mathcal{C}\times\mathcal{C}}$ if $\mathcal{C}$ is convex. In case $\mathcal{C}$ is non-convex, however, as there may exist pairs of points for which $d_{\mathcal{C}} \neq d_{\mathbb{R}^m}|_{\mathcal{C}\times\mathcal{C}}$. We call such pairs *inconsistent*. An example of an inconsistent pair is shown in Figure 1. We

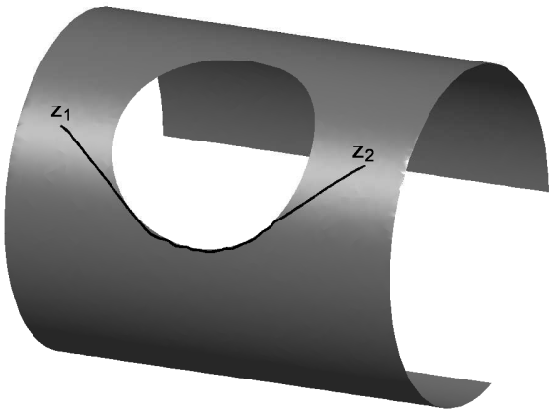


Fig. 1 Example of two points $\mathbf{z}_1$ and $\mathbf{z}_2$, for which the straight line connecting the points after embedding into $\mathbb{R}^2$ is shorter than the geodesic $c_{\mathcal{M}}(\mathbf{z}_1, \mathbf{z}_2)$ (solid black curve), due to non-convexity.

denote the set of all consistent pairs by

$$P = \{(i, j) : d_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) = d_{\mathbb{R}^m}|_{\mathcal{C}\times\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j)\} \subseteq I_N \times I_N.$$

where $I_N = \{1 \dots N\}$. In the TCIE algorithm, we find a subset $\bar{P} \subseteq P$ of pairs of points that can be consistently represented by an MDS problem. The algorithm is as follows

1: Compute the $N \times N$ matrix of *geodesic distances* $\boldsymbol{\Delta} = (\delta_{ij})$.
2: Detect the boundary points $\partial\mathcal{M}$ of the data manifold.
3: Detect a subset of *consistent distances* according to either

$$\bar{P}_1 = \{(i,j) : c_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j) \cap \partial\mathcal{M} = \emptyset\},$$

(criterion 1), where $c_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j)$ is the geodesic connecting $\mathbf{z}_i$ and $\mathbf{z}_j$, or (criterion 2)

$$\bar{P}_2 = \{(i,j) : d_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j) \leq d_{\mathcal{M}}(\mathbf{z}_j, \partial\mathcal{M}) + d_{\mathcal{M}}(\mathbf{z}_i, \partial\mathcal{M})\},$$

where $d_{\mathcal{M}}(\mathbf{z}, \partial\mathcal{M}) = \inf_{\mathbf{z}' \in \partial\mathcal{M}} d_{\mathcal{M}}(\mathbf{z}, \mathbf{z}')$ denotes the distance of $\mathbf{z}$ from the boundary of $\mathcal{M}$.
4: Solve the MDS problem for consistent pairs only,

$$\mathbf{X}^* = \underset{\mathbf{X}}{\operatorname{argmin}} \sum_{i=0, i<j} w_{ij} (d_{ij}(\mathbf{X}) - \delta_{ij})^2,$$

where $w_{ij} = 1$ if $(i,j) \in \bar{P}$ and $w_{ij} = 0$ otherwise.

The choice of ignoring inconsistent pairs identified by either $\bar{P}_1$ or $\bar{P}_2$ can be generalized, for example, to smoother weight functions. The three main steps of the algorithm are (i) detection of boundary points, (ii) detection of a set of consistent geodesics, (iii) solution of a weighted MDS problem. In the sequel we will detail each of these steps.

2.1 Detection of Boundary Points

Step 2 in Alg. 1 involves detection of boundary points in multidimensional data. There exist many algorithms for the detection of boundaries in point clouds. Most of the related papers focus on practical applications for surface processing and modeling (see for example Boult and Kender 1986; Gopi 2002; Freedman 2002; Belton and Lichti 2007), though some algorithms emerged from the field of numerical solution for PDEs (Haque and Dilts 2007), or metric geometry (Chazal et al 2007), while other techniques for boundary detection and local dimensionality estimation were motivated by perceptual research (Guy and Medioni 1997; Tong et al 2004). Most of these methods are limited by design to specific intrinsic and extrinsic dimensions (m and M respectively), although some methods are more easily adaptable to higher dimensional data (see for example Mordohai and Medioni 2005; Chazal et al 2007). We show here a few methods for solving this generic problem and refer the reader to existing literature for a broader overview.

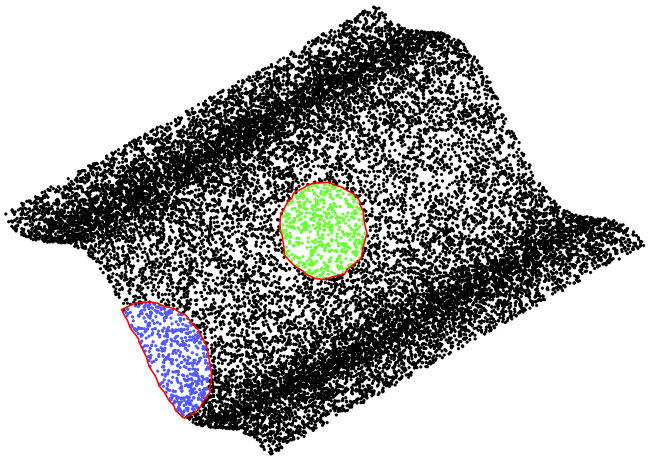


Fig. 2 An example of two neighborhoods of points in $\mathcal{M}$, one of which is close to the boundary $\partial\mathcal{M}$. In the example, $N = 20000$, and each neighborhood was chosen to include the 500 nearest neighbors of a point on the manifold.

One approach for boundary detection on high dimensional manifolds is based on the observation that each boundary point in an m-dimensional Riemannian manifold is locally homeomorphic to a half-space of $\mathbb{R}^m$, where the boundary point is mapped to the hyperplane bordering the half-space in $\mathbb{R}^m$. An example of two such neighborhoods is shown for a two-dimensional manifold (surface) in Figure 2.

We therefore expect the boundary point to have all its neighbors on one side of a single hyperplane in an m-dimensional mapping of its neighborhood. Multidimensional scaling of the neighborhood distances matrix can be used to obtain such a mapping.

Looking at the normal direction to this hyperplane, the mean of the neighboring points should be far from the boundary point. The first boundary detection method we present follows this line of thought.

1: **for** $i = 1, ..., N$ **do**
2: Find the set $\mathcal{N}(i)$ of the K nearest neighbors of the point i.
3: Apply MDS to the $K \times K$ matrix $\boldsymbol{\Delta}_K = (\delta_{k,l \in \mathcal{N}(i)})$ and obtain a set of local coordinates $\mathbf{x}'_1, ..., \mathbf{x}'_K \in \mathbb{R}^m$, where $\mathbf{x}'_1$ denotes the mapping of point i.
4: Compute $\boldsymbol{\mu}(i) = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} \mathbf{x}_i$, the mean of the neighbors of $\mathbf{x}_1$. $|\mathcal{N}(i)|$ denotes the cardinality of $\mathcal{N}(i)$.
5: Set $\bar{d}_1(i)$ to be the distance between $\mathbf{x}'_1$ and $\boldsymbol{\mu}(i)$, normalized by the average distance between points in $\mathbf{x}'_1, ..., \mathbf{x}'_K$.
6: **if** $\bar{d}_1(i)$ is larger than some threshold τ_a **then**
7: Label point i as `boundary`.
8: **end if**
9: **end for**

The described method is similar to that of Belton and Lichti (2007), where each coordinate is normalized separately, based on the standard deviation of points along the tangent space direction. Since our neighborhoods are chosen according to a uniform metric, we only normalized with respect to the neighbourhood diameter.

The second algorithm is similarly motivated, yet is more heuristic in nature. Directions are selected according to neighboring points, this avoiding the need to artificially determine the normal direction, or using the mean of the points, which may be sensitive to sampling. Assuming that for an interior point, for all directions, the distribution of projected points is homogeneous, the algorithm tries to detect directions for which the projection of the sampled neighboring points has a single-sided distribution. This scheme uses neighboring points in order to determine directions of projection, voting among possible directions in order to obtain a more robust classification. This may be done as follows,

1: **for** $i = 1, ..., N$ **do**
2: Find the set $\mathcal{N}(i)$ of the K nearest neighbors of the point i.
3: Apply MDS to the $K \times K$ matrix $\mathbf{\Delta}_K = (\delta_{kl \in \mathcal{N}(i)})$ and obtain a set of local coordinates $\mathbf{x}'_1, ..., \mathbf{x}'_K \in \mathbb{R}^m$.
4: **for** $j = 1, ..., K$ **do**
5: If $\frac{\left|\{\mathbf{x} \in \mathcal{N}(i) : \langle \mathbf{x}'_i - \mathbf{x}'_j, \mathbf{x} - \mathbf{x}'_i \rangle > 0\}\right|}{\left|\{\mathbf{x} \in \mathcal{N}(i) : \langle \mathbf{x}'_i - \mathbf{x}'_j, \mathbf{x} - \mathbf{x}'_i \rangle \leq 0\}\right|} \leq \tau_c$ mark j as `candidate`.
6: **end for**
7: **if** the number of `candidate` points is larger than τ_d **then**
8: Label point i as `boundary`.
9: **end if**
10: **end for**

Another property, which may be useful in boundary detection is the fact that a small neighborhood around an interior point should be isotropic in shape and sampling, whereas it should be anisotropic for a boundary point, as seen in the example in Figure 2. Belton and Lichti (2007), however, claim that directly using this property in an algorithm would be too sensitive to non-uniform sampling.

In our experiments, the proposed algorithms 2 and 3 performed similarly on noisy data. Other boundary detection algorithms can be used as well. We expect a voting mechanism (Tong et al 2004) to be quite beneficial for robust detection of the boundaries. For manifolds with a large intrinsic dimensionality, dense sampling is usually required for reliable boundary detection.

2.2 Detection of Inconsistent Geodesics

An important part of the TCIE algorithm is the detection of inconsistent pairs. We find all point pairs adhering to consistency criteria (1) or (2), which include all inconsistent pairs, as we shall prove.

The first consistency criterion requires us to check whether geodesics touch the boundary. Once we have detected the boundary points, we use a modification of the Dijkstra algorithm (Dijkstra 1959), as summarized below, using a notation similar to the one used by Cormen et al (1990).

1: **for** $\mathbf{z}_u \in \mathcal{M} \setminus \{\mathbf{z}_s\}$ **do**
2: $d(\mathbf{z}_s, \mathbf{z}_u) \leftarrow \infty$
3: **end for**
4: $d(\mathbf{z}_s, \mathbf{z}_s) \leftarrow 0$
5: Let Q be a queue of remaining vertices, sorted according to current distance to $\mathbf{z}_s$. Let S be the set of all vertices whose distance to $\mathbf{z}_s$ has already been fixed by the algorithm. Set $w_{ij} \leftarrow 1$ for all point pairs i, j.
6: **while** $Q \neq \varnothing$ **do**
7: Let $\mathbf{z}_u$ be the minimum distance vertex stored in Q
8: Add $\mathbf{z}_u$ to S
9:
10: **for** $\mathbf{z}_v \in \mathcal{N}_u$ **do**
11: **if** $d(\mathbf{z}_s, \mathbf{z}_v) > d(\mathbf{z}_s, \mathbf{z}_u) + d_{\mathbb{R}^M}(\mathbf{z}_u, \mathbf{z}_v)$ **then**
12: $d(\mathbf{z}_s, \mathbf{z}_v) \leftarrow d(\mathbf{z}_s, \mathbf{z}_u) + d_{\mathbb{R}^M}(\mathbf{z}_u, \mathbf{z}_v)$
13: **if** $w_{su} = 0$ or ($\mathbf{z}_u \in \partial\mathcal{M}$ and $d(\mathbf{z}_s, \mathbf{z}_u) > 0$) **then**
14: $w_{sv} = 0$
15: **else**
16: $w_{sv} = 1$
17: **end if**
18: **end if**
19: **end for**
20: **end while**

Note that the second condition in line 13 of the algorithm protects paths with only a boundary end point from being removed. This way we eliminate only geodesics for which the point of intersection with the boundary is a midpoint. Similar modifications can be made to the Bellman-Ford and Floyd algorithms, or other dynamic programming algorithms (for example, Kimmel and Sethian 1998). We note that for the criterion defining $\bar{P}_2$, detection of inconsistent geodesics is done simply by comparing the relevant geodesic distances.

In describing the algorithm, we assume a continuous case, in which the manifold is sampled with some given density. We make the same assumptions on the sampling density and uniformity made by Bernstein et al

(2001), who proved the convergence of the graph distances approximation, used by the Isomap algorithm (Schwartz et al 1989; Tenenbaum et al 2000), to the geodesic distances on the manifold. Also note that the requirement of a positive density function prevents problems that may occur in geodesics approximated by a graph when the surface is sampled in a regular pattern (as is the case with a Cartesian grid covering $\mathbb{R}^m$). In our case, there is also the question of whether or not we remove too many geodesics. The answer is related to the topology of the manifold.

In the continuous setting, our algorithm approximates an isometry φ^{-1} between $\mathcal{M}$ with the geodesic metric δ and $\mathcal{C} \subset \mathbb{R}^m$ with the geodesic metric $d_{\mathcal{C}}$. In the case where $\mathcal{C}$ is a convex region, geodesics connecting points in $\mathcal{C}$ are always straight lines, and the geodesic metric is identical to the restricted Euclidean metric. When $\mathcal{C}$ is no longer a convex region, $\bar{P}_1$ restricts our choice of point pairs, selecting only consistent distances, as shown by the following proposition.

Proposition 1 *Let $\mathcal{M}$ be a manifold, isometric to $\mathcal{C} \subseteq \mathbb{R}^m$, and $c_{\mathcal{M}}(\cdot\cdot)$ denote geodesics in $\mathcal{M}$. Then $\bar{P}_1 = \{(i,j) : c_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j) \cap \partial\mathcal{M} = \emptyset\} \subseteq P$.*

Therefore, for every geodesic in $\mathcal{M}$ which was not detected as touching the boundary, the image under φ^{-1} is a line, which is approximated correctly by the MDS procedure.

In the case where $\mathcal{C}$ is no longer a subset of a Euclidean space, but rather part of a manifold $\mathcal{C}'$ endowed with another metric, we claim that selecting only point pairs from $\bar{P}_2$ still leaves us only with consistent pairs.

Proposition 2 *For $\mathcal{C}$ and $\mathcal{C}'$ as described above,*

$$\bar{P}_2 = \{(i,j) : d_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j) \leq d_{\mathcal{M}}(\mathbf{z}_j, \partial\mathcal{M}) + d_{\mathcal{M}}(\mathbf{z}_i, \partial\mathcal{M})\} \subseteq P$$

Proofs of Propositions 1 and 2 are given in the appendix.

Note that for a parametrization manifold $\mathcal{C}'$ with an arbitrary Riemannian metric, the MDS procedure would not be able to give us the correct mapping. This would require the use of a more general procedure, as done by Bronstein et al (2006a). Criterion 2 may still be useful in cases where the metric on $\mathcal{C}'$ is close to Euclidean, and yet we only want to use geodesics which do not leave $\mathcal{C}$.

2.3 Weighted LS-MDS

The final stage of our approach is solving the MDS problem for the subset $\bar{P}_i, i \in \{1,2\}$ of distances. One way to include only consistent point pairs in the optimization is to use the *weighted stress*,

$$\mathbf{X}^* = \operatorname*{argmin}_{\mathbf{X}} \sum_{i=0,i<j} w_{ij}(d_{ij}(\mathbf{X}) - \delta_{ij})^2,$$

where $w_{ij} = 1$ if $(i,j) \in \bar{P}$ and $w_{ij} = 0$ otherwise. This allows us, by choosing the right weights, to minimize the error only for consistent geodesics.

The geodesics that were not marked as inconsistent have their weights set to one. We also allow a positive weight for short geodesics, in order to keep the connectivity of the manifold, even at boundary points. All other geodesics have their weights set to zero. We then use the *Scaling by Majorizing a Complicated Function* (SMACOF) algorithm, as detailed in subsection 3.1, to minimize the weighted stress.

We note that the correctness of these conditions depends on the assumption that our manifold is isometric to a subregion of an Euclidean space, similarly to the underlying assumption of Isomap.

3 Implementation Considerations

For determining the shortest paths we used the Dijkstra algorithm implementation supplied by Tenenbaum et al (2000), with the Isomap code, to which the detection of geodesics touching boundary points was added. The remaining components of the TCIE algorithm were implemented in MATLAB. In practice, the Dijkstra algorithm takes less than 10% of the total running time for 2000 points, with asymptotic complexity of $O(N^2 \log N)$. Solving the MDS optimization problem consumes most of the time (although $O(N^2)$ per iteration). The boundary detection takes $O(N^2)$, but with much smaller constants. We note that while the number of SMACOF iterations is not invariant to the number of samples, in practice it rises slowly with increase of N, depending on the topology of the manifold and the noise level.

3.1 The SMACOF algorithm

We now turn to the problem of minimizing the weighted stress function,

$$s(\mathbf{X}) = \sum_{i<j} w_{ij}(d_{ij}(\mathbf{X}) - \delta_{ij})^2,$$

which known as the *least-squares MDS*, or LS-MDS, problem. Trying to solve the LS-MDS problem, we consider the gradient of $s(\mathbf{X})$ with respect to $\mathbf{X}$, which can be written (Borg and Groenen 1997) as

$$\nabla s(\mathbf{X}) = 2\mathbf{V}\mathbf{X} - 2\mathbf{B}(\mathbf{X})\mathbf{X},$$

where $\mathbf{V}$ and $\mathbf{B}$ are matrices whose elements are given by

$$(\mathbf{V})_{ij} = \begin{cases} -w_{ij} & \text{if } i \neq j \\ \sum_{k \neq i} w_{ik} & \text{if } i = j \end{cases}$$

and

$$(\mathbf{B})_{ij} = \begin{cases} -w_{ij}\delta_{ij} d_{ij}^{-1}(\mathbf{X}) & \text{if } i \neq j \text{ and } d_{ij}(\mathbf{X}) \neq 0 \\ 0 & \text{if } i \neq j \text{ and } d_{ij}(\mathbf{X}) = 0 \\ -\sum_{k \neq i} b_{ik} & \text{if } i = j \end{cases}$$

In order to minimize the stress function, the following iterative scheme was proposed by Guttman (1968); de Leeuw (1977, 1984, 1988). From the first-order optimality condition, one obtains the iteration equation

$$\mathbf{X}^{(k+1)} = \mathbf{V}^{\dagger}\mathbf{B}(\mathbf{X}^{(k)})\mathbf{X}^{(k)}. \quad (1)$$

Iteratively performing the transformation (1) converges to a local minimum of the stress cost function. This process is known as the *SMACOF algorithm* (see e.g. Borg and Groenen 1997). It can be shown to be equivalent to a weighted gradient descent with constant step size (Bronstein et al 2006b).

A remarkable property of the SMACOF algorithm is that it guarantees a monotonously decreasing sequence of stress values, which is uncommon when constant-step gradient descent is used. This property is shown by developing the iteration formula (1) using a technique known as *iterative majorization*. At the same time, the convergence of the SMACOF algorithm is slow, and a large number of iterations may be required for high accuracy, depending on the size of the data set and the metric used.

3.2 Numerical Properties and Convergence

The LS-MDS optimization problem is a non-convex one, and as such convex optimization methods might converge to local minima (Trosset and Mathar 1997). In our experiments, we have seen that removing more distances from the stress function caused the problem to be liable to local convergence. Such local minima appear as a fold over, or a flip in the obtained mapping. In general, the number of remaining weights depends on the surface topology, as well as the number of sampled points in the surface [1].

We reduce the risk of local convergence by starting from a classical scaling (as mentioned by Trosset et. al. (Kearsley et al 1998; Aharon and Kimmel 2006)) or unweighted least squares scaling solution. This allows the algorithm to avoid some of the local minima. Although the solutions found by classical scaling and LS-MDS may differ, under the assumption of correct distance approximation, the solutions are likely to have a similar structure. Another possible benefit of using classical scaling prior to least square scaling is to use the recursive subspaces property of the classical MDS in order to approximately determine the intrinsic dimension of the manifold prior to its exact embedding. The estimated dimension can be used both in the weighted least-squares mapping stage and for the detection of boundary points. The validity of such an approximation for non-convex manifolds is beyond the scope of this paper and is deferred to future work.

Using the unweighted LS-MDS problem to avoid local minima, and then gradually changing the problem into the weighted one has the flavor of graduated non-convexity (Blake and Zisserman 1987). Using such a gradual approach for changing between the full and weighted MDS problems did not seem to significantly improve robustness to local convergence in our experiments, and ways of better utilizing such an approach remain a subject for future research.

3.3 Convergence Acceleration by Vector Extrapolation Methods and Multiresolution

To speed up the convergence of the SMACOF iterations, we employ *vector extrapolation*, as described in Rosman et al (2008). Vector extrapolation methods use a sequence of solutions at subsequent iterations of the optimization algorithm and extrapolate the limit of the iterations sequence. While these algorithms were derived assuming a linear iterative scheme, in practice, they work well also for nonlinear schemes, such as some processes in computational fluid dynamics (Sidi 1991). For further details, we refer to works by Cabay and Jackson (1976), Mešina (1977), Eddy (1979) and Smith et al (1987), as well as to a technical report where vector extrapolation acceleration of multidimensional scaling is detailed (Rosman et al 2008).

The main idea of vector extrapolation is, given a sequence of solutions $\mathbf{X}^{(k)}$ from iterations $k = 0, 1, ...$, to approximate the limit $\lim_{k\to\infty} \mathbf{X}^{(k)}$, which should coincide with the optimal solution $\mathbf{X}^{*}$. The extrapolation $\hat{\mathbf{X}}^{(k)}$ is constructed as an affine combination of the last $K+1$ iterates, $\mathbf{X}^{(k)}..\mathbf{X}^{(k+K)}$

$$\hat{\mathbf{X}}^{(k)} = \sum_{j=0}^{K} \gamma_j \mathbf{X}^{(k+j)}; \qquad \sum_{j=0}^{K} \gamma_j = 1.$$

The coefficients γ_j can be determined in various ways. In the *reduced rank extrapolation* (RRE) method, which

[1] Typically, in our experiments $\mathbf{W}$ contained between 6% to 18% nonzero weights.

is the extrapolation method used in the present study, γ_j are obtained by the solution of the minimization problem,

$$\min_{\gamma_0,..,\gamma_K} \| \sum_{j=0}^{K} \gamma_j \Delta \mathbf{X}^{(k+j)} \|, \quad \text{s.t.} \sum_{j=0}^{K} \gamma_j = 1,$$

where $\Delta \mathbf{X}^{(k)} = \mathbf{X}^{(k+1)} - \mathbf{X}^{(k)}$. This minimization can be shown to seek for a solution minimizing the residual in the linear case.

An efficient implementation of the RRE and minimal polynomial extrapolation (MPE) algorithms is described by Sidi (1991). The algorithm proceeds as follows:

1: Choose the integers k and n, and input the vectors $\mathbf{x}_n, \mathbf{x}_{n+1}, \ldots, \mathbf{x}_{n+k}$.
2: Form the $N \times k+1$ matrix $\mathbf{U}_k^{(n)}$ whose $k+1$ columns are $\Delta \mathbf{X}^{(k+j)}, j = 0, \ldots, k$.
3: Solve the overdetermined linear system $\mathbf{U}_k^{(n)} \boldsymbol{\gamma} = \mathbf{0}$, where $\boldsymbol{\gamma} = [\gamma_0, \gamma_1, \ldots, \gamma_k]^{\mathrm{T}}$, by least squares, subject to the constraint $\sum_{i=0}^{k} \gamma_i = 1$ (RRE).
4: Compute the vector $\mathbf{s}_{n,k} = \sum_{i=0}^{k} \gamma_i \mathbf{x}_{n+i}$ as approximation to $\lim_{i \to \infty} \mathbf{x}_i = \mathbf{s}$.

After obtaining $\mathbf{s}_{n,k}$, we use it as an initial solution for additional iterations of the SMACOF algorithm, followed by extrapolation, and so forth. This is known as *cycling*, and is a commonly used technique for extrapolating nonlinear sequences (Smith et al 1987).

Another way to accelerate the solution of the MDS problem is using *multiresolution* (MR) methods (see e.g. Chalmers 1996; Platt 2004; Silva and Tenenbaum 2004; Bronstein et al 2006b; Brandes and Pich 2007). The main idea is to subsequently approximate the solution by solving the MDS problem at different resolution levels.

At each level, we work with a *grid* consisting of points with indices $\Omega_L \subset \Omega_{L-1} \subset \ldots \subset \Omega_0 = \{1, ..., N\}$, such that $|\Omega_l| = N_l$. At the lth level, the data is represented as an $N_l \times N_l$ matrix $\boldsymbol{\Delta}_l$, obtained by extracting the rows and columns of $\boldsymbol{\Delta}_0 = \boldsymbol{\Delta}$, corresponding to the indices Ω_l. The solution $\mathbf{X}_l^*$ of the MDS problem on the lth level is transferred to the next level $l-1$ using an *interpolation operator* P_l^{l-1}, which can be represented as an $N_{l-1} \times N_l$ matrix.

1: Construct the hierarchy of grids $\Omega_0, \ldots, \Omega_L$ and interpolation operators $P_1^0, \ldots, P_L^{L-1}$.
2: Start with some initial $\mathbf{X}_L^{(0)}$ at the coarsest grid, and $l = L$.
3: **for** l=L,L-1,...0 **do**
4: Solve the lth level MDS problem
$\mathbf{X}_l^* = \underset{\mathbf{X}_l \in \mathbb{R}^{N_l \times m}}{\operatorname{argmin}} \sum_{i,j \in \Omega_l} w_{ij} (d_{ij}(\mathbf{X}_l) - \delta_{ij})^2$
using SMACOF iterations initialized with $X_l^{(0)}$.

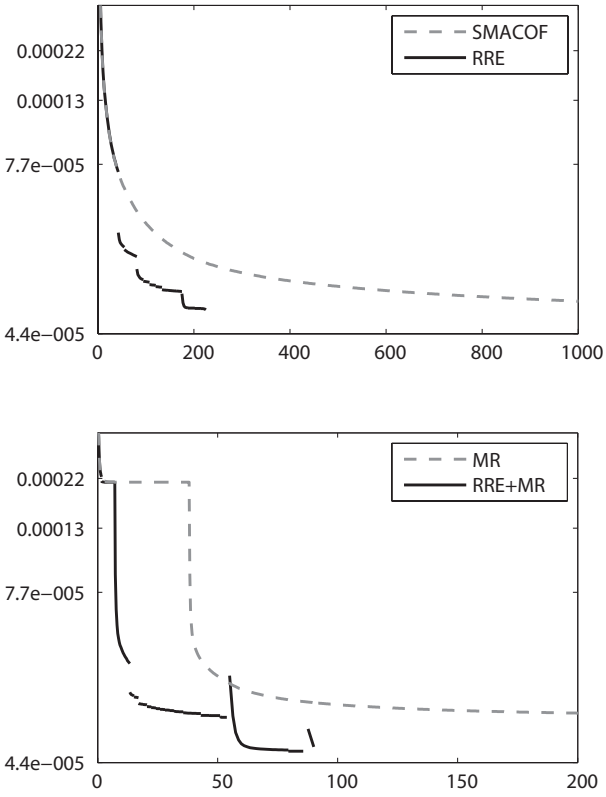


Fig. 3 Convergence (in terms of stress value) of basic SMACOF (top, dashed gray), SMACOF with RRE (top, black), SMACOF with multiresolution acceleration (bottom, dashed gray), and SMACOF with both RRE and multiscale (bottom, black), as a function of approximate CPU time, in seconds. Convergence at each scale was stopped at the same relative change of the stress value.

5: Interpolate the solution to the next resolution level, $\mathbf{X}_{l-1}^{(0)} = P_l^{l-1}(\mathbf{X}_l^*)$
6: **end for**

We use a modification of the *farthest point sampling* (FPS) (Gonzalez 1985; Hochbaum and Shmoys 1985; Eldar et al 1997) strategy to construct the grids, in which we add more points from the boundaries, to allow correct interpolation of the fine grid using the coarse grid elements. We use linear interpolation with weights obtained by solving a least squares fitting problem, with a regularization term ensuring all available nearest neighbors are used.

The multiresolution scheme can be further accelerated by applying vector extrapolation methods at each resolution level. In our experiments we used the RRE method, giving us a three-fold speedup beyond a simple multiresolution scheme.

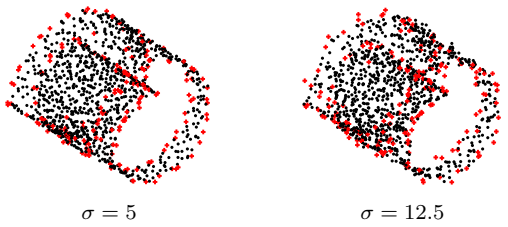


Fig. 4 Left to right: Swiss roll surface contaminated by Gaussian noise with $\sigma = 5$ and $\sigma = 12.5$. Detected boundary points are shown as red crosses.

4 Results

In order to assess the performance of the proposed approach, a few experiments were performed. In the first experiment, we used the Swiss roll surface with a large rectangular hole, sampled at 1200 points. Flattening was performed for points sampled on the manifold with additive i.i.d. Gaussian noise in each coordinate of each point. Two instances of the surface with different noise variances are shown in Figure 4.

We compare the proposed algorithm to Isomap, LLE, Laplacian eigenmaps, diffusion maps and Hessian LLE, in Figures 5 and 6[2]. Our algorithm finds a representation of the manifold with relatively small distortion. Adding i.i.d. Gaussian noise to the coordinates of the sampled points, our method remains accurate compared to other popular algorithms that exhibit large distortions. This can be seen, for example, for 1200 points, with $\sigma = 5, 12.5$, in Figure 6, where for comparison, $\mathrm{diam}(\mathcal{M}) \approx 2500$. The algorithm was allowed to converge until the relative change in the weighted stress was below some threshold. Tests with higher noise levels were also performed, with similar results. This includes noise levels at which the boundary detection quality deteriorated. Using multiscale further reduces the computational cost of the solution by a factor of two, for the problem shown in the example. We note that the speedup depends on both the manifold topology and metric, as well as the problem size. The reduction in computational effort was typical for all the problems we tested, up to 2550 points. Computation time, was about a few minutes, from two minutes for 1200 points, to about 6 on 2550 points, on an AMD Opteron$^{\mathrm{TM}}$ running at 2600 MHz.

In the next experiment we map a set of rendered images of the Stanford bunny. For this set of images we have a known natural parametrization, given by the set of locations of the cameras used to render the object. The images were compared using an L_2 metric. The mapping resulting from TCIE and Isomap, as well as the distribution of the camera centers, are shown in Figure 7. A hole was cut in the middle of the manifold of viewing angles, in order to test the effect of non-convexity on the resulting mappings. While the mapping is topologically equivalent to a subregion of the underlying parametrization space of Euler angle coordinates, the metric used depends on the albedo and the surface normal at each viewpoint and the projection shape, and as such, it is reasonable to expect only a local isometry transforming the Euler angles coordinates to the mapping obtained by our algorithm. We note, however, that the mapping obtained by TCIE does not inflate the hole as the one obtained by Isomap. Finally, we note that the algorithm performed well even when non-uniform and noisy sampling of the manifold resulted in false positive detection of boundary points, visible in Figure 7. An explanation for this behavior is that in the presence of of false positive boundary points, the algorithm simply relies on a more limited, yet non-local, support of distances for each point. For more example applications of the proposed algorithm, the reader is referred to the technical report (Rosman et al 2009).

[2] We used the same number of neighboring points (12) in all algorithms.

5 Conclusions

We presented a new framework for nonlinear dimensionality reduction, applicable to flat data manifolds with non-convex boundaries and non-trivial topology. We showed that by a careful selection of the geodesics we can robustly flatten non-convex manifolds. Since the proposed method uses global information it is less sensitive to noise than methods that use local distances between the data points, as confirmed in our experiments. The optimization scheme used by one approach benefits from vector extrapolation methods and multiresolution optimization.

In future work, we intend to generalize our results to non-Euclidean spaces. We would like to improve its computational efficiency using multigrid methods (Bronstein et al 2006b). Another issue we intend to explore is the robustness to changes in the sampling density of the manifold. Finally, we note that the main limitation of the proposed algorithm is its memory complexity and sensitivity to local minima, and we are currently investigating ways to overcome these limitations.

Appendix A

Proof of Proposition 1

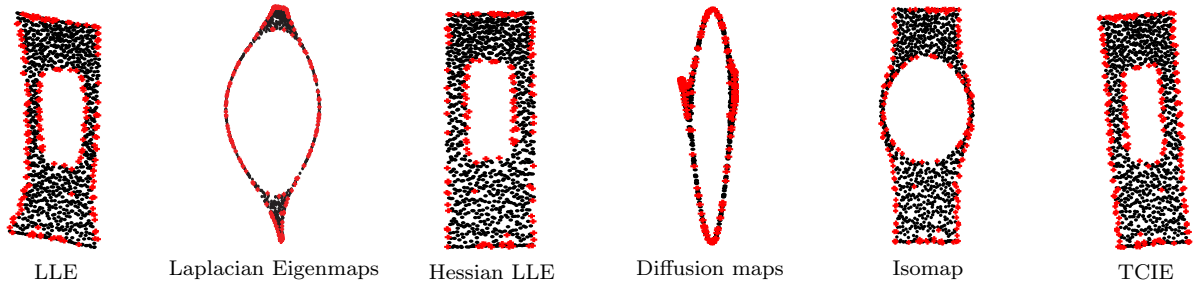


Fig. 5 Embedding of the Swiss roll contaminated by Gaussian noise with $\sigma = 5$, as produced by LLE, Laplacian eigenmaps, Hessian LLE, diffusion maps, Isomap, and our algorithm. Detected boundary points are shown as red crosses.

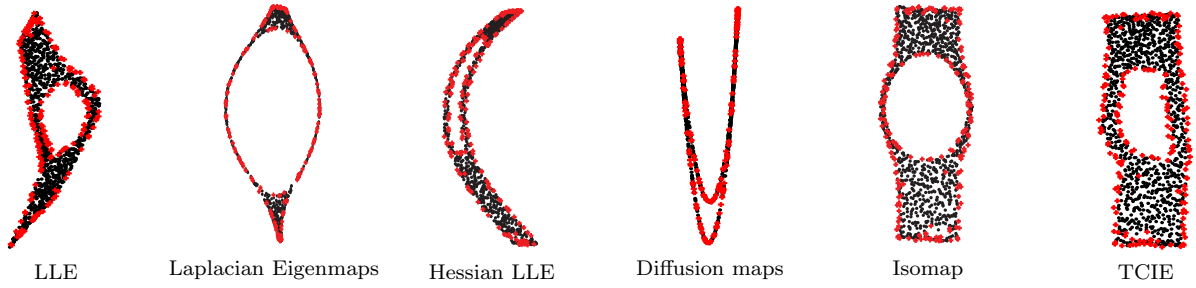


Fig. 6 Embedding of the Swiss roll contaminated by Gaussian noise with $\sigma = 12.5$, as produced by LLE, Laplacian eigenmaps, Hessian LLE, diffusion maps, Isomap, and our algorithm. Detected boundary points are shown as red crosses.

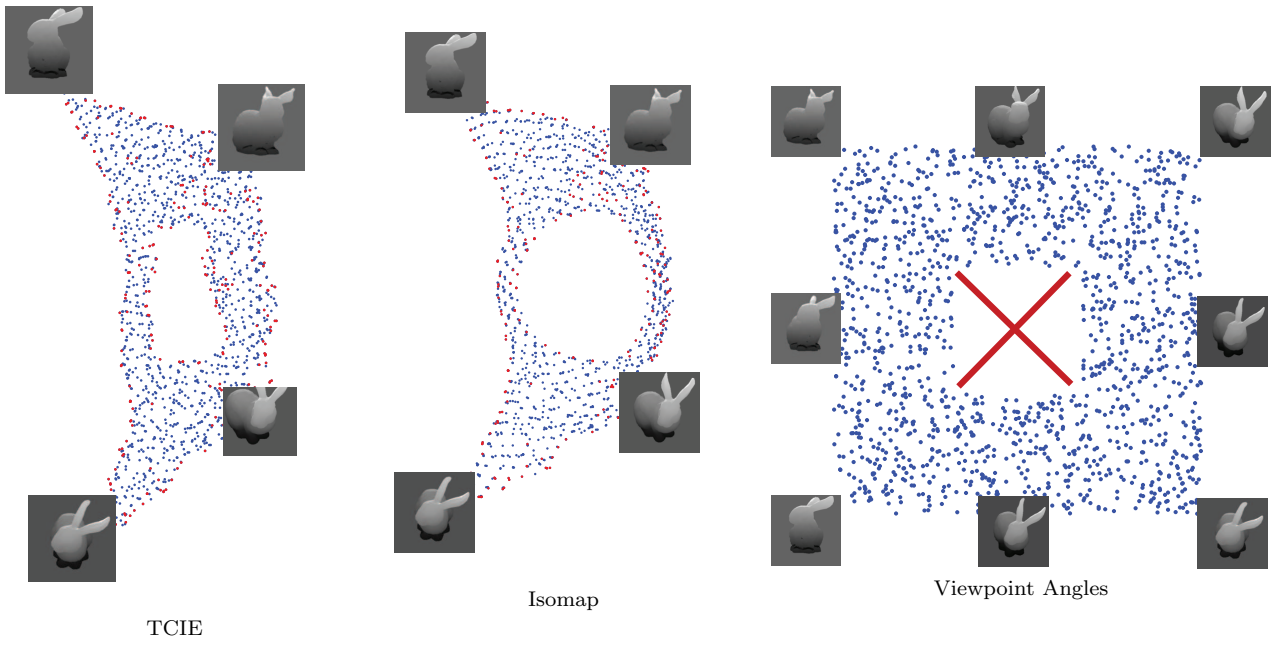


Fig. 7 A manifold mapped using L_2 distances between 2550 images rendered from various viewpoints around the Stanford bunny. Left to right, top to bottom: The mapping obtained by TCIE, and Isomap (using 12-nearest neighbors neighborhoods), as well as the set of the two Euler angles that parameterize the viewpoints, and a view of the positions of the camera centers with respect to the viewed object, marked as blue dots. The red "x" marks viewpoints intentionally removed from the image set. The normalized stress obtained by TCIE was 243.5, compared to 1052.9 obtained by Isomap.

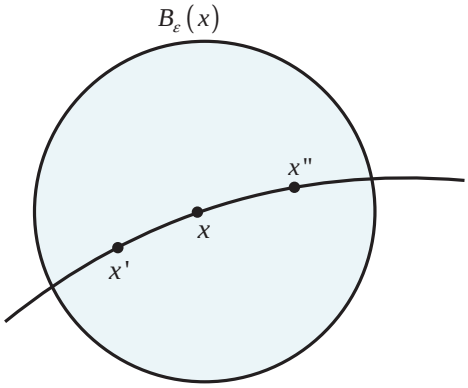


Fig. 8 An illustrations of points x, x', x'' used in the proof of Proposition 1

Let $(i,j) \in \bar{P}_1$. To prove the proposition, it is sufficient to show that the pair of points (i,j) is consistent, i.e., $(i,j) \in P$. Let $c_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j)$ be the geodesic connecting $\mathbf{z}_i$ and $\mathbf{z}_j$ in $\mathcal{M}$, and let $c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j)$ be its image under φ^{-1} in $\mathcal{C}$. Since $c_{\mathcal{M}}(\mathbf{z}_i, \mathbf{z}_j) \cap \partial\mathcal{M} = \emptyset$ and because of the isometry, $c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) \subset \text{int}(\mathcal{C})$, where $\text{int}(\mathcal{C})$ denotes the interior of $\mathcal{C}$.

Assume that (i,j) is inconsistent. This implies that $d_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) \neq d_{\mathbb{R}^m}(\mathbf{x}_i, \mathbf{x}_j)$. Because of the way the geodesic metric is defined, $d_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) > d_{\mathbb{R}^m}(\mathbf{x}_i, \mathbf{x}_j)$. The uniqueness of $c_{\mathbb{R}^m}(\mathbf{x}_i, \mathbf{x}_j)$ states that $c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j)$ cannot be a straight line (there exists a single straight line connecting each two points in Euclidean geometry). Therefore, there exits a point $\mathbf{x} \in c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j)$, in whose proximity $c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j)$ is not a straight line.

Since $c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) \subset \text{int}(\mathcal{C})$, and $\text{int}(\mathcal{C})$ is an open set, there exists a Euclidean ball $B_\epsilon(\mathbf{x})$ with the Euclidean metric $d_{\mathbb{R}^m}$ around $\mathbf{x}$ of radius $\epsilon > 0$. Let us take two points on the segment of the geodesic within the ball, $\mathbf{x}', \mathbf{x}'' \in c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) \cap B_\epsilon(\mathbf{x})$, as illustrated in Figure **?**. The geodesic $c_{\mathcal{C}}(\mathbf{x}', \mathbf{x}'')$ coincides with the segment of $c_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j)$ between $\mathbf{x}', \mathbf{x}''$. Yet, this segment is not a straight line, therefore we can shorten the geodesic by replacing this segment with $c_{\mathbb{R}^m}(\mathbf{x}', \mathbf{x}'')$, in contradiction to the fact that $c_{\mathcal{C}}(\mathbf{x}_1, \mathbf{x}_2)$ is a geodesic. Therefore, $(i,j) \in P$.

In the more general case, where $(\mathcal{M}, d_{\mathcal{M}})$ is not isometric to a subregion of a Euclidean space, the criterion defining $\bar{P}_2$ ensures that if the manifold is isometric to a subregion $\mathcal{C}$ of a space $\mathcal{C}'$ with Riemannian metric, we only select geodesics for which the geodesic metric and the metric of $\mathcal{C}'$ restricted to $\mathcal{C}$ identify. This is the case assumed by Proposition 2.

Proof of Proposition 2 Assume we have a pair of points for which the geodesic and the restricted metric on $\mathcal{C}$ are not the same, $(i,j) \notin P$. Clearly,

$$d_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) > d_{\mathcal{C}'}(\mathbf{x}_i, \mathbf{x}_j), \tag{2}$$

because of the way the geodesic metric is defined. On the other hand observe that the geodesic connecting the two points in $\mathcal{C}$ is not equal to the geodesic connecting them in $\mathcal{C}'$. Specifically, the geodesic in $\mathcal{C}'$ must cross the boundary of $\mathcal{C}$ (otherwise the distances would be equal). This results in the inequality

$$d_{\mathcal{C}'}(\mathbf{x}_i, \mathbf{x}_j) > d_{\mathcal{C}}(\mathbf{x}_i, \partial\mathcal{C}) + d_{\mathcal{C}}(\mathbf{x}_j, \partial\mathcal{C}).$$

Where the segments connecting $\mathbf{x}_i$ and $\mathbf{x}_i$ to $\partial\mathcal{C}$ are inside $\text{int}(\mathcal{C})$. Here we assume there is only one excursion outside of $\mathcal{C}$. Combining Inequality 2, we obtain

$$d_{\mathcal{C}}(\mathbf{x}_i, \mathbf{x}_j) > d_{\mathcal{C}}(\mathbf{x}_i, \partial\mathcal{C}) + d_{\mathcal{C}}(\mathbf{x}_j, \partial\mathcal{C}),$$

and therefore $(i,j) \notin \bar{P}_2$.

Acknowledgements We would like to thank Prof. Avram Sidi who shared with us his insights regarding vector extrapolation algorithms. We would also like to thank respective owners of the code of various nonlinear dimensionality reduction algorithms shown in this publication.

References

Aharon M, Kimmel R (2006) Representation analysis and synthesis of lip images using dimensionality reduction. International Journal of Computer Vision 67(3):297–312

Belkin M, Niyogi P (2002) Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Dietterich TG, Becker S, Ghahramani Z (eds) Advances in Neural Inf. Proc. Sys., MIT Press, Cambridge, MA, vol 14, pp 585–591

Belton D, Lichti DD (2007) Classification and segmentation of terrestrial laserscanner point clouds using local variance information. ISPRS Image Engineering and Vision Metrology 61(5):307–324

Ben-Tal A, Nemirovski AS (2001) Lectures on modern convex optimization: analysis, algorithms, and engineering applications. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA

Bernstein M, de Silva V, Langford JC, Tenenbaum JB (2001) Graph approximations to geodesics on embedded manifolds. Technical report, Stanford University

Blake A, Zisserman A (1987) Visual Reconstruction. The MIT Press, London, England

Borg I, Groenen P (1997) Modern multidimensional scaling: Theory and applications. Springer Verlag, New York

Boult TE, Kender JR (1986) Visual surface reconstruction using sparse depth data. In: Computer Vision and Pattern Recognition, vol 86, pp 68–76

Brand M (2005) Nonrigid embeddings for dimensionality reduction. In: Gama J, Camacho R, Brazdil P, Jorge A, Torgo L (eds) ECML, Springer, Lecture Notes in Computer Science, vol 3720, pp 47–59

Brandes U, Pich C (2007) Eigensolver methods for progressive multidimensional scaling of large data. In: Kaufmann M, Wagner D (eds) Graph Drawing, Karlsruhe, Germany, September 18-20, 2006, Springer, pp pp. 42–53

Bronstein AM, Bronstein MM, Kimmel R (2005) Three-dimensional face recognition. International Journal of Computer Vision 64(1):5–30

Bronstein AM, Bronstein MM, Kimmel R (2006a) Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. Proc Natl Acad Sci USA 103(5):1168–1172

Bronstein MM, Bronstein AM, Kimmel R, Yavneh I (2006b) Multigrid multidimensional scaling. Numerical Linear Algebra with Applications, Special issue on multigrid methods 13(2-3):149–171

Cabay S, Jackson L (1976) A polynomial extrapolation method for finding limits and antilimits of vector sequences. SIAM J Numer Anal 13:734–752

Chalmers M (1996) A linear iteration time layout algorithm for visualising high-dimensional data. In: IEEE Visualization, pp 127–132

Chazal F, Cohen-Steiner D, Mérigot Q (2007) Stability of boundary measures. Research Report 6219, INRIA Sophia Antipolis

Choi H, Choi S (2007) Robust kernel isomap. Pattern Recogn 40(3):853–862

Coifman RR, Lafon S, Lee AB, Maggioni M, Nadler B, Warner F, Zucker SW (2005) Geometric diffusions as a tool for harmonic analysis and structure definition of data. Proc Natl Acad Sci USA 102(21):7426–7431

Cormen TH, Leiserson CE, Rivest RL (1990) Introduction to Algorithms. MIT Press and McGraw-Hill

Diaz RM, Arencibia AQ (eds) (2003) Coloring of DT-MRI Fiber Traces using Laplacian Eigenmaps, Springer Verlag, Las Palmas de Gran Canaria, Spain

Dijkstra EW (1959) A note on two problems in connection with graphs. Numerische Mathematik 1:269–271

Duda RO, Hart PE, Stork DG (2000) Pattern Classification and Scene Analysis, 2nd edn. Wiley-Interscience

Eddy RP (1979) Extrapolationg to the limit of a vector sequence. In: Wang PC (ed) Information Linkage between Applied Mathematics and Industry, Academic Press, pp 387–396

Elad A, Kimmel R (2003) On bending invariant signatures for surfaces. IEEE Trans Pattern Anal Mach Intell 25(10):1285–1295

Eldar Y, Lindenbaum M, Porat M, Zeevi Y (1997) The farthest point strategy for progressive image sampling. IEEE Transactions on Image Processing 6(9):1305–1315

Freedman D (2002) Efficient simplicial reconstructions of manifolds from their samples. IEEE Trans Pattern Anal Mach Intell 24(10):1349–1357

Gonzalez TF (1985) Clustering to minimize the maximum intercluster distance. Theor Comput Sci 38:293–306

Gopi M (2002) On sampling and reconstructing surfaces with boundaries. In: Canadian Conference on Computational Geometry, pp 49–53

Grimes C, Donoho DL (2002) When does Isomap recover the natural parameterization of families of articulated images? Tech. Rep. 2002-27, Department of Statistics, Stanford University, Stanford, CA 94305-4065

Grimes C, Donoho DL (2003) Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. Proc Natl Acad Sci USA 100(10):5591–5596

Guttman L (1968) A general nonmetric technique for finding the smallest coordinate space for a configuration of points. Psychometrika 33:469–506

Guy G, Medioni G (1997) Inference of surfaces, 3D curves, and junctions from sparse, noisy, 3D data. IEEE Trans Pattern Anal Mach Intell 19(11):1265–1277

Haque A, Dilts GA (2007) Three-dimensional boundary detection for particle methods. J Comp Phys 226(2):1710–1730

Hochbaum D, Shmoys D (1985) A best possible approximation for the k-center problem. Mathematics of Operations Research 10(2):180–184

Kearsley A, Tapia R, Trosset MW (1998) The solution of the metric stress and sstress problems in multidimensional scaling using Newton's method. Computational Statistics 13(3):369–396

Keller Y, Lafon S, Krauthammer M (2005) Protein cluster analysis via directed diffusion. In: The fifth Georgia Tech International Conference on Bioinformatics

Kimmel R, Sethian JA (1998) Computing geodesic paths on manifolds. Proc Natl Acad Sci USA 95(15):8431–8435

de Leeuw J (1977) Applications of convex analysis to multidimensional scaling. In: Barra J, Brodeau F, Romier G, Custem BV (eds) Recent Developments in Statistics, North Holland Publishing Company, Amsterdam, pp 133–146

de Leeuw J (1984) Differentiability of Kruskal's stress at a local minimum. Psychometrika 49(1):111–113

de Leeuw J (1988) Convergence of the majorization method for multidimensional scaling. Journal of Classification 5(2):163–180

Mešina M (1977) Convergence acceleration for the iterative solution of the equations $X = AX + f$. Comput Methods Appl Mech Engrg 10:165–173

Mordohai P, Medioni G (2005) Unsupervised dimensionality estimation and manifold learning in high-dimensional spaces by tensor voting. In: Proceedings of International Joint Conference on Artificial Intelligence, pp 798–803

Platt JC (2004) Fastmap, metricmap, and landmark MDS are all nystrom algorithms. Tech. Rep. MSR-TR-2004-26, Microsoft Research (MSR)

Pless R (2003) Using Isomap to explore video sequences. In: International Conference on Computer Vision, Nice, France, pp 1433–1440

Rosman G, Bronstein AM, Bronstein MM, Kimmel R (2006) Topologically constrained isometric embedding. In: Proc. Conf. on Machine Learning and Pattern Recognition (MLPR)

Rosman G, Bronstein AM, Bronstein MM, Sidi A, Kimmel R (2008) Fast multidimensional scaling using vector extrapolation. Technical Report CIS-2008-01, Technnion, Israel Institute of Technology

Rosman G, Bronstein AM, Bronstein MM, Kimmel R (2009) Nonlinear dimensionality reduction by topologically constrained isometric embedding. Technical Report CIS-2009-05, Technion

Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290:2323–2326

Schwartz EL, Shaw A, Wolfson E (1989) A numerical solution to the generalized mapmaker's problem: Flattening nonconvex polyhedral surfaces. IEEE Trans Pattern Anal Mach Intell 11:1005–1008

Sidi A (1991) Efficient implementation of minimal polynomial and reduced rank extrapolation methods. J Comput Appl Math 36(3):305–337

Silva VD, Tenenbaum JB (2004) Sparse multidimensional scaling using landmark points. Tech. rep., Stanford University

Smith DA, Ford WF, Sidi A (1987) Extrapolation methods for vector sequences. SIAM Review 29(2):199–233

Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319–2323

Tong WS, Tang CK, Mordohai P, Medioni G (2004) First order augmentation to tensor voting for boundary inference and multiscale analysis in 3d. IEEE Trans Pattern Anal Mach Intell 26(5):594–611

Trosset M, Mathar R (1997) On the existence of nonglobal minimizers of the stress criterion for metric multidimensional scaling. American Statistical Association: Proceedings Statistical Computing Section pp 158–162

Weinberger KQ, Saul LK (2004) Unsupervised learning of image manifolds by semidefinite programming. In: Computer Vision and Pattern Recognition, Washington D.C., vol 2, pp 988–995

Weinberger KQ, Packer BD, Saul LK (2005) Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In: Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics, Barbados

Williams CKI (2002) On a connection between kernel pca and metric multidimensional scaling. Mach Learn 46(1-3):11–19, DOI http://dx.doi.org/10.1023/A:1012485807823

Zigelman G, Kimmel R, Kiryati N (2002) Texture mapping using surface flattening via multidimensional scaling. IEEE Transactions on Visualization and Computer Graphics 8(2):198–207