Tilburg centre for Creative Computing
Tilburg University
http://www.uvt.nl/ticc

P.O. Box 90153
5000 LE Tilburg, The Netherlands
Email: ticc@uvt.nl

October 26, 2009

TiCC TR 2009–005

Dimensionality Reduction: A Comparative Review

Laurens van der Maaten **Eric Postma**

Jaap van den Herik
TiCC, Tilburg University

Abstract

In recent years, a variety of nonlinear dimensionality reduction techniques have been proposed that aim to address the limitations of traditional techniques such as PCA and classical scaling. The paper presents a review and systematic comparison of these techniques. The performances of the nonlinear techniques are investigated on artificial and natural tasks. The results of the experiments reveal that nonlinear techniques perform well on selected artificial tasks, but that this strong performance does not necessarily extend to real-world tasks. The paper explains these results by identifying weaknesses of current nonlinear techniques, and suggests how the performance of nonlinear dimensionality reduction techniques may be improved.

Dimensionality Reduction: A Comparative Review

Laurens van der Maaten **Eric Postma** **Jaap van den Herik**

TiCC, Tilburg University

1 Introduction

Real-world data, such as speech signals, digital photographs, or fMRI scans, usually has a high dimensionality. In order to handle such real-world data adequately, its dimensionality needs to be reduced. Dimensionality reduction is the transformation of high-dimensional data into a meaningful representation of reduced dimensionality. Ideally, the reduced representation should have a dimensionality that corresponds to the intrinsic dimensionality of the data. The intrinsic dimensionality of data is the minimum number of parameters needed to account for the observed properties of the data [49]. Dimensionality reduction is important in many domains, since it mitigates the curse of dimensionality and other undesired properties of high-dimensional spaces [69]. As a result, dimensionality reduction facilitates, among others, classification, visualization, and compression of high-dimensional data. Traditionally, dimensionality reduction was performed using linear techniques such as Principal Components Analysis (PCA) [98], factor analysis [117], and classical scaling [126]. However, these linear techniques cannot adequately handle complex nonlinear data.

In the last decade, a large number of nonlinear techniques for dimensionality reduction have been proposed. See for an overview, e.g., [26, 110, 83, 131]. In contrast to the traditional linear techniques, the nonlinear techniques have the ability to deal with complex nonlinear data. In particular for real-world data, the nonlinear dimensionality reduction techniques may offer an advantage, because real-world data is likely to form a highly nonlinear manifold. Previous studies have shown that nonlinear techniques outperform their linear counterparts on complex artificial tasks. For instance, the Swiss roll dataset comprises a set of points that lie on a spiral-like two-dimensional manifold that is embedded within a three-dimensional space. A vast number of nonlinear techniques are perfectly able to find this embedding, whereas linear techniques fail to do so. In contrast to these successes on artificial datasets, successful applications of nonlinear dimensionality reduction techniques on natural datasets are less convincing. Beyond this observation, it is not clear to what extent the performances of the various dimensionality reduction techniques differ on artificial and natural tasks (a comparison is performed in [94], but this comparison is very limited in scope with respect to the number of techniques and tasks that are addressed).

Motivated by the lack of a systematic comparison of dimensionality reduction techniques, this paper presents a comparative study of the most important linear dimensionality reduction technique (PCA), and twelve frontranked nonlinear dimensionality reduction techniques. The aims of the paper are (1) to investigate to what extent novel nonlinear dimensionality reduction techniques outperform the traditional PCA on real-world datasets and (2) to identify the inherent weaknesses of the twelve nonlinear dimensionality reduction techniques. The investigation is performed by both a theoretical and an empirical evaluation of the dimensionality reduction techniques. The identification is performed by a careful analysis of the empirical results on specifically designed artificial datasets and on a selection of real-world datasets.

Next to PCA, the paper investigates the following twelve nonlinear techniques: (1) Kernel PCA, (2) Isomap, (3) Maximum Variance Unfolding, (4) diffusion maps, (5) Locally Linear Embedding, (6) Laplacian Eigenmaps, (7) Hessian LLE, (8) Local Tangent Space Analysis, (9) Sammon mapping, (10) multilayer autoencoders, (11) Locally Linear Coordination, and (12) manifold charting. Although our comparative review includes the most important nonlinear techniques for dimensionality reduction, it is not exhaustive. In the appendix, we list other important (nonlinear) dimensionality reduction techniques that are not included in our comparative review. There, we briefly explain why these techniques are not included.

The outline of the remainder of this paper is as follows. In Section 2, we give a formal definition of dimensionality reduction and subdivide the thirteen dimensionality reduction techniques into nine convex techniques and four non-convex techniques. Section 3 presents and discusses the nine convex dimensionality reduction techniques. Subsequently, Section 4 describes and discusses the four non-convex techniques for dimensionality reduction. Section 5 lists all techniques by theoretical characteristics. Then, in Section 6, we present an empirical comparison of all described techniques for dimensionality reduction on five artificial datasets and five natural datasets. Section 7 discusses the results of the experiments; moreover, it identifies weaknesses and points of improvement of the selected nonlinear techniques. Section 8 provides our conclusions.

2 Dimensionality Reduction

The problem of (nonlinear) dimensionality reduction can be defined as follows. Assume we have a dataset represented in a $n \times D$ matrix $\mathbf{X}$ consisting of n datavectors $\mathbf{x}_i$ ($i \in \{1, 2, \ldots, n\}$) with dimensionality D. Assume further that this dataset has intrinsic dimensionality d (where $d < D$, and often $d \ll D$). Here, in mathematical terms, intrinsic dimensionality means that the points in dataset $\mathbf{X}$ are lying on or near a manifold with dimensionality d that is embedded in the D-dimensional space. Note that we make no assumptions on the structure of this manifold: the manifold may be non-Riemannian because of discontinuities (i.e., the manifold may consist of a number of disconnected submanifolds). Dimensionality reduction techniques transform dataset $\mathbf{X}$ with dimensionality D into a new dataset $\mathbf{Y}$ with dimensionality d, while retaining the geometry of the data as much as possible. In general, neither the geometry of the data manifold, nor the intrinsic dimensionality d of the dataset $\mathbf{X}$ are known. Therefore, dimensionality reduction is an ill-posed problem that can only be solved by assuming certain properties of the data (such as its intrinsic dimensionality). Throughout the paper, we denote a high-dimensional datapoint by $\mathbf{x}_i$, where $\mathbf{x}_i$ is the ith row of the D-dimensional data matrix $\mathbf{X}$. The low-dimensional counterpart of $\mathbf{x}_i$ is denoted by $\mathbf{y}_i$, where $\mathbf{y}_i$ is the ith row of the d-dimensional data matrix $\mathbf{Y}$. In the remainder of the paper, we adopt the notation presented above, and we assume the dataset $\mathbf{X}$ is zero-mean.

Figure 1 shows a taxonomy of techniques for dimensionality reduction. We subdivide techniques for dimensionality reduction into convex and non-convex techniques. Convex techniques optimize an objective function that does not contain any local optima, whereas non-convex techniques optimize objective functions that do contain local optima. The further subdivisions in the taxonomy are discussed in the review in the following two sections.

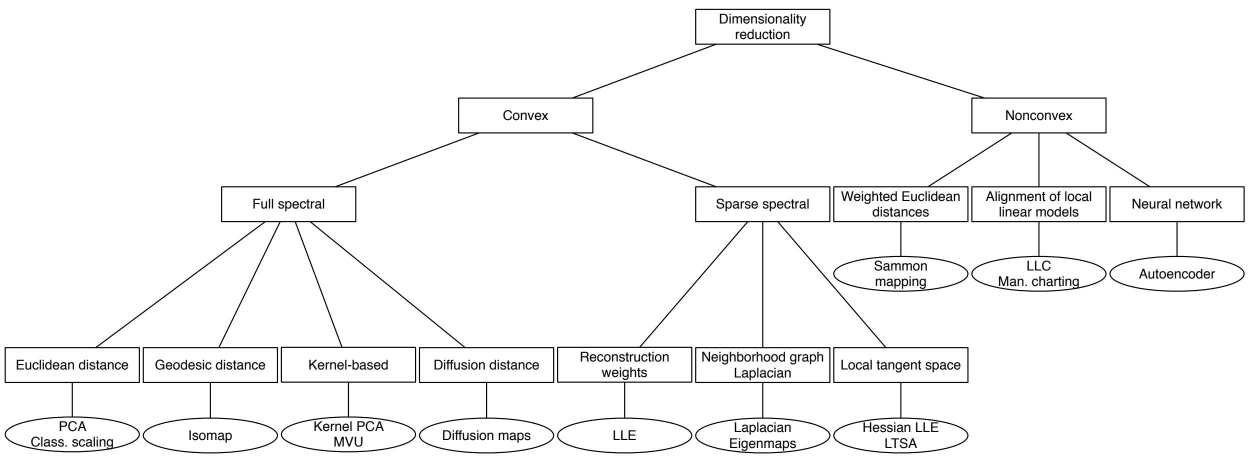


Figure 1: Taxonomy of dimensionality reduction techniques.

3 Convex Techniques for Dimensionality Reduction

Convex techniques for dimensionality reduction optimize an objective function that does not contain any local optima, i.e., the solution space is convex [22]. Most of the selected dimensionality reduction techniques fall in the class of convex techniques. In these techniques, the objective function usually has the form of a (generalized) Rayleigh quotient: the objective function is of the form $\phi(\mathbf{Y}) = \frac{\mathbf{Y}^T\mathbf{A}\mathbf{Y}}{\mathbf{Y}^T\mathbf{B}\mathbf{Y}}$. It is well known that a function of this form can be optimized by solving a generalized eigenproblem. One technique (Maximum Variance Unfolding) solves an additional semidefinite program using an interior point method. We subdivide convex dimensionality reduction techniques into techniques that perform an eigendecomposition of a full matrix (subsection 3.1) and those that perform an eigendecomposition of a sparse matrix (subsection 3.2).

3.1 Full Spectral Techniques

Full spectral techniques for dimensionality reduction perform an eigendecomposition of a full matrix that captures the covariances between dimensions or the pairwise similarities between datapoints (possibly in a feature space that is constructed by means of a kernel function). In this subsection, we discuss five such techniques: (1) PCA / classical scaling, (2) Isomap, (3) Kernel PCA, (4) Maximum Variance Unfolding, and (5) diffusion maps. The techniques are discussed separately in subsections 3.1.1 to 3.1.5.

3.1.1 PCA / Classical Scaling

Principal Components Analysis (PCA) [98, 65] is a linear technique for dimensionality reduction, which means that it performs dimensionality reduction by embedding the data into a linear subspace of lower dimensionality. Although there exist various techniques to do so, PCA is by far the most popular (unsupervised) linear technique. Therefore, in our comparison, we only include PCA.

PCA constructs a low-dimensional representation of the data that describes as much of the variance in the data as possible. This is done by finding a linear basis of reduced dimensionality for the data, in which the amount of variance in the data is maximal.

In mathematical terms, PCA attempts to find a linear mapping $\mathbf{M}$ that maximizes the cost function $\mathrm{trace}\left(\mathbf{M}^T \mathrm{cov}(\mathbf{X})\mathbf{M}\right)$, where $\mathrm{cov}(\mathbf{X})$ is the sample covariance matrix of the data $\mathbf{X}$. It can be shown that this linear mapping is formed by the d principal eigenvectors (i.e., principal components) of the sample covariance matrix of the zero-mean data[1]. Hence, PCA solves the eigenproblem

$$\mathrm{cov}(\mathbf{X})\mathbf{M} = \lambda\mathbf{M}. \tag{1}$$

The eigenproblem is solved for the d principal eigenvalues λ. The low-dimensional data representations $\mathbf{y}_i$ of the datapoints $\mathbf{x}_i$ are computed by mapping them onto the linear basis $\mathbf{M}$, i.e., $\mathbf{Y} = \mathbf{X}\mathbf{M}$.

PCA is identical to the traditional technique for multidimensional scaling called classical scaling [126]. The input into classical scaling is, like the input into most other multidimensional scaling techniques, a pairwise Euclidean distance matrix $\mathbf{D}$ whose entries d_{ij} represent the Euclidean distance between the high-dimensional datapoints $\mathbf{x}_i$ and $\mathbf{x}_j$. Classical scaling finds the linear mapping $\mathbf{M}$ that minimizes[2] the cost function

$$\phi(\mathbf{Y}) = \sum_{ij} \left(d_{ij}^2 - \|\mathbf{y}_i - \mathbf{y}_j\|^2\right), \tag{2}$$

in which $\|\mathbf{y}_i - \mathbf{y}_j\|^2$ is the squared Euclidean distance between the low-dimensional datapoints $\mathbf{y}_i$ and $\mathbf{y}_j$, $\mathbf{y}_i$ is restricted to be $\mathbf{x}_i\mathbf{M}$, and $\|\mathbf{m}_j\|^2 = 1$ for $\forall j$. It can be shown [126, 143] that the minimum

[1]PCA maximizes $\mathbf{M}^T \mathrm{cov}(\mathbf{X})\mathbf{M}$ with respect to $\mathbf{M}$, under the constraint that the L2-norm of each column $\mathbf{m}_j$ of $\mathbf{M}$ is 1, i.e., that $\|\mathbf{m}_j\|^2 = 1$. This constraint can be enforced by introducing a Lagrange multiplier λ. Hence, an unconstrained maximization of $\mathbf{m}_j^T \mathrm{cov}(\mathbf{X})\mathbf{m}_j + \lambda(1 - \mathbf{m}_j^T\mathbf{m}_j)$ is performed. The stationary points of this quantity are to be found when $\mathrm{cov}(\mathbf{X})\mathbf{m}_j = \lambda\mathbf{m}_j$.

[2]Note that the pairwise distances between the high-dimensional datapoints are constants in this cost function, and thus serve no purpose in the optimization. The optimization thus amounts to maximizing $\sum_{ij} \|\mathbf{M}\mathbf{x}_i - \mathbf{M}\mathbf{x}_j\|^2$, as a result of which it maximizes variance in the low-dimensional space (hence, it is identical to PCA).

of this cost function is given by the eigendecomposition of the Gram matrix $\mathbf{K} = \mathbf{X}\mathbf{X}^T$ of the high-dimensional data. The entries of the Gram matrix can be obtained by double-centering the pairwise squared Euclidean distance matrix, i.e., by computing

$$k_{ij} = -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_l d_{il}^2 - \frac{1}{n}\sum_l d_{jl}^2 + \frac{1}{n^2}\sum_{lm} d_{lm}^2\right). \qquad (3)$$

The minimum of the cost function in Equation 2 can now be obtained by multiplying the principal eigenvectors of the double-centered squared Euclidean distance matrix (i.e., the principal eigenvectors of the Gram matrix) with the square-root of their corresponding eigenvalues. The similarity of classical scaling to PCA is due to a relation between the eigenvectors of the covariance matrix and the Gram matrix of the high-dimensional data: it can be shown that the eigenvectors $\mathbf{u}_i$ and $\mathbf{v}_i$ of the matrices $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}\mathbf{X}^T$ are related through $\sqrt{\lambda_i}\mathbf{v}_i = \mathbf{X}\mathbf{u}_i$ [29]. The connection between PCA and classical scaling is described in more detail in, e.g., [143, 99].

PCA may also be viewed upon as a latent variable model called probabilistic PCA [103]. This model uses a Gaussian prior over the latent space, and a linear-Gaussian noise model. The probabilistic formulation of PCA leads to an EM-algorithm that may be computationally more efficient for very high-dimensional data. By using Gaussian processes, probabilistic PCA may also be extended to learn nonlinear mappings between the high-dimensional and the low-dimensional space [80]. Another extension of PCA also includes minor components (i.e., the eigenvectors corresponding to the smallest eigenvalues) in the linear mapping, as minor components may be of relevance in classification settings [141].

PCA and classical scaling have been successfully applied in a large number of domains such as face recognition [127], coin classification [66], and seismic series analysis [100]. PCA and classical scaling suffer from two main drawbacks.

First, in PCA, the size of the covariance matrix is proportional to the dimensionality of the datapoints. As a result, the computation of the eigenvectors might be infeasible for very high-dimensional data. In datasets in which $n < D$, this drawback may be overcome by performing classical scaling instead of PCA, because the classical scaling scales with the number of datapoints instead of with the number of dimensions in the data. Alternatively, iterative techniques such as Simple PCA [96] or probabilistic PCA [103] may be employed.

Second, the cost function in Equation 2 reveals that PCA and classical scaling focus mainly on retaining large pairwise distances d_{ij}^2, instead of focusing on retaining the small pairwise distances, which is much more important.

3.1.2 Isomap

Classical scaling has proven to be successful in many applications, but it suffers from the fact that it mainly aims to retain pairwise Euclidean distances, and does not take into account the distribution of the neighboring datapoints. If the high-dimensional data lies on or near a curved manifold, such as in the Swiss roll dataset [122], classical scaling might consider two datapoints as near points, whereas their distance over the manifold is much larger than the typical interpoint distance. Isomap [122] is a technique that resolves this problem by attempting to preserve pairwise geodesic (or curvilinear) distances between datapoints. Geodesic distance is the distance between two points measured over the manifold. In Isomap [122], the geodesic distances between the datapoints $\mathbf{x}_i$ $(i = 1, 2, \ldots, n)$ are computed by constructing a neighborhood graph G, in which every datapoint $\mathbf{x}_i$ is connected with its k nearest neighbors $\mathbf{x}_{i_j}$ $(j = 1, 2, \ldots, k)$ in the dataset $\mathbf{X}$. The shortest path between two points in the graph forms an estimate of the geodesic distance between these two points, and can easily be computed using Dijkstra's or Floyd's shortest-path algorithm [41, 47]. The geodesic distances between all datapoints in $\mathbf{X}$ are computed, thereby forming a pairwise geodesic distance matrix. The low-dimensional representations $\mathbf{y}_i$ of the datapoints $\mathbf{x}_i$ in the low-dimensional space $\mathbf{Y}$ are computed by applying classical scaling (see 3.1.1) on the resulting pairwise geodesic distance matrix.

An important weakness of the Isomap algorithm is its topological instability [7]. Isomap may construct erroneous connections in the neighborhood graph G. Such short-circuiting [82] can severely impair the performance of Isomap. Several approaches have been proposed to overcome the problem of short-circuiting, e.g., by removing datapoints with large total flows in the shortest-path algorithm [31] or by removing nearest neighbors that violate local linearity of the neighborhood graph [111]. A second weakness is that Isomap may suffer from 'holes' in the manifold. This problem can be dealt with by tearing manifolds with holes [82]. A third weakness of Isomap is that it can fail if the manifold is non-convex [121]. Despite these three weaknesses, Isomap was successfully applied on tasks such as wood inspection [94], visualization of biomedical data [86], and head pose estimation [102].

3.1.3 Kernel PCA

Kernel PCA (KPCA) is the reformulation of traditional linear PCA in a high-dimensional space that is constructed using a kernel function [112]. In recent years, the reformulation of linear techniques using the 'kernel trick' has led to the proposal of successful techniques such as kernel ridge regression and Support Vector Machines [115]. Kernel PCA computes the principal eigenvectors of the kernel matrix, rather than those of the covariance matrix. The reformulation of PCA in kernel space is straightforward, since a kernel matrix is similar to the inproduct of the datapoints in the high-dimensional space that is constructed using the kernel function. The application of PCA in the kernel space provides Kernel PCA the property of constructing nonlinear mappings.

Kernel PCA computes the kernel matrix K of the datapoints $\mathbf{x}_i$. The entries in the kernel matrix are defined by

$$k_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j), \tag{4}$$

where κ is a kernel function [115], which may be any function that gives rise to a positive-semidefinite kernel $\mathbf{K}$. Subsequently, the kernel matrix $\mathbf{K}$ is double-centered using the following modification of the entries

$$k_{ij} = -\frac{1}{2}\left(k_{ij} - \frac{1}{n}\sum_l k_{il} - \frac{1}{n}\sum_l k_{jl} + \frac{1}{n^2}\sum_{lm} k_{lm}\right). \tag{5}$$

The centering operation corresponds to subtracting the mean of the features in traditional PCA: it subtracts the mean of the data in the feature space defined by the kernel function κ. As a result, the data in the features space defined by the kernel function is zero-mean. Subsequently, the principal d eigenvectors $\mathbf{v}_i$ of the centered kernel matrix are computed. The eigenvectors of the covariance matrix $\mathbf{a}_i$ (in the feature space constructed by κ) can now be computed, since they are related to the eigenvectors of the kernel matrix $\mathbf{v}_i$ (see, e.g., [29]) through

$$\mathbf{a}_i = \frac{1}{\sqrt{\lambda_i}}\mathbf{v}_i. \tag{6}$$

In order to obtain the low-dimensional data representation, the data is projected onto the eigenvectors of the covariance matrix $\mathbf{a}_i$. The result of the projection (i.e., the low-dimensional data representation $\mathbf{Y}$) is given by

$$\mathbf{y}_i = \left\{\sum_{j=1}^{n} a_1^{(j)}\kappa(\mathbf{x}_j, \mathbf{x}_i), \ldots, \sum_{j=1}^{n} a_d^{(j)}\kappa(\mathbf{x}_j, \mathbf{x}_i)\right\}, \tag{7}$$

where $a_1^{(j)}$ indicates the jth value in the vector $\mathbf{a}_1$ and κ is the kernel function that was also used in the computation of the kernel matrix. Since Kernel PCA is a kernel-based method, the mapping performed by Kernel PCA relies on the choice of the kernel function κ. Possible choices for the kernel function include the linear kernel (making Kernel PCA equal to traditional PCA), the polynomial kernel, and the Gaussian kernel that is given in Equation 8 [115]. Notice that when the linear kernel is employed, the kernel matrix K is equal to the Gram matrix, and the procedure described above is identical to classical scaling (see 3.1.1).

An important weakness of Kernel PCA is that the size of the kernel matrix is proportional to the square of the number of instances in the dataset. An approach to resolve this weakness is proposed in [124]. Also, Kernel PCA mainly focuses on retaining large pairwise distances (even though these are now measured in feature space). Kernel PCA has been successfully applied to, e.g., face recognition [72], speech recognition [87], and novelty detection [64].

Like Kernel PCA, the Gaussian Process Latent Variable Model (GPLVM) also uses kernel functions to construct non-linear variants of (probabilistic) PCA [80]. However, the GPLVM is not simply the probabilistic counterpart of Kernel PCA: in the GPLVM, the kernel function is defined over the low-dimensional latent space, whereas in Kernel PCA, the kernel function is defined over the high-dimensional data space.

3.1.4 MVU

As described above, Kernel PCA allows for performing PCA in the feature space that is defined by the kernel function κ. Unfortunately, it is unclear how the kernel function κ should be selected. Maximum Variance Unfolding (MVU, formerly known as Semidefinite Embedding) is a technique that attempts to resolve this problem by *learning* the kernel matrix. MVU learns the kernel matrix by defining a neighborhood graph on the data (as in Isomap) and retaining pairwise distances in the resulting graph [138]. MVU is different from Isomap in that explicitly attempts to 'unfold' the data manifold. It does so by maximizing the Euclidean distances between the datapoints, under the constraint that the distances in the neighborhood graph are left unchanged (i.e., under the constraint that the local geometry of the data manifold is not distorted). The resulting optimization problem can be solved using semidefinite programming. MVU starts with the construction of a neighborhood graph G, in which each datapoint $\mathbf{x}_i$ is connected to its k nearest neighbors $\mathbf{x}_{i_j}$ $(j = 1, 2, \ldots, k)$. Subsequently, MVU attempts to maximize the sum of the squared Euclidean distances between all datapoints, under the constraint that the distances inside the neighborhood graph G are preserved. In other words, MVU performs the following optimization problem.

$$\begin{aligned}&\text{Maximize } \sum_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|^2 \text{ subject to (1), with:}\\ &(1)\ \|\mathbf{y}_i - \mathbf{y}_j\|^2 = \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for } \forall (i,j) \in G\end{aligned}$$

MVU reformulates the optimization problem as a semidefinite programming problem (SDP) [130] by defining the kernel matrix $\mathbf{K}$ as the outer product of the low-dimensional data representation $\mathbf{Y}$. The optimization problem then reduces to the following SDP, which learns the kernel matrix $\mathbf{K}$.

$$\begin{aligned}&\text{Maximize } \operatorname{trace}(\mathbf{K}) \text{ subject to (1), (2), and (3), with:}\\ &(1)\ k_{ii} + k_{jj} - 2k_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2 \text{ for } \forall (i,j) \in G\\ &(2)\ \sum_{ij} k_{ij} = 0\\ &(3)\ \mathbf{K} \succeq 0\end{aligned}$$

The solution $\mathbf{K}$ of the SDP is the kernel matrix that is used as input for Kernel PCA. The low-dimensional data representation $\mathbf{Y}$ is obtained by performing an eigendecomposition of the kernel matrix $\mathbf{K}$ that was constructed by solving the SDP.

MVU has a weakness similar to Isomap: short-circuiting may impair the performance of MVU, because it adds constraints to the optimization problem that prevent successful unfolding of the manifold. Despite this weakness, MVU was successfully applied to, e.g., sensor localization [139] and DNA microarray data analysis [71].

3.1.5 Diffusion Maps

The diffusion maps (DM) framework [76, 91] originates from the field of dynamical systems. Diffusion maps are based on defining a Markov random walk on the graph of the data. By performing the random walk for a number of timesteps, a measure for the proximity of the datapoints is obtained. Using this measure, the so-called diffusion distance is defined. In the low-dimensional representation of the data, the pairwise diffusion distances are retained as good as possible. The key idea behind the diffusion distance is that it is based on integrating over all paths through the graph. This makes the diffusion distance more robust to short-circuiting than, e.g., the geodesic distance that is employed in Isomap.

In the diffusion maps framework, a graph of the data is constructed first. The weights of the edges in the graph are computed using the Gaussian kernel function, leading to a matrix $\mathbf{W}$ with entries

$$w_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}}, \tag{8}$$

where σ indicates the variance of the Gaussian. Subsequently, normalization of the matrix W is performed in such a way that its rows add up to 1. In this way, a matrix $\mathbf{P}^{(1)}$ is formed with entries

$$p_{ij}^{(1)} = \frac{w_{ij}}{\sum_k w_{ik}}. \tag{9}$$

Since diffusion maps originate from dynamical systems theory, the resulting matrix $\mathbf{P}^{(1)}$ is considered a Markov matrix that defines the forward transition probability matrix of a dynamical process. Hence, the matrix $\mathbf{P}^{(1)}$ represents the probability of a transition from one datapoint to another datapoint in a single timestep. The forward probability matrix for t timesteps $\mathbf{P}^{(t)}$ is thus given by $\left(\mathbf{P}^{(1)}\right)^t$. Using the random walk forward probabilities $p_{ij}^{(t)}$, the diffusion distance is defined by

$$D^{(t)}(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_k \frac{\left(p_{ik}^{(t)} - p_{jk}^{(t)}\right)^2}{\psi(\mathbf{x}_k)^{(0)}}}. \tag{10}$$

In the equation, $\psi(\mathbf{x}_i)^{(0)}$ is a term that attributes more weight to parts of the graph with high density. It is defined by $\psi(\mathbf{x}_i)^{(0)} = \frac{m_i}{\sum_j m_j}$, where m_i is the degree of node $\mathbf{x}_i$ defined by $m_i = \sum_j p_{ij}$. From Equation 10, it can be observed that pairs of datapoints with a high forward transition probability have a small diffusion distance. Since the diffusion distance is based on integrating over all paths through the graph, it is more robust to short-circuiting than the geodesic distance that is employed in Isomap. In the low-dimensional representation of the data $\mathbf{Y}$, diffusion maps attempt to retain the diffusion distances. Using spectral theory on the random walk, it has been shown (see, e.g., [76]) that the low-dimensional representation $\mathbf{Y}$ that retains the diffusion distances $D^{(t)}(\mathbf{x}_i, \mathbf{x}_j)$ as good as possible (under a squared error criterion) is formed by the d nontrivial principal eigenvectors of the eigenproblem

$$\mathbf{P}^{(t)}\mathbf{v} = \lambda\mathbf{v}. \tag{11}$$

Because the graph is fully connected, the largest eigenvalue is trivial (viz. $\lambda_1 = 1$), and its eigenvector $\mathbf{v}_1$ is thus discarded. The low-dimensional representation $\mathbf{Y}$ is given by the next d principal eigenvectors. In the low-dimensional representation, the eigenvectors are normalized by their corresponding eigenvalues. Hence, the low-dimensional data representation is given by

$$\mathbf{Y} = \{\lambda_2\mathbf{v}_2, \lambda_3\mathbf{v}_3, \ldots, \lambda_{d+1}\mathbf{v}_{d+1}\}. \tag{12}$$

Diffusion maps have been successfully applied to, e.g., shape matching [101] and gene expression analysis [145].

3.2 Sparse Spectral Techniques

In the previous subsection, we discussed five techniques that construct a low-dimensional representation of the high-dimensional data by performing an eigendecomposition of a full matrix. In contrast, the four techniques discussed in this subsection solve a sparse (generalized) eigenproblem. All presented sparse spectral techniques only focus on retaining local structure of the data. We discuss the sparse spectral dimensionality reduction techniques (1) LLE, (2) Laplacian Eigenmaps, (3) Hessian LLE, and (4) LTSA separately in subsection 3.2.1 to 3.2.4.

3.2.1 LLE

Local Linear Embedding (LLE) [105] is a technique that is similar to Isomap (and MVU) in that it constructs a graph representation of the datapoints. In contrast to Isomap, it attempts to preserve solely local properties of the data. As a result, LLE is less sensitive to short-circuiting than Isomap, because only a small number of local properties are affected if short-circuiting occurs. Furthermore, the preservation of local properties allows for successful embedding of non-convex manifolds. In LLE, the local properties of the data manifold are constructed by writing the high-dimensional datapoints as a linear combination of their nearest neighbors. In the low-dimensional representation of the data, LLE attempts to retain the reconstruction weights in the linear combinations as good as possible.

LLE describes the local properties of the manifold around a datapoint $\mathbf{x}_i$ by writing the datapoint as a linear combination $\mathbf{w}_i$ (the so-called reconstruction weights) of its k nearest neighbors $\mathbf{x}_{i_j}$. Hence, LLE fits a hyperplane through the datapoint $\mathbf{x}_i$ and its nearest neighbors, thereby assuming that the manifold is locally linear. The local linearity assumption implies that the reconstruction weights $\mathbf{w}_i$ of the datapoints $\mathbf{x}_i$ are invariant to translation, rotation, and rescaling. Because of the invariance to these transformations, any linear mapping of the hyperplane to a space of lower dimensionality preserves the reconstruction weights in the space of lower dimensionality. In other words, if the low-dimensional data representation preserves the local geometry of the manifold, the reconstruction weights $\mathbf{w}_i$ that reconstruct datapoint $\mathbf{x}_i$ from its neighbors in the high-dimensional data representation also reconstruct datapoint $\mathbf{y}_i$ from its neighbors in the low-dimensional data representation. As a consequence, finding the d-dimensional data representation $\mathbf{Y}$ amounts to minimizing the cost function

$$\phi(\mathbf{Y}) = \sum_i \|\mathbf{y}_i - \sum_{j=1}^{k} w_{ij}\mathbf{y}_{i_j}\|^2 \text{ subject to } \|\mathbf{y}^{(k)}\|^2 = 1 \text{ for } \forall k, \qquad (13)$$

where $\mathbf{y}^{(k)}$ represents the kth column of the solution matrix $\mathbf{Y}$. The constraint on the covariance of the columns of $\mathbf{Y}$ is required to exclude the trivial solution $\mathbf{Y} = \mathbf{0}$. Roweis and Saul [105] showed[3] that the coordinates of the low-dimensional representations $\mathbf{y}_i$ that minimize this cost function are found by computing the eigenvectors corresponding to the smallest d nonzero eigenvalues of the inproduct $(\mathbf{I} - \mathbf{W})^T(\mathbf{I} - \mathbf{W})$, where $\mathbf{W}$ is a sparse $n \times n$ matrix whose entries are set to 0 if i and j are not connected in the neighborhood graph, and equal to the corresponding reconstruction weight otherwise. In this formula, $\mathbf{I}$ is the $n \times n$ identity matrix.

The popularity of LLE has led to the proposal of linear variants of the algorithm [58, 74], and to successful applications to, e.g., superresolution [27] and sound source localization [43]. However, there also exist experimental studies that report weak performance of LLE. In [86], LLE was reported to fail in the visualization of even simple synthetic biomedical datasets. In [68], it is claimed that LLE performs worse than Isomap in the derivation of perceptual-motor actions. A possible explanation lies in the difficulties that LLE has when confronted with manifolds that contain holes [105]. In addition, LLE tends to collapse large portions of the data very close together in the low-dimensional space, because the covariance constraint on the solution is too simple [129]. Also, the covariance constraint may give rise to undesired rescalings of the data manifold in the embedding [52].

3.2.2 Laplacian Eigenmaps

Similar to LLE, Laplacian Eigenmaps find a low-dimensional data representation by preserving local properties of the manifold [10]. In Laplacian Eigenmaps, the local properties are based on the pairwise distances between near neighbors. Laplacian Eigenmaps compute a low-dimensional representation of the data in which the distances between a datapoint and its k nearest neighbors are minimized. This is done in a weighted manner, i.e., the distance in the low-dimensional data representation between a datapoint and its first nearest neighbor contributes more to the cost function than the distance between the datapoint and its second nearest neighbor. Using spectral graph theory, the minimization of the cost function is defined as an eigenproblem.

[3] $\phi(\mathbf{Y}) = (\mathbf{Y} - \mathbf{WY})^2 = \mathbf{Y}^T(\mathbf{I} - \mathbf{W})^T(\mathbf{I} - \mathbf{W})\mathbf{Y}$ is the function that has to be minimized. Hence, the eigenvectors of $(\mathbf{I} - \mathbf{W})^T(\mathbf{I} - \mathbf{W})$ corresponding to the smallest nonzero eigenvalues form the solution that minimizes $\phi(\mathbf{Y})$.

The Laplacian Eigenmaps algorithm first constructs a neighborhood graph G in which every datapoint $\mathbf{x}_i$ is connected to its k nearest neighbors. For all points $\mathbf{x}_i$ and $\mathbf{x}_j$ in graph G that are connected by an edge, the weight of the edge is computed using the Gaussian kernel function (see Equation 8), leading to a sparse adjacency matrix $\mathbf{W}$. In the computation of the low-dimensional representations $\mathbf{y}_i$, the cost function that is minimized is given by

$$\phi(\mathbf{Y}) = \sum_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|^2 w_{ij}. \tag{14}$$

In the cost function, large weights w_{ij} correspond to small distances between the high-dimensional datapoints $\mathbf{x}_i$ and $\mathbf{x}_j$. Hence, the difference between their low-dimensional representations $\mathbf{y}_i$ and $\mathbf{y}_j$ highly contributes to the cost function. As a consequence, nearby points in the high-dimensional space are put as close together as possible in the low-dimensional representation.

The computation of the degree matrix $\mathbf{M}$ and the graph Laplacian $\mathbf{L}$ of the graph $\mathbf{W}$ allows for formulating the minimization problem in Equation 14 as an eigenproblem [4]. The degree matrix $\mathbf{M}$ of $\mathbf{W}$ is a diagonal matrix, of which the entries are the row sums of $\mathbf{W}$ (i.e., $m_{ii} = \sum_j w_{ij}$). The graph Laplacian $\mathbf{L}$ is computed by $\mathbf{L} = \mathbf{M} - \mathbf{W}$. It can be shown that the following holds[4]

$$\phi(\mathbf{Y}) = \sum_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|^2 w_{ij} = 2\mathbf{Y}^T\mathbf{L}\mathbf{Y}. \tag{15}$$

Hence, minimizing $\phi(\mathbf{Y})$ is proportional to minimizing $\mathbf{Y}^T\mathbf{L}\mathbf{Y}$ subject to $\mathbf{Y}^T\mathbf{M}\mathbf{Y} = \mathbf{I}_n$, a covariance constraint that is similar to that of LLE. The low-dimensional data representation $\mathbf{Y}$ can thus be found by solving the generalized eigenvalue problem

$$\mathbf{L}\mathbf{v} = \lambda\mathbf{M}\mathbf{v} \tag{16}$$

for the d smallest nonzero eigenvalues. The d eigenvectors $\mathbf{v}_i$ corresponding to the smallest nonzero eigenvalues form the low-dimensional data representation $\mathbf{Y}$.

Laplacian Eigenmaps suffer from many of the same weaknesses as LLE, such as the presence of a trivial solution that is prevented from being selected by a covariance constraint that can easily be cheated on. Despite these weaknesses, Laplacian Eigenmaps (and its variants) have been successfully applied to, e.g., face recognition [58] and the analysis of fMRI data [25]. In addition, variants of Laplacian Eigenmaps may be applied to supervised or semi-supervised learning problems [33, 11]. A linear variant of Laplacian Eigenmaps is presented in [59]. In spectral clustering, clustering is performed based on the sign of the coordinates obtained from Laplacian Eigenmaps [93, 116, 140].

3.2.3 Hessian LLE

Hessian LLE (HLLE) [42] is a variant of LLE that minimizes the ‘curviness’ of the high-dimensional manifold when embedding it into a low-dimensional space, under the constraint that the low-dimensional data representation is locally isometric. This is done by an eigenanalysis of a matrix $\mathcal{H}$ that describes the curviness of the manifold around the datapoints. The curviness of the manifold is measured by means of the local Hessian at every datapoint. The local Hessian is represented in the local tangent space at the datapoint, in order to obtain a representation of the local Hessian that is invariant to differences in the positions of the datapoints. It can be shown[5] that the coordinates of the low-dimensional representation can be found by performing an eigenanalysis of an estimator $\mathcal{H}$ of the manifold Hessian.

Hessian LLE starts with identifying the k nearest neighbors for each datapoint $\mathbf{x}_i$ using Euclidean distance. In the neighborhood, local linearity of the manifold is assumed. Hence, a basis for the local tangent space at point $\mathbf{x}_i$ can be found by applying PCA on its k nearest neighbors $\mathbf{x}_{i_j}$. In other words, for every datapoint $\mathbf{x}_i$, a basis for the local tangent space at point $\mathbf{x}_i$ is determined by computing the d principal eigenvectors $\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_d\}$ of the covariance matrix $\mathrm{cov}(\mathbf{x}_{i\cdot})$. Note that the above

[4]Note that $\phi(\mathbf{Y}) = \sum_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|^2 w_{ij} = \sum_{ij} (\|\mathbf{y}_i\|^2 + \|\mathbf{y}_j\|^2 - 2\mathbf{y}_i\mathbf{y}_j^T) w_{ij} = \sum_i \|\mathbf{y}_i\|^2 m_{ii} + \sum_j \|\mathbf{y}_j\|^2 m_{jj} - 2\sum_{ij} \mathbf{y}_i\mathbf{y}_j^T w_{ij} = 2\mathbf{Y}^T\mathbf{M}\mathbf{Y} - 2\mathbf{Y}^T\mathbf{W}\mathbf{Y} = 2\mathbf{Y}^T\mathbf{L}\mathbf{Y}$

[5]The derivation can be found in [42].

requires that $k \geq d$. Subsequently, an estimator for the Hessian of the manifold at point $\mathbf{x}_i$ in local tangent space coordinates is computed. In order to do this, a matrix $\mathbf{Z}_i$ is formed that contains (in the columns) all cross products of $\mathbf{M}$ up to the dth order (including a column with ones). The matrix $\mathbf{Z}_i$ is orthonormalized by applying Gram-Schmidt orthonormalization [2]. The estimation of the tangent Hessian $\mathbf{H}_i$ is now given by the transpose of the last $\frac{d(d+1)}{2}$ columns of the matrix $\mathbf{Z}_i$. Using the Hessian estimators in local tangent coordinates, a matrix $\mathcal{H}$ is constructed with entries

$$\mathcal{H}_{lm} = \sum_i \sum_j \left((\mathbf{H}_i)_{jl} \times (\mathbf{H}_i)_{jm} \right). \tag{17}$$

The matrix $\mathcal{H}$ represents information on the curviness of the high-dimensional data manifold. An eigenanalysis of $\mathcal{H}$ is performed in order to find the low-dimensional data representation that minimizes the curviness of the manifold. The eigenvectors corresponding to the d smallest nonzero eigenvalues of $\mathcal{H}$ are selected and form the matrix $\mathbf{Y}$, which contains the low-dimensional representation of the data.

Hessian LLE shares many characteristics with Laplacian Eigenmaps: it simply replaces the manifold Laplacian by the manifold Hessian. As a result, Hessian LLE suffers from many of the same weaknesses as Laplacian Eigenmaps and LLE. A successful application of Hessian LLE to sensor localization has been presented by [97].

3.2.4 LTSA

Similar to Hessian LLE, Local Tangent Space Analysis (LTSA) is a technique that describes local properties of the high-dimensional data using the local tangent space of each datapoint [149]. LTSA is based on the observation that, if local linearity of the manifold is assumed, there exists a linear mapping from a high-dimensional datapoint to its local tangent space, and that there exists a linear mapping from the corresponding low-dimensional datapoint to the same local tangent space [149]. LTSA attempts to align these linear mappings in such a way, that they construct the local tangent space of the manifold from the low-dimensional representation. In other words, LTSA simultaneously searches for the coordinates of the low-dimensional data representations, and for the linear mappings of the low-dimensional datapoints to the local tangent space of the high-dimensional data.

Similar to Hessian LLE, LTSA starts with computing bases for the local tangent spaces at the datapoints $\mathbf{x}_i$. This is done by applying PCA on the k datapoints $\mathbf{x}_{i_j}$ that are neighbors of datapoint $\mathbf{x}_i$. This results in a mapping $\mathbf{M}_i$ from the neighborhood of $\mathbf{x}_i$ to the local tangent space Θ_i. A property of the local tangent space Θ_i is that there exists a linear mapping $\mathbf{L}_i$ from the local tangent space coordinates θ_{i_j} to the low-dimensional representations $\mathbf{y}_{i_j}$. Using this property of the local tangent space, LTSA performs the following minimization

$$\min_{\mathbf{Y}_i, \mathbf{L}_i} \sum_i \| \mathbf{Y}_i \mathbf{J}_k - \mathbf{L}_i \Theta_i \|^2, \tag{18}$$

where $\mathbf{J}_k$ is the centering matrix (i.e., the matrix that performs the transformation in Equation 5) of size k [115]. In [149], it is shown that the solution $\mathbf{Y}$ of the minimization is formed by the eigenvectors of an alignment matrix $\mathbf{B}$, that correspond to the d smallest nonzero eigenvalues of $\mathbf{B}$. The entries of the alignment matrix $\mathbf{B}$ are obtained by iterative summation (for all matrices $\mathbf{V}_i$ and starting from $b_{ij} = 0$ for $\forall ij$)

$$\mathbf{B}_{\mathcal{N}_i \mathcal{N}_i} = \mathbf{B}_{\mathcal{N}_{i-1} \mathcal{N}_{i-1}} + \mathbf{J}_k \left(\mathbf{I} - \mathbf{V}_i \mathbf{V}_i^T \right) \mathbf{J}_k, \tag{19}$$

where $\mathcal{N}_i$ represents the set of indices of the nearest neighbors of datapoint $\mathbf{x}_i$. Subsequently, the low-dimensional representation $\mathbf{Y}$ is obtained by computation of the eigenvectors corresponding to the d smallest nonzero eigenvectors of the symmetric matrix $\frac{1}{2}(\mathbf{B} + \mathbf{B}^T)$.

Like the other sparse spectral dimensionality reduction techniques, LTSA may be hampered by the presence of a trivial solution in the cost function. In [123], a successful application of LTSA to microarray data is reported. A linear variant of LTSA is proposed in [147].

4 Non-convex Techniques for Dimensionality Reduction

In the previous section, we discussed techniques that construct a low-dimensional data representation by optimizing a convex objective function by means of an eigendecomposition. In this section, we discuss four techniques that optimize a non-convex objective function. Specifically, we discuss a non-convex techniques for multidimensional scaling that forms an alternative to classical scaling called Sammon mapping (subsection 4.1), a technique based on training multilayer neural networks (subsection 4.2), and two techniques that construct a mixture of local linear models and perform a global alignment of these linear models (subsection 4.3 and 4.4).

4.1 Sammon Mapping

In Section 3.1.1, we discussed classical scaling, a convex technique for multidimensional scaling [126], and noted that the main weakness of this technique is that it mainly focuses on retaining large pairwise distances, and not on retaining the small pairwise distances, which are much more important to the geometry of the data. Several multidimensional scaling variants have been proposed that aim to address this weakness [3, 38, 81, 108, 62, 92, 129]. In this subsection, we discuss one such MDS variant called Sammon mapping [108].

Sammon mapping adapts the classical scaling cost function (see Equation 2) by weighting the contribution of each pair (i, j) to the cost function by the inverse of their pairwise distance in the high-dimensional space d_{ij}. In this way, the cost function assigns roughly equal weight to retaining each of the pairwise distances, and thus retains the local structure of the data better than classical scaling. Mathematically, the Sammon cost function is given by

$$\phi(\mathbf{Y}) = \frac{1}{\sum_{ij} d_{ij}} \sum_{i \neq j} \frac{(d_{ij} - \|\mathbf{y}_i - \mathbf{y}_j\|)^2}{d_{ij}}, \tag{20}$$

where d_{ij} represents the pairwise Euclidean distance between the high-dimensional datapoints $\mathbf{x}_i$ and $\mathbf{x}_j$, and the constant in front is added in order to simplify the gradient of the cost function. The minimization of the Sammon cost function is generally performed using a pseudo-Newton method [34]. Sammon mapping is mainly used for visualization purposes [88].

The main weakness of Sammon mapping is that it assigns a much higher weight to retaining a distance of, say, 10^{-5} than to retaining a distance of, say, 10^{-4}. Successful applications of Sammon mapping have been reported on, e.g., gene data [44] and and geospatial data [119].

4.2 Multilayer Autoencoders

Multilayer autoencoders are feed-forward neural networks with an odd number of hidden layers [39, 63] and shared weights between the top and bottom layers (although asymmetric network structures may be employed as well). The middle hidden layer has d nodes, and the input and the output layer have D nodes. An example of an autoencoder is shown schematically in Figure 2. The network is trained to minimize the mean squared error between the input and the output of the network (ideally, the input and the output are equal). Training the neural network on the datapoints $\mathbf{x}_i$ leads to a network in which the middle hidden layer gives a d-dimensional representation of the datapoints that preserves as much structure in tha dataset $\mathbf{X}$ as possible. The low-dimensional representations $\mathbf{y}_i$ can be obtained by extracting the node values in the middle hidden layer, when datapoint $\mathbf{x}_i$ is used as input. In order to allow the autoencoder to learn a nonlinear mapping between the high-dimensional and low-dimensional data representation, sigmoid activation functions are generally used (except in the middle layer, where a linear activation function is usually employed).

Multilayer autoencoders usually have a high number of connections. Therefore, backpropagation approaches converge slowly and are likely to get stuck in local minima. In [61], this drawback is overcome using a learning procedure that consists of three main stages. First, the recognition layers of the network (i.e., the layers from $\mathbf{X}$ to $\mathbf{Y}$) are trained one-by-one using Restricted Boltzmann Machines[6]

[6]As an alternative, it is possible to pretrain each layer using a small denoising autoencoder [134].

(RBMs). An RBM is a Markov Random Field with a bipartite graph structure of visible and hidden nodes. Typically, the nodes are binary stochastic random variables (i.e., they obey a Bernoulli distribution) but for continuous data the binary nodes may be replaced by mean-field logistic or exponential family nodes [142]. RBMs can be trained efficiently using an unsupervised learning procedure that minimizes the so-called contrastive divergence [60]. Second, the reconstruction layers of the network (i.e., the layers from $\mathbf{Y}$ to $\mathbf{X}'$) are formed by the inverse of the trained recognition layers. In other words, the autoencoder is unrolled. Third, the unrolled autoencoder is finetuned in a supervised manner using backpropagation. The three-stage training procedure overcomes the susceptibility to local minima of standard backpropagation approaches [78].

The main weakness of autoencoders is that their training may be tedious, although this weakness is (partially) addressed by recent advances in deep learning. Autoencoders have succesfully been applied to problems such as missing data imputation [1] and HIV analysis [18].

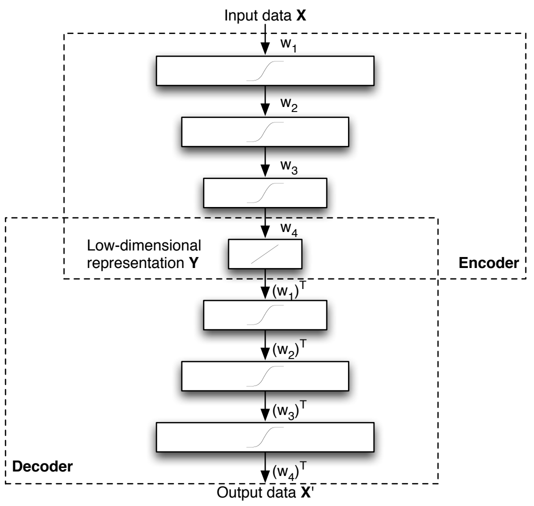


Figure 2: Schematic structure of an autoencoder.

4.3 LLC

Locally Linear Coordination (LLC) [120] computes a number of locally linear models and subsequently performs a global alignment of the linear models. This process consists of two steps: (1) computing a mixture of local linear models on the data by means of an EM-algorithm and (2) aligning the local linear models in order to obtain the low-dimensional data representation using a variant of LLE.

LLC first constructs a mixture of m factor analyzers (MoFA)[7] using the EM algorithm [40, 50, 70]. Alternatively, a mixture of probabilistic PCA model (MoPPCA) could be employed [125]. The local linear models in the mixture are used to construct m data representations $\mathbf{z}_{ij}$ and their corresponding responsibilities r_{ij} (where $j \in \{1, \ldots, m\}$) for every datapoint $\mathbf{x}_i$. The responsibilities r_{ij} describe to what extent datapoint $\mathbf{x}_i$ corresponds to the model j; they satisfy $\sum_j r_{ij} = 1$. Using the local models and the corresponding responsibilities, responsibility-weighted data representations $\mathbf{u}_{ij} = r_{ij}\mathbf{z}_{ij}$ are computed. The responsibility-weighted data representations $\mathbf{u}_{ij}$ are stored in a $n \times mD$ block matrix $\mathbf{U}$. The alignment of the local models is performed based on $\mathbf{U}$ and on a matrix $\mathbf{M}$ that is given by $\mathbf{M} = (\mathbf{I}_n - \mathbf{W})^T(\mathbf{I}_n - \mathbf{W})$. Herein, the matrix $\mathbf{W}$ contains the reconstruction weights computed by LLE (see 3.2.1), and $\mathbf{I}_n$ denotes the $n \times n$ identity matrix. LLC aligns the local models by solving the generalized eigenproblem

$$\mathbf{A}\mathbf{v} = \lambda \mathbf{B}\mathbf{v}, \tag{21}$$

for the d smallest nonzero eigenvalues[8]. In the equation, $\mathbf{A}$ is the inproduct of $\mathbf{M}^T\mathbf{U}$ and $\mathbf{B}$ is the inproduct of $\mathbf{U}$. The d eigenvectors $\mathbf{v}_i$ form a matrix $\mathbf{L}$ that defines a linear mapping from the responsibility-

[7]Note that the mixture of factor analyzers (and the mixture of probabilistic PCA models) is a mixture of Gaussians model with a restriction on the covariance of the Gaussians.

[8]The derivation of this eigenproblem can be found in [120].

weighted data representation $\mathbf{U}$ to the underlying low-dimensional data representation $\mathbf{Y}$. The low-dimensional data representation is thus obtained by computing $\mathbf{Y} = \mathbf{UL}$.

The main weakness of LLC is that the fitting of the mixture of factor analyzers is susceptible to the presence of local maxima in the log-likelihood function. LLC has been successfully applied to face images of a single person with variable pose and expression, and to handwritten digits [120].

4.4 Manifold Charting

Similar to LLC, manifold charting constructs a low-dimensional data representation by aligning a MoFA or a MoPPCA model [23]. In contrast to LLC, manifold charting does not minimize a cost function that corresponds to another dimensionality reduction technique (such as the LLE cost function). Manifold charting minimizes a convex cost function that measures the amount of disagreement between the linear models on the global coordinates of the datapoints. The minimization of this cost function can be performed by solving a generalized eigenproblem.

Manifold charting first performs the EM algorithm to learn a mixture of factor analyzers, in order to obtain m low-dimensional data representations $\mathbf{z}_{ij}$ and corresponding responsibilities r_{ij} (where $j \in \{1, \ldots, m\}$) for all datapoints $\mathbf{x}_i$. Manifold charting finds a linear mapping $\mathbf{M}$ from the data representations $\mathbf{z}_{ij}$ to the global coordinates $\mathbf{y}_i$ that minimizes the cost function

$$\phi(\mathbf{Y}) = \sum_{i=1}^{n} \sum_{j=1}^{m} r_{ij} \|\mathbf{y}_i - \mathbf{y}_{ij}\|^2, \tag{22}$$

where $\mathbf{y}_i = \sum_{k=1}^{m} r_{ik}\mathbf{y}_{ik}$, and $\mathbf{y}_{ij} = \mathbf{z}_{ij}\mathbf{M}$. The intuition behind the cost function is that whenever there are two linear models in which a datapoint has a high responsibility, these linear models should agree on the final coordinate of the datapoint. The cost function can be rewritten in the form

$$\phi(\mathbf{Y}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} r_{ij} r_{ik} \|\mathbf{y}_{ij} - \mathbf{y}_{ik}\|^2, \tag{23}$$

which allows the cost function to be rewritten in the form of a Rayleigh quotient. The Rayleigh quotient can be constructed by the definition of a block-diagonal matrix $\mathbf{D}$ with m blocks by

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \mathbf{D}_m \end{pmatrix}, \tag{24}$$

where $\mathbf{D}_j$ is the sum of the weighted covariances of the data representations $\mathbf{z}_{ij}$. Hence, $\mathbf{D}_j$ is given by

$$\mathbf{D}_j = \sum_{i=1}^{n} r_{ij} \operatorname{cov}(\begin{bmatrix}\mathbf{Z}_j & \mathbf{1}\end{bmatrix}). \tag{25}$$

In Equation 25, the 1-column is added to the data representation $\mathbf{Z}_j$ in order to facilitate translations in the construction of $\mathbf{y}_i$ from the data representations $\mathbf{z}_{ij}$. Using the definition of the matrix $\mathbf{D}$ and the $n \times mD$ block-diagonal matrix $\mathbf{U}$ with entries $\mathbf{u}_{ij} = r_{ij} \begin{bmatrix}\mathbf{z}_{ij} & \mathbf{1}\end{bmatrix}$, the manifold charting cost function can be rewritten as

$$\phi(\mathbf{Y}) = \mathbf{L}^T(\mathbf{D} - \mathbf{U}^T\mathbf{U})\mathbf{L}, \tag{26}$$

where $\mathbf{L}$ represents the linear mapping on the matrix $\mathbf{Z}$ that can be used to compute the final low-dimensional data representation $\mathbf{Y}$. The linear mapping $\mathbf{L}$ can thus be computed by solving the generalized eigenproblem

$$(\mathbf{D} - \mathbf{U}^T\mathbf{U})\mathbf{v} = \lambda \mathbf{U}^T\mathbf{U}\mathbf{v}, \tag{27}$$

for the d smallest nonzero eigenvalues. The d eigenvectors $\mathbf{v}_i$ form the columns of the linear combination $\mathbf{L}$ from $\begin{bmatrix}\mathbf{U} & \mathbf{1}\end{bmatrix}$ to $\mathbf{Y}$.

5 Characterization of the Techniques

In Section 3 and 4, we provided an overview of techniques for dimensionality reduction. This section lists the techniques by three theoretical characterizations. First, relations between the dimensionality reduction techniques are identified (subsection 5.1). Second, we list and discuss a number of general properties of the techniques such as the nature of the objective function that is optimized and the computational complexity of the technique (subsection 5.2). Third, the out-of-sample extension of the techniques is discussed (subsection 5.3).

5.1 Relations

Many of the techniques discussed in Section 3 and 4 are highly interrelated, and in certain special cases even equivalent. In the previous sections, we already mentioned some of these relations, but in this subsection, we discuss the relations between the techniques in more detail. Specifically, we discuss three types of relations between the techniques.

First, traditional PCA is identical to performing classical scaling and to performing Kernel PCA with a linear kernel, due to the relation between the eigenvectors of the covariance matrix and the double-centered squared Euclidean distance matrix [143], which is in turn equal to the Gram matrix. Autoencoders in which only linear activation functions are employed are very similar to PCA as well [75].

Second, performing classical scaling on a pairwise geodesic distance matrix is identical to performing Isomap. Similarly, performing Isomap with the number of nearest neighbors k set to $n-1$ is identical to performing classical scaling (and thus also to performing PCA and to performing Kernel PCA with a linear kernel). Diffusion maps are also very similar to classical scaling, however, they attempt to retain a different type of pairwise distances (the so-called diffusion distances). The main discerning property of diffusion maps its pairwise distance measure between the high-dimensional datapoints is based on integrating over all paths through the graph defined on the data.

Third, the spectral techniques Kernel PCA, Isomap, LLE, and Laplacian Eigenmaps can all be viewed upon as special cases of the more general problem of learning eigenfunctions [14, 57]. As a result, Isomap, LLE, and Laplacian Eigenmaps[9] can be considered as special cases of Kernel PCA that use a specific kernel function κ. For instance, this relation is visible in the out-of-sample extensions of Isomap, LLE, and Laplacian Eigenmaps [17]. The out-of-sample extension for these techniques is performed by means of a so-called Nyström approximation [6, 99], which is known to be equivalent to the Kernel PCA projection (see 5.3 for more details). Laplacian Eigenmaps and Hessian LLE are also intimately related: they only differ in the type of differential operator they define on the data manifold.

Diffusion maps in which $t = 1$ are fairly similar to Kernel PCA with the Gaussian kernel function. There are two main differences between the two: (1) no centering of the Gram matrix is performed in diffusion maps (although centering is not generally considered to be essential in Kernel PCA [115]) and (2) diffusion maps do not employ the principal eigenvector of the kernel matrix, whereas Kernel PCA does. MVU can also be viewed upon as a special case of Kernel PCA, in which the the solution of the SDP is the kernel function. In turn, Isomap can be viewed upon as a technique that finds an approximate solution to the MVU problem [144]. Evaluation of the dual MVU problem has also shown that LLE and Laplacian Eigenmaps show great resemblance to MVU [144].

As a consequence of the relations between the techniques, our empirical comparative evaluation in Section 6 does not include (1) classical scaling, (2) Kernel PCA using a linear kernel, and (3) autoencoders with linear activation functions, because they are similar to PCA. Furthermore, we do not evaluate Kernel PCA using a Gaussian kernel in the experiments, because of its resemblance to diffusion maps; instead we use a polynomial kernel.

5.2 General Properties

In Table 1, the thirteen dimensionality reduction techniques are listed by four general properties: (1) the parametric nature of the mapping between the high-dimensional and the low-dimensional space,

[9]The same also holds for Hessian LLE and LTSA, but up to our knowledge, the kernel functions for these techniques have never been derived.

Technique	*Parametric*	*Parameters*	*Computational*	*Memory*
PCA	yes	none	$O(D^3)$	$O(D^2)$
Class. scaling	no	none	$O(n^3)$	$O(n^2)$
Isomap	no	k	$O(n^3)$	$O(n^2)$
Kernel PCA	no	$\kappa(\cdot,\cdot)$	$O(n^3)$	$O(n^2)$
MVU	no	k	$O((nk)^3)$	$O((nk)^3)$
Diffusion maps	no	σ, t	$O(n^3)$	$O(n^2)$
LLE	no	k	$O(pn^2)$	$O(pn^2)$
Laplacian Eigenmaps	no	k, σ	$O(pn^2)$	$O(pn^2)$
Hessian LLE	no	k	$O(pn^2)$	$O(pn^2)$
LTSA	no	k	$O(pn^2)$	$O(pn^2)$
Sammon mapping	no	none	$O(in^2)$	$O(n^2)$
Autoencoders	yes	net size	$O(inw)$	$O(w)$
LLC	yes	m, k	$O(imd^3)$	$O(nmd)$
Manifold charting	yes	m	$O(imd^3)$	$O(nmd)$

Table 1: Properties of techniques for dimensionality reduction.

(2) the main free parameters that have to be optimized, (3) the computational complexity of the main computational part of the technique, and (4) the memory complexity of the technique. We discuss the four general properties below.

For property 1, Table 1 shows that most techniques for dimensionality reduction are non-parametric. This means that the technique does not specify a direct mapping from the high-dimensional to the low-dimensional space (or vice versa). The non-parametric nature of most techniques is a disadvantage for two main reasons: (1) it is not possible to generalize to held-out or new test data without performing the dimensionality reduction technique again and (2) it is not possible to obtain insight in how much information of the high-dimensional data was retained in the low-dimensional space by reconstructing the original data from the low-dimensional data representation and measuring the error between the reconstructed and true data.

For property 2, Table 1 shows that the objective functions of most nonlinear techniques for dimensionality reduction all have free parameters that need to be optimized. By free parameters, we mean parameters that directly influence the cost function that is optimized. The reader should note that non-convex techniques for dimensionality reduction have additional free parameters, such as the learning rate and the permitted maximum number of iterations. Moreover, LLE uses a regularization parameter in the computation of the reconstruction weights. The presence of free parameters has both advantages and disadvantages. The main advantage of the presence of free parameters is that they provide more flexibility to the technique, whereas their main disadvantage is that they need to be tuned to optimize the performance of the dimensionality reduction technique.
For properties 3 and 4, Table 1 provides insight into the computational and memory complexities of the computationally most expensive algorithmic components of the techniques. The computational complexity of a dimensionality reduction technique is of importance to its practical applicability. If the memory or computational resources needed are too large, application becomes infeasible. The computational complexity of a dimensionality reduction technique is determined by: (1) properties of the dataset such as the number of datapoints n and their dimensionality D, and (2) by parameters of the techniques, such as the target dimensionality d, the number of nearest neighbors k (for techniques based on neighborhood graphs) and the number of iterations i (for iterative techniques). In Table 1, p denotes the ratio of nonzero elements in a sparse matrix to the total number of elements, m indicates the number of local models in a mixture of factor analyzers, and w is the number of weights in a neural network. Below, we discuss the computational complexity and the memory complexity of each of the entries in the table.

The computationally most demanding part of PCA is the eigenanalysis of the $D \times D$ covariance matrix[10], which is performed using a power method in $O(D^3)$. The corresponding memory complexity of PCA is $O(D^2)$. In datasets in which $n < D$, the computational and memory complexity of PCA can be reduced to $O(n^3)$ and $O(n^2)$, respectively (see Section 3.1.1). Classical scaling, Isomap, diffusion maps, and Kernel PCA perform an eigenanalysis of an $n \times n$ matrix using a power method in $O(n^3)$. Because these full spectral techniques store a full $n \times n$ kernel matrix, the memory complexity of these techniques is $O(n^2)$.

In addition to the eigendecomposition of Kernel PCA, MVU solves a semidefinite program (SDP) with nk constraints. Both the computational and the memory complexity of solving an SDP are cube in the number of constraints [21]. Since there are nk constraints, the computational and memory complexity of the main part of MVU is $O((nk)^3)$. Training an autoencoder using RBM training or backpropagation has a computational complexity of $O(inw)$. The training of autoencoders may converge very slowly, especially in cases where the input and target dimensionality are very high (since this yields a high number of weights in the network). The memory complexity of autoencoders is $O(w)$.

The main computational part of LLC and manifold charting is the computation of the MoFA or MoP-PCA model, which has computational complexity $O(imd^3)$. The corresponding memory complexity is $O(nmd)$. Sammon mapping has a computational complexity of $O(in^2)$. The corresponding memory complexity is $O(n^2)$, although the memory complexity may be reduced by computing the pairwise distances on-the-fly.

Similar to, e.g., Kernel PCA, sparse spectral techniques perform an eigenanalysis of an $n \times n$ matrix. However, for these techniques the $n \times n$ matrix is sparse, which is beneficial, because it lowers the computational complexity of the eigenanalysis. Eigenanalysis of a sparse matrix (using Arnoldi methods [5] or Jacobi-Davidson methods [48]) has computational complexity $O(pn^2)$, where p is the ratio of nonzero elements in the sparse matrix to the total number of elements. The memory complexity is $O(pn^2)$ as well.

From the discussion of the four general properties of the techniques for dimensionality reduction above, we make four observations: (1) most nonlinear techniques for dimensionality reduction do not provide a parametric mapping between the high-dimensional and the low-dimensional space, (2) all nonlinear techniques require the optimization of one or more free parameters, (3) when $D < n$ (which is true in most cases), nonlinear techniques have computational disadvantages compared to PCA, and (4) a number of nonlinear techniques suffer from a memory complexity that is square or cube with the number of datapoints n. From these observations, it is clear that nonlinear techniques impose considerable demands on computational resources, as compared to PCA. Attempts to reduce the computational and/or memory complexities of nonlinear techniques have been proposed for, e.g., Isomap [37, 79], MVU [136, 139], and Kernel PCA [124].

5.3 Out-of-sample Extension

An important requirement for dimensionality reduction techniques is the ability to embed new high-dimensional datapoints into an existing low-dimensional data representation. So-called out-of-sample extensions have been developed for a number of techniques to allow for the embedding of such new datapoints, and can be subdivided into parametric and nonparametric out-of-sample extensions.

In a parametric out-of-sample extension, the dimensionality reduction technique provides all parameters that are necessary in order to transform new data from the high-dimensional to the low-dimensional space (see Table 1 for an overview of parametric dimensionality reduction techniques). In linear techniques such as PCA, this transformation is defined by the linear mapping $\mathbf{M}$ that was applied to the original data. For autoencoders, the trained network defines the transformation from the high-dimensional to the low-dimensional data representation.

For the other nonlinear dimensionality reduction techniques, a parametric out-of-sample extension is not available, and therefore, a nonparametric out-of-sample extension is required. Nonparametric out-of-sample extensions perform an estimation of the transformation from the high-dimensional to the low-dimensional space. For instance, the out-of-sample extension of Kernel PCA [112] employs the

[10]In cases in which $n \gg D$, the main computational part of PCA may be the computation of the covariance matrix. We ignore this for now.

so-called Nyström approximation [99], which approximates the eigenvectors of a large $n \times n$ matrix based on the eigendecomposition of an $m \times m$ submatrix of the large matrix (with $m < n$). A similar out-of-sample extension for Isomap, LLE, and Laplacian Eigenmaps has been presented in [17], in which the techniques are redefined in the Kernel PCA framework and the Nyström approximation is employed. Similar nonparametric out-of-sample extensions for Isomap are proposed in [31, 37]. For MVU, an approximate out-of-sample extension has been proposed that is based on computing a linear transformation from a set of landmark points to the complete dataset [136]. An alternative out-of-sample extension for MVU finds this linear transformation by computing the eigenvectors corresponding to the smallest eigenvalues of the graph Laplacian [139]. A third out-of-sample extension for MVU approximates the kernel eigenfunction using Gaussian basis functions [30].

A nonparametric out-of-sample extension that can be applied to all nonlinear dimensionality reduction techniques is proposed in [85]. The technique finds the nearest neighbor of the new datapoint in the high-dimensional representation, and computes the linear mapping from the nearest neighbor to its corresponding low-dimensional representation. The low-dimensional representation of the new datapoint is found by applying the same linear mapping to this datapoint.

From the description above, we may observe that linear and nonlinear techniques for dimensionality reduction are quite similar in that they allow the embedding of new datapoints. However, for a significant number of nonlinear techniques, only nonparametric out-of-sample extensions are available, which leads to estimation errors in the embedding of new datapoints.

6 Experiments

In this section, a systematic empirical comparison of the performance of the techniques for dimensionality reduction is performed. We perform the comparison by measuring generalization errors in classification tasks on two types of datasets: (1) artificial datasets and (2) natural datasets. In addition to generalization errors, we measure the 'trustworthiness' and 'continuity' of the low-dimensional embeddings as proposed in [132].

The setup of our experiments is described in subsection 6.1. In subsection 6.2, the results of our experiments on five artificial datasets are presented. Subsection 6.3 presents the results of the experiments on five natural datasets.

6.1 Experimental Setup

In our experiments on both the artificial and the natural datasets, we apply the thirteen techniques for dimensionality reduction on the high-dimensional representation of the data. Subsequently, we assess the quality of the resulting low-dimensional data representations by evaluating to what extent the local structure of the data is retained. The evaluation is performed in two ways: (1) by measuring the generalization errors of 1-nearest neighbor classifiers that are trained on the low-dimensional data representation (as is done, e.g., in [109]) and (2) by measuring the 'trustworthiness' and the 'continuity' of the low-dimensional embeddings [132]. The trustworthiness measures the proportion of points that are too close together in the low-dimensional space. The trustworthiness measure is defined as

$$T(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in U_i^{(k)}} \left(r(i,j) - k\right), \tag{28}$$

where $r(i,j)$ represents the rank of the low-dimensional datapoint j according to the pairwise distances between the low-dimensional datapoints. The variable $U_i^{(k)}$ indicates the set of points that are among the k nearest neighbors in the low-dimensional space but not in the high-dimensional space. The continuity measure is defined as

$$C(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in V_i^{(k)}} \left(\hat{r}(i,j) - k\right), \tag{29}$$

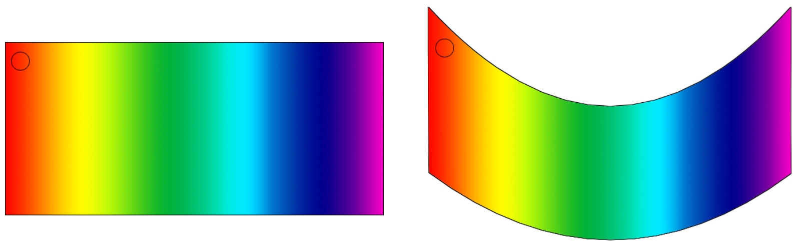

(a) True underlying manifold. (b) Reconstructed manifold up to a nonlinear warping.

Figure 3: Two low-dimensional data representations.

where $\hat{r}(i, j)$ represents the rank of the high-dimensional datapoint j according to the pairwise distances between the the high-dimensional datapoints. The variable $V_i^{(k)}$ indicates the set of points that are among the k nearest neighbors in the high-dimensional space but not in the low-dimensional space.

The generalization errors of the 1-nearest neighbor classifiers, the trustworthiness, and the continuity evaluate to what extent the local structure of the data is retained (the 1-nearest neighbor classifier does so because of its high variance). We opt for an evaluation of the local structure of the data, because for successful visualization or classification of data, its local structure needs to be retained. An evaluation of the quality based on generalization errors, trustworthiness, and continuity has an important advantage over measuring reconstruction errors, because a high reconstruction error does not necessarily imply that the dimensionality reduction technique performed poorly. For instance, if a dimensionality reduction technique recovers the true underlying manifold in Figure 3(a) up to a nonlinear warping, such as in Figure 3(b), this leads to a high reconstruction error, whereas the local structure of the two manifolds is nearly identical (as the circles indicate). Moreover, for real-world datasets the true underlying manifold of the data is usually unknown, as a result of which reconstruction errors cannot be computed for the natural datasets.

For all dimensionality reduction techniques except for Isomap, MVU, and sparse spectral techniques (the so-called manifold learners), we performed experiments without out-of-sample extension, because our main interest is in the performance of the dimensionality reduction techniques, and not in the quality of the out-of-sample extension. In the experiments with Isomap, MVU, and sparse spectral techniques, we employ out-of-sample extensions (see subsection 5.3) in order to embed datapoints that are not connected to the largest component of the neighborhood graph which is constructed by these techniques. The use of the out-of-sample extension of the manifold learners is necessary because the traditional implementations of Isomap, MVU, and sparse spectral techniques can only embed the datapoints that comprise the largest component of the neighborhood graph.

The parameter settings employed in our experiments are listed in Table 2. Most parameters were optimized using an exhaustive grid search within a reasonable range. The range of parameters for which we performed experiments is shown in Table 2. For one parameter (the bandwidth σ in diffusion maps and Laplacian Eigenmaps), we employed fixed values in order to restrict the computational requirements of our experiments. The value of k in the k-nearest neighbor classifiers was set to 1. We determined the target dimensionality in the experiments by means of the maximum likelihood intrinsic dimensionality estimator [84]. Note that for Hessian LLE and LTSA, the dimensionality of the actual low-dimensional data representation cannot be higher than the number of nearest neighbors that was used to construct the neighborhood graph. The generalization errors of the 1-nearest neighbor classifiers were measured using leave-one-out validation.

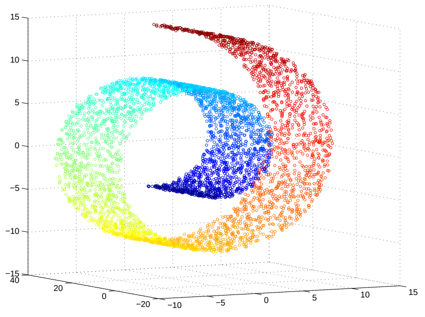
6.1.1 Five Artificial Datasets

We performed experiments on five artificial datasets. The datasets were specifically selected to investigate how the dimensionality reduction techniques deal with: (i) data that lies on a low-dimensional manifold that is isometric to Euclidean space, (ii) data lying on a low-dimensional manifold that is not isometric to Euclidean space, (iii) data that lies on or near a disconnected manifold, and (iv) data

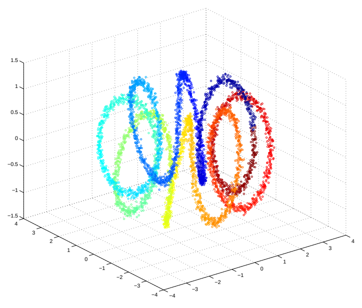
Technique	*Parameter settings*
PCA	None
Isomap	$5 \leq k \leq 15$
Kernel PCA	$\kappa = (\mathbf{XX}^T + 1)^5$
MVU	$5 \leq k \leq 15$
Diffusion maps	$10 \leq t \leq 100 \quad \sigma = 1$
LLE	$5 \leq k \leq 15$
Laplacian Eigenmaps	$5 \leq k \leq 15 \quad \sigma = 1$
Hessian LLE	$5 \leq k \leq 15$
LTSA	$5 \leq k \leq 15$
Sammon mapping	None
Autoencoders	Three hidden layers
LLC	$5 \leq k \leq 15 \quad 5 \leq m \leq 25$
Manifold charting	$5 \leq m \leq 25$

Table 2: Parameter settings for the experiments.

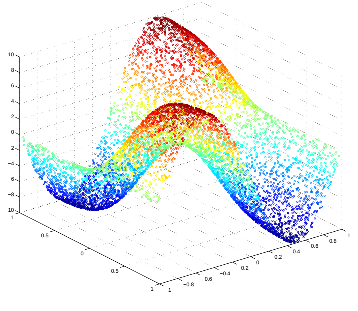
forming a manifold with a high intrinsic dimensionality. The artificial datasets on which we performed experiments are: the Swiss roll dataset (addressing i), the helix dataset (ii), the twin peaks dataset (ii), the broken Swiss roll dataset (iii), and the high-dimensional (HD) dataset (iv). The equations that were used to generate the artificial datasets are given in the appendix. Figure 4 shows plots of the first four artificial datasets. The HD dataset consists of points randomly sampled from a 5-dimensionial non-linear manifold embedded in a 10-dimensional space. In order to ensure that the generalization errors of the k-nearest neighbor classifiers reflect the quality of the data representations produced by the dimensionality reduction techniques, we assigned all datapoints to one of two classes according to a checkerboard pattern on the manifold. All artificial datasets consist of 5,000 samples. We opted for a fixed number of datapoints in each dataset, because in real-world applications, obtaining more training data is usually expensive.



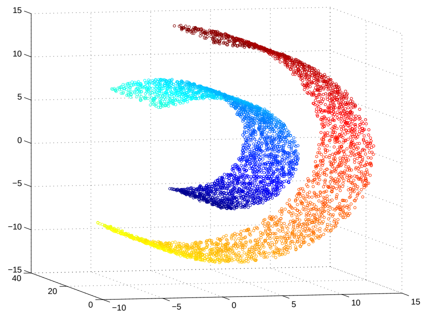
(a) Swiss roll dataset.



(b) Helix dataset.



(c) Twinpeaks dataset.



(d) Broken Swiss roll dataset.

Figure 4: Four of the artificial datasets.

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
Swiss roll (2D)	3.68%	29.76%	**3.40%**	30.24%	4.12%	33.50%	3.74%	22.06%	3.56%	3.90%	22.34%	49.00%	26.72%	22.66%
Helix (1D)	1.24%	35.50%	13.18%	38.04%	7.48%	35.44%	32.32%	15.24%	52.22%	**0.92%**	52.22%	52.22%	27.44%	25.94%
Twin peaks (2D)	0.40%	0.26%	0.22%	**0.12%**	0.56%	0.26%	0.94%	0.88%	0.14%	0.18%	0.32%	49.06%	11.04%	0.30%
Broken Swiss (2D)	2.14%	25.96%	14.48%	32.06%	32.06%	58.26%	36.94%	10.66%	**6.48%**	15.86%	27.40%	87.86%	37.06%	32.24%
HD (5D)	24.19%	22.18%	23.26%	27.46%	25.38%	23.14%	**20.74%**	24.70%	50.02%	42.62%	20.70%	49.18%	34.14%	21.34%

Table 3: Generalization errors of 1-NN classifiers trained on artificial datasets (smaller numbers are better).

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
Swiss roll (2D)	−−	0.88	0.99	0.88	**1.00**	0.81	**1.00**	0.92	**1.00**	**1.00**	0.89	0.46	0.81	0.88
Helix (1D)	−−	0.78	0.74	0.71	0.96	0.73	0.83	0.87	0.35	**1.00**	0.35	0.64	0.76	0.83
Twin peaks (2D)	−−	0.98	0.98	0.99	0.99	**1.00**	0.99	0.99	0.99	0.99	**1.00**	0.52	0.86	**1.00**
Broken Swiss (2D)	−−	0.96	**0.97**	0.96	**0.97**	0.78	0.94	**0.97**	0.92	0.89	**0.97**	0.70	0.86	0.96
HD (5D)	−−	**1.00**	0.98	**1.00**	0.98	**1.00**	**1.00**	0.98	0.56	0.94	**1.00**	0.68	0.89	**1.00**

Table 4: Trustworthinesses $T(12)$ on the artificial datasets (larger numbers are better).

6.1.2 Five Natural Datasets

For our experiments on natural datasets, we selected five datasets that represent tasks from a variety of domains: (1) the MNIST dataset, (2) the COIL20 dataset, (3) the NiSIS dataset, (4) the ORL dataset, and (5) the HIVA dataset. The MNIST dataset is a dataset of 60,000 handwritten digits. For computational reasons, we randomly selected 5,000 digits for our experiments. The images in the MNIST dataset have 28×28 pixels, and can thus be considered as points in a 784-dimensional space. The COIL20 dataset contains images of 20 different objects, depicted from 72 viewpoints, leading to a total of 1,440 images. The size of the images is 32×32 pixels, yielding a 1,024-dimensional space. The NiSIS dataset is a publicly available dataset for pedestrian detection, which consists of 3,675 grayscale images of size 36×18 pixels (leading to a space of dimensionality 648). The ORL dataset is a face recognition dataset that contains 400 grayscale images of 112×92 pixels that depict 40 faces under various conditions (i.e., the dataset contains 10 images per face). The HIVA dataset is a drug discovery dataset with two classes. It consists of 3,845 datapoints with dimensionality 1,617.

6.2 Experiments on Artificial Datasets

In Table 3, we present the generalization errors of 1-nearest neighbor classifiers that were trained and tested on the low-dimensional data representations obtained from the dimensionality reduction techniques. We ran the experiments for all parameter settings described in Table 2, and for each technique, we report the best generalization error of all runs in Table 3. In the table, the left column indicates the name of the dataset and the target dimensionality to which we attempted to transform the high-dimensional data. The best performing technique for each dataset is shown in boldface. Table 4 presents the corresponding trustworthiness values of the low-dimensional embeddings (again, only the best trustworthiness of all runs is reported). Table 5 presents the corresponding continuity measures. From the results in Table 3, 4, and 5, we make four observations.

First, the results reveal the strong performance of techniques based on neighborhood graphs (Isomap, MVU, LLE, Laplacian Eigenmaps, Hessian LLE, and LTSA) on artificial datasets such as the Swiss roll dataset. Techniques that do not employ neighborhood graphs (viz. PCA, diffusion maps, Kernel PCA, Sammon mapping, and autoencoders) perform poorly on artificial datasets such as the Swiss roll dataset. The performance of the two techniques that align local linear models (LLC and manifold charting) on

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
Swiss roll (2D)	−−	**1.00**	0.99	0.99	**1.00**	0.91	**1.00**	0.99	**1.00**	**1.00**	**1.00**	0.50	0.99	**1.00**
Helix (1D)	−−	0.98	0.97	0.98	**1.00**	0.98	0.99	0.99	0.50	**1.00**	0.50	0.75	0.98	0.99
Twin peaks (2D)	−−	**1.00**	0.99	0.99	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.50	0.98	**1.00**
Broken Swiss (2D)	−−	**1.00**	0.98	0.99	**1.00**	0.90	0.98	0.99	0.99	0.99	**1.00**	0.73	0.99	**1.00**
HD (5D)	−−	**1.00**	0.99	0.99	0.99	**1.00**	0.99	0.99	0.56	0.98	**1.00**	0.89	0.91	**1.00**

Table 5: Continuity $C(12)$ on the artificial datasets (larger numbers are better).

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
MNIST (20D)	5.11%	**6.74%**	12.64%	13.86%	13.58%	25.00%	10.02%	11.30%	91.66%	90.32%	6.90%	7.18%	16.12%	14.84%
COIL20 (5D)	0.14%	3.82%	15.69%	7.78%	25.14%	11.18%	22.29%	95.00%	50.35%	4.17%	**0.83%**	51.11%	4.31%	27.36%
ORL (8D)	2.50%	4.75%	27.50%	6.25%	24.25%	90.00%	11.00%	97.50%	56.00%	12.75%	2.75%	**6.25%**	11.25%	22.50%
NiSIS (15D)	8.24%	**7.95%**	13.36%	9.55%	15.67%	48.98%	15.48%	47.59%	48.98%	24.68%	48.98%	9.22%	26.86%	18.91%
HIVA (15D)	4.63%	5.05%	4.92%	5.07%	4.94%	5.46%	4.97%	4.81%	**3.51%**	**3.51%**	**3.51%**	5.12%	**3.51%**	4.79%

Table 6: Generalization errors of 1-NN classifiers trained on natural datasets (smaller numbers are better).

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
MNIST (20D)	−−	**1.00**	0.96	0.99	0.92	0.95	0.96	0.89	0.54	0.54	**1.00**	**1.00**	0.93	0.97
COIL20 (5D)	−−	**0.99**	0.89	0.98	0.92	0.91	0.93	0.27	0.69	0.96	**0.99**	0.88	0.96	0.92
ORL (8D)	−−	**0.99**	0.78	0.98	0.95	0.49	0.95	0.29	0.76	0.94	**0.99**	**0.99**	0.79	0.82
NiSIS (15D)	−−	**0.99**	0.89	**0.99**	0.90	0.40	0.92	0.47	0.47	0.82	0.47	**0.99**	0.85	0.89
HIVA (15D)	−−	0.97	0.87	0.89	0.89	0.75	0.80	0.78	0.42	0.54	0.42	**0.98**	0.91	0.95

Table 7: Trustworthinesses $T(12)$ on the natural datasets (larger numbers are better).

the Swiss roll dataset are comparable to those of techniques that do not employ neighborhood graphs (although manifold charting outperforms LLC).

Second, from the results of the experiments on the helix dataset, we observe that Hessian LLE, a technique that performs strong on the Swiss roll dataset, may perform less well on manifolds that are not isometric to Euclidean space. The performance of LLE on the helix dataset is also notably worse than its performance on the Swiss roll dataset. The other techniques based on neighborhood graphs (Isomap, MVU, LLE, Laplacian Eigenmaps, and LTSA) perform strong on the helix dataset, despite the non-isometric nature of the dataset.

Third, the high generalization errors (and relatively low trutworthinesses and continuities) on the broken Swiss roll dataset indicate that most nonlinear techniques for dimensionality reduction do not perform well under the presence of disconnected (i.e., non-smooth) manifolds in the data.

Fourth, from the results on the HD dataset, we observe that nonlinear techniques may have problems when they are faced with a dataset with a high intrinsic dimensionality. In particular, Hessian LLE and LTSA perform disappointing on the dataset with a high intrinsic dimensionality. On the HD dataset, the performance of PCA is surprisingly strong.

Taken together, the results show that manifold learners perform well on data that forms a low-dimensional manifold, such as the frequently used Swiss roll dataset. However, the results also reveal that the strong performance on the Swiss roll dataset does not always generalize to more complex datasets, such as datasets with disconnected manifolds, manifolds that are not isometric to the Euclidean space, or manifolds with a high intrinsic dimensionality.

6.3 Experiments on Natural Datasets

Table 6 presents the generalization errors of 1-nearest neighbor classifiers that were trained on the low-dimensional data representations of the natural datasets that were obtained from the dimensionality reduction techniques. Table 7 presents the corresponding trustworthiness values. Again, we only present the results of the best run of all runs that use the range of parameters that is given in Table 2. From the results in Table 6 and 7, we make two observations.

First, we observe that the performance of manifold learners on the natural datasets is disappointing compared to the performance of these techniques on the artificial datasets. In contrast, many techniques that do not employ neighborhood graphs such as PCA, Sammon mapping, and autoencoders perform

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
MNIST (20D)	−−	**1.00**	0.94	0.89	0.93	0.95	0.96	0.70	0.50	0.50	**1.00**	**1.00**	0.91	0.96
COIL20 (5D)	−−	**1.00**	0.90	0.98	0.97	0.92	0.95	0.47	0.71	0.99	**1.00**	0.92	0.96	0.95
ORL (8D)	−−	**0.99**	0.76	0.95	0.97	0.57	0.95	0.49	0.76	0.94	**0.99**	0.98	0.80	0.79
NiSIS (15D)	−−	**1.00**	0.84	0.98	0.94	0.48	0.91	0.48	0.47	0.64	0.47	**1.00**	0.84	0.89
HIVA (15D)	−−	**0.99**	0.84	0.88	0.94	0.80	0.80	0.54	0.51	0.62	0.51	**0.99**	0.87	0.96

Table 8: Continuity $C(12)$ on the natural datasets (larger numbers are better).

well on (most of) the natural datasets. In particular, PCA and autoencoders outperform the other techniques on four of the five datasets when the techniques are assessed based on the trustworthiness of their embeddings. On the COIL-20 dataset, the performance of autoencoders is slightly less strong, most likely due to the small number of instances that constitute this dataset, which hampers the successful training of the large number of weights in the network. Globally, the difference between the results of the experiments on the artificial and the natural datasets is significant: techniques that perform well on artificial datasets perform poorly on natural datasets, and vice versa.

Second, the results show that on many natural datasets, the classification performance of our classifiers was not improved by performing dimensionality reduction. Presumably, this is due to the characteristics of the intrinsic dimensionality estimator which we employed. This estimator may select target dimensionalities that are suboptimal in the sense that they do not minimize the generalization error of the trained classifiers. However, since we aim to compare the performance of dimensionality reduction techniques, and not to minimize generalization errors on classification problems, this observation is of no relevance to our study.

7 Discussion

In the previous sections, we presented a comparative study of techniques for dimensionality reduction. We observed that most nonlinear techniques do not outperform PCA on natural datasets, despite their ability to learn the structure of complex nonlinear manifolds. This section discusses the main weaknesses of current nonlinear techniques for dimensionality reduction that explain the results of our experiments. In addition, the section presents ideas on how to overcome these weaknesses. The discussion is subdivided into four parts. Subsection 7.1 discusses the main weaknesses of full spectral dimensionality reduction techniques. In subsection 7.2, we address five weaknesses of sparse spectral techniques for dimensionality reduction. Subsection 7.3 discusses the main weaknesses of the non-convex dimensionality reduction techniques. Subsection 7.4 summarizes the main weaknesses of current nonlinear techniques for dimensionality reduction and presents some concluding remarks on the future development of dimensionality reduction techniques.

7.1 Full Spectral Techniques

Our discussion on the results of full spectral techniques for dimensionality reduction is subdivided into two parts. First, we discuss the results of the two neighborhood graph-based techniques, Isomap and MVU. Second, we discuss weaknesses explaining the results of the two kernel-based techniques, Kernel PCA and diffusion maps.

For the first part, we remark that full spectral techniques for dimensionality reduction that employ neighborhood graphs, such as Isomap and MVU, are subject to many of the weaknesses of sparse spectral techniques that we will discuss in subsection 7.2. In particular, the construction of the neighborhood graph is susceptible to the curse of dimensionality, overfitting, and the presence of outliers (see 7.2 for a detailed explanation). In addition to this problems, Isomap suffers from short-circuiting: a single erroneous connection in the neighborhood graph may severely affect the pairwise geodesic distances, as a result of which the data is poorly embedded in the low-dimensional space. Moreover, Isomap uses classical scaling to construct a low-dimensional embedding from the pairwise geodesic distances. The cost function of classical scaling causes Isomap to focus on retaining the large geodesic distances, instead of on the small geodesic distance that constitute the local structure of the data. A possible solution to this problem is presented in [146]. MVU suffers from a similar problem as Isomap: a single short-circuit in the neighborhod graph may lead to an erroneous constraint in the semidefinite program that severely affects the performance of MVU.

For the second part, we remark that kernel-based techniques for dimensionality reduction (i.e., Kernel PCA and diffusion maps) do not suffer from the weaknesses of neighborhood graph-based techniques. However, the performance of Kernel PCA and diffusion maps on the Swiss roll dataset indicates that (similar to PCA) these techniques are incapable of modeling complex nonlinear manifolds. The main reason for this incapability is that kernel-based methods require the selection of a proper kernel function. In general, model selection in kernel methods is performed using some form of hold-out testing [54], leading to high computational costs. Alternative approaches to model selection for kernel methods are based on, e.g., maximizing the between-class margins or the data variance (as in MVU) using semidefinite programming [55, 77]. Despite these alternative approaches, the construction of a proper kernel remains an important obstacle for the successful application of Kernel PCA. In addition, depending on the selection of the kernel, kernel-based techniques for dimensionality reduction may suffer from similar weaknesses as other manifold learners. In particular, when a Gaussian kernel with a small bandwidth σ is employed, Kernel PCA and diffusion maps may be susceptible to the curse of intrinsic dimensionality (see 7.2), i.e., their performance may be inversely proportional to the intrinsic dimensionality of the data. Diffusion maps largely resolve the short-circuiting problems of Isomap by integrating over all paths through a graph defined of the data, however, they are still subject to the second problem of Isomap: diffusion maps focus on retaining large diffusion distances in the low-dimensional embedding, instead of on retaining the small diffusion distances that constitute the local structure of the data.

7.2 Sparse Spectral Techniques

The results of our experiments show that the performance of the popular sparse spectral techniques, such as LLE, is rather disappointing on many real-world datasets. Most likely, the poor performance of these techniques is due to one or more of the following five weaknesses.

First, sparse spectral techniques for dimensionality reduction suffer from a fundamental weakness in their cost function. For instance, the optimal solution of the cost function of LLE (see Equation 13) is the trivial solution in which the coordinates of all low-dimensional points $\mathbf{y}_i$ are zero. This solution is not selected because LLE has a constraint on the covariance of the solution, viz., the constraint $\|\mathbf{y}^{(k)}\|^2 = 1$ for $\forall k$. Although the covariance constraint may seem to have resolved the problem of selecting a trivial solution, it is easy to 'cheat' on it. In particular, LLE often constructs solutions in which most points are embedded on the origin, and there are a few 'strings' coming out of the origin that make sure the covariance constraint is met (at a relatively small cost). Moreover, the simple form of the covariance constraint in LLE may give rise to undesired rescalings of the manifold [52]. The same problems also apply to Laplacian Eigenmaps, Hessian LLE, and LTSA, which have similar covariance constraints.

Second, all sparse spectral dimensionality reduction techniques suffer from the curse of dimensionality of the embedded manifold, i.e., from the curse of the intrinsic dimensionality of the data [16, 136, 15], because the number of datapoints that is required to characterize a manifold properly grows exponentially with the intrinsic dimensionality of the manifold. The susceptibility to the curse of dimensionality is a fundamental weakness of all local learners, and therefore, it also applies to learning techniques that employ Gaussian kernels (such as Support Vector Machines). For artificial datasets with low intrinsic dimensionality such as the Swiss roll dataset, this weakness does not apply. However, in most real-world tasks, the intrinsic dimensionality of the data is much higher. For instance, the face space is estimated to consist of at least 100 dimensions [90]. As a result, the performance of local techniques is poor on many real-world datasets, which is illustrated by the results of our experiments with the natural datasets.

Third, the inferior performance of sparse spectral techniques for dimensionality reduction arises from the eigenproblems that the techniques attempt to solve. Typically, the smallest eigenvalues in these problems are very small (around 10^{-7} or smaller), whereas the largest eigenvalues are fairly big (around 10^2 or larger). Eigenproblems with these properties are extremely hard to solve, even for state-of-the-art eigensolvers. The eigensolver may not be able to identify the smallest eigenvalues of the eigenproblem, and as a result, the dimensionality reduction technique might produce suboptimal solutions. The good performance of Isomap and MVU (that search for the largest eigenvalues) compared to sparse spectral techniques (that search for the smallest eigenvalues) may be explained by the difficulty of solving eigenproblems.

Fourth, local properties of a manifold do not necessarily follow the global structure of the manifold (as noted in, e.g., [104, 24]) in the presence of noise around the manifold. In other words, sparse spectral techniques suffer from overfitting on the manifold. Moreover, sparse spectral techniques suffer from folding [23]. Folding is caused by a value of k that is too high with respect to the sampling density of (parts of) the manifold. Folding causes the local linearity assumption to be violated, leading to radial or other distortions. In real-world datasets, folding is likely to occur because the data density may vary over the manifold (i.e., because the data distribution is not uniform over the manifold). An approach that might overcome this weakness for datasets with small intrinsic dimensionality is adaptive neighborhood selection. Techniques for adaptive neighborhood selection are presented in, e.g., [135, 89, 107]. Furthermore, sparse spectral techniques for dimensionality reduction are sensitive to the presence of outliers in the data [28]. In local techniques for dimensionality reduction, outliers are connected to their k nearest neighbors, even when they are very distant. As a consequence, outliers degrade the performance of local techniques for dimensionality reduction. A possible approach to resolve this problem is the usage of an ϵ-neighborhood. In an ϵ-neighborhood, datapoints are connected to all datapoints that lie within a sphere with radius ϵ. A second approach to overcome the problem of outliers is preprocessing the data by removing outliers [148, 95].

Fifth, the local linearity assumption of sparse spectral techniques for dimensionality reduction implies that the techniques assume that the manifold contains no discontinuities (i.e., that the manifold is smooth). The results of our experiments with the broken Swiss dataset illustrate the incapability of sparse spectral dimensionality reduction techniques to model non-smooth manifolds. In real-world datasets, the underlying manifold is not likely to be smooth. For instance, a dataset that contains different object classes is likely to constitute a disconnected underlying manifold. In addition, most sparse spectral techniques cannot deal with manifolds that are not isometric to Euclidean space, which is illustrated by the results of our experiments with the helix and twinpeaks datasets. This may be a problem, because for instance, a dataset of objects depicted under various orientations gives rise to a manifold that is closed (similar to the helix dataset).

In addition to these five weaknesses, Hessian LLE and LTSA cannot transform data to a dimensionality higher than the number of nearest neighbors in the neighborhood graph, which might lead to difficulties with datasets with a high intrinsic dimensionality.

7.3 Non-convex Techniques

Obviously, the main problem of non-convex techniques is that they optimize non-convex objective functions, as a result of which they suffer from the presence of local optima in the objective functions. For instance, the EM algorithm that is employed in LLC and manifold charting is likely to get stuck in a local maximum of the log-likelihood function. In addition, LLC and manifold charting are hampered by the presence of outliers in the data. In techniques that perform global alignment of linear models (such as LLC), the sensitivity to the presence of outliers may be addressed by replacing the mixture of factor analyzers by a mixture of t-distributed subspaces (MoTS) model [36, 35]. The intuition behind the use of the MoTS model is that a t-distribution is less sensitive to outliers than a Gaussian (which tends to overestimate variances) because it is heavier-tailed.

For autoencoders, the presence of local optima in the objective function has largely been overcome by the pretraining of the network using RBMs or denoising autoencoders. A limitation of autoencoders is that they are only applicable on datasets of reasonable dimensionality. If the dimensionality of the dataset is very high, the number of weights in the network is too large to find an appropriate setting of the network. This limitation of autoencoders may be addressed by preprocessing the data using PCA. Moreover, successful training of autoencoders requires the availibility of sufficient amounts of data, as illustrated by our results with autoencoders on the COIL-20 dataset.

Despite the problems of the non-convex techniques mentioned above, our results show that convex techniques for dimensionality reduction do not necessarily outperform non-convex techniques for dimensionality reduction. In particular, multilayer autoencoders perform very well on all five natural datasets. Most likely, these results are due to the larger freedom in designing non-convex techniques, allowing the incorporation of procedures that circumvent many of the problems of (both full and sparse) spectral techniques mentioned above. In particular, multilayer autoencoders provide a deep architecture (i.e., an architecture with multiple nonlinear layers), as opposed to shallow architectures (i.e., architectures with a single layer of nonlinearity) such as the convex techniques that are discussed in this study [15]. The main advantage of such a deep architecture is that, in theory, it may require exponentially less datapoints to learn the structure of highly varying manifolds, as illustrated for a d-bits parity dataset in [13]. Hence, although convex techniques are much more popular in dimensionality reduction (and in machine learning in general), our results suggest that suboptimally optimizing a sensible objective function is a more viable approach than optimizing a convex objective function that contains obvious flaws. This claim is also supported by strong results that were recently obtained with t-SNE [128], a non-convex multidimensional scaling variant that was published after we performed our comparative study.

7.4 Main Weaknesses

Taken together, the results of our experiments indicate that, to date, nonlinear dimensionality reduction techniques perform strongly on selected datasets that typically contain well-sampled smooth manifolds, but that this strong performance does not necessarily extend to real-world data. This result agrees with the results of studies reported in the literature. On selected datasets, nonlinear techniques for dimensionality reduction outperform linear techniques [94, 123], but nonlinear techniques perform poorly on various other natural datasets [56, 68, 67, 86]. In particular, our results establish three main weaknesses of the popular sparse spectral techniques for dimensionality reduction: (1) flaws in their objective functions, (2) numerical problems in their eigendecompositions, and (3) their susceptibility to the curse of dimensionality. Some of these weaknesses also apply to Isomap and MVU.

From the first weakness, we may infer that a requirement for future dimensionality reduction techniques is that either the minimum of the cost function is a non-trivial solution, or the constraints on the objective are sufficiently complex as to prevent the technique from selecting a solution that is close to the trivial solution. Our results suggest the development of such cost function should be pursued, even if this prompts the use of a non-convex objective function. In the design of a non-convex technique, there is much more freedom to construct a sensible objective function that is not hampered by obvious flaws. The strong results of autoencoders support this claim, as well as recent results presented for t-SNE [128].

The second weakness leads to exactly the same suggestion, but for a different reason: convex objective functions are often hard to optimize as well. In particular, sparse eigendecompositions are subject to numerical problems because it is hard to distinguish the smallest eigenvalues from the trivial zero eigenvalue. Moreover, interior point methods such as those employed to solve the SDP in MVU require the computation of the Hessian, which may be prohibiting successful optimization for computational reasons (on medium-sized or large datasets, MVU can only be performed using a variety of approximations that result in suboptimal solutions).

From the third weakness, we may infer that a requirement for future techniques for dimensionality reduction is that they do not rely completely on local properties of the data. It has been suggested that the susceptibility to the curse of dimensionality may be addressed using techniques in which the global structure of the data manifold is represented in a number of linear models [16], however, the performance of LLC and manifold charting in our experiments is not good enough to support this suggestion. The strong performance of autoencoders in our experiments suggests that it is beneficial to use deep architectures that contain more than one layer of nonlinearity.

8 Conclusions

The paper presented a review and comparative study of techniques for dimensionality reduction. From the results obtained, we may conclude that nonlinear techniques for dimensionality reduction are, despite their large variance, often not capable of outperforming traditional linear techniques such as PCA. In the future, we foresee the development of new nonlinear techniques for dimensionality reduction that (i) do not suffer from the presence of trivial optimal solutions, (ii) may be based on non-convex objective functions, and (iii) do not rely on neighbourhood graphs to model the local structure of the data manifold. The other important concern in the development of novel techniques for dimensionality reduction is their optimization, which should be computationally and numerically feasible in practice.

Acknowledgements

The work is supported by NWO/CATCH, project RICH (grant 640.002.401). We thank the Dutch State Service for Cultural Heritage (RCE) for their cooperation.

A Related Techniques

The comparative review presented in this paper addresses all main techniques for (nonlinear) dimensionality reduction. However, it is not exhaustive.

The comparative review does not include self-organizing maps [73] and their probabilistic extension GTM [19], because these techniques combine a dimensionality reduction technique with clustering, as a result of which they do not fit in the dimensionality reduction framework that we discussed in Section 2. Techniques for Independent Component Analysis [12] are not included in our review, because they were mainly designed for blind-source separation. Linear Discriminant Analysis [46], Generalized Discriminant Analysis [9], and Neighborhood Components Analysis [53, 106], and recently proposed metric learners [32, 8, 51, 137] are not included in the review, because of their supervised nature. Furthermore, our comparative review does not cover a number of techniques that are variants or extensions of the thirteen reviewed dimensionality reduction techniques. These variants include factor analysis [117], Gaussian Process Latent Variable Models [80], principal curves [28], kernel maps [118], conformal eigenmaps [113], Geodesic Nullspace Analysis [24], Structure Preserving Embedding [114], variants of multidimensional scaling [3, 38, 45, 62, 92], techniques that (similarly to LLC and manifold charting) globally align a mixture of linear models [104, 109, 133], and linear variants of LLE [58, 74], Laplacian Eigenmaps [59], and LTSA [147]. Also, our review does not cover latent variable models that are tailored to a specific type of data such as Latent Dirichlet Allocation [20].

B Details of the Artificial Datasets

In this appendix, we present the equations that we used to generate the five artificial datasets. Suppose we have two random numbers p_i and q_i that were sampled from a uniform distribution with support $[0, 1]$. For the Swiss roll dataset, the datapoint x_i is constructed by computing $\mathbf{x}_i = [t_i \cos(t_i), t_i \sin(t_i), 30q_i]$, where $t_i = \frac{3\pi}{2}(1 + 2p_i)$. For the broken Swiss roll dataset, all datapoints $\mathbf{x}_i$ for which $\frac{2}{5} < t_i < \frac{4}{5}$ are rejected and resampled. For the helix dataset, the datapoint $\mathbf{x}_i$ is contructed by computing $\mathbf{x}_i = [(2 + \cos(8p_i)) \cos(p_i), (2 + \cos(8p_i)) \sin(p_i), \sin(8p_i)]$. For the twinpeaks dataset, the datapoint $\mathbf{x}_i$ is constructed by $\mathbf{x}_i = [1 - 2p_i, \sin(\pi - 2\pi p_i)) \tanh(3 - 6q_i)]$. To all datapoints, Gaussian noise with a small variance is added. The HD dataset is constructed by sampling from a five-dimensional uniform distribution (with domain $[0, 1]$ over each dimension), and embedding these in a ten-dimensional space by computing ten different combinations of the five random variables, some of which are linear and some of which are nonlinear. Matlab code that generates all artificial datasets is available in the Matlab Toolbox for Dimensionality Reduction at `http://ict.ewi.tudelft.nl/~lvandermaaten/dr`.

References

[1] M. Abdella and T. Marwala. The use of genetic algorithms and neural networks to approximate missing data in database. In *Proceedings of the IEEE International Conference on Computational Cybernetics*, pages 207–212, 2005.

[2] G. Afken. *Gram-Schmidt Orthogonalization*. Academic Press, Orlando, FL, USA, 1985.

[3] D.K. Agrafiotis. Stochastic proximity embedding. *Journal of Computational Chemistry*, 24(10):1215–1221, 2003.

[4] W.N. Anderson and T.D. Morley. Eigenvalues of the Laplacian of a graph. *Linear and Multilinear Algebra*, 18:141–145, 1985.

[5] W.E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9:17–25, 1951.

[6] C. Baker. *The numerical treatment of integral equations*. Clarendon Press, 1977.

[7] M. Balasubramanian and E.L. Schwartz. The Isomap algorithm and topological stability. *Science*, 295(5552):7, 2002.

[8] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a Mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6(1):937–965, 2006.

[9] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

[10] M. Belkin and P. Niyogi. Laplacian Eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, volume 14, pages 585–591, Cambridge, MA, USA, 2002. The MIT Press.

[11] M. Belkin and P. Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56(1–3):209–239, 2004.

[12] A.J. Bell and T.J. Sejnowski. An information maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[13] Y. Bengio. Learning deep architectures for AI. Technical Report 1312, Université de Montréal, 2007.

[14] Y. Bengio, O. Delalleau, N. Le Roux, J.-F. Paiement P. Vincent, and M. Ouimet. Learning eigenfunctions links spectral embedding and Kernel PCA. *Neural Computation*, 16(10):2197–2219, 2004.

[15] Y. Bengio and Y. LeCun. Scaling learning algorithms towards AI. In L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors, *Large-Scale Kernel Machines*, pages 321–360. MIT Press, 2007.

[16] Y. Bengio and M. Monperrus. Non-local manifold tangent learning. In *Advances in Neural Information Processing Systems*, volume 17, pages 129–136, Cambridge, MA, USA, 2004. The MIT Press.

[17] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems*, volume 16, Cambridge, MA, USA, 2004. The MIT Press.

[18] B.L. Betechuoh, T. Marwala, and T. Tettey. Autoencoder networks for HIV classification. *Current Science*, 91(11):1467–1473, 2006.

[19] C. Bishop, M. Svensen, and C. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[20] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[21] B. Borchers and J.G. Young. Implementation of a primaldual method for SDP on a shared memory parallel architecture. *Computational Optimization and Applications*, 37(3):355–369, 2007.

[22] S.P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, New York, NY, USA, 2004.

[23] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems*, volume 15, pages 985–992, Cambridge, MA, USA, 2002. The MIT Press.

[24] M. Brand. From subspaces to submanifolds. In *Proceedings of the* 15^{th}*British Machine Vision Conference*, London, UK, 2004. British Machine Vision Association.

[25] A. Brun, H.-J. Park, H. Knutsson, and C.-F. Westin. Coloring of DT-MRI fiber traces using Laplacian Eigenmaps. In *Proccedings of the Eurocast 2003, Neuro Image Workshop*, 2003.

[26] C.J.C. Burges. *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers*, chapter Geometric Methods for Feature Selection and Dimensional Reduction: A Guided Tour. Kluwer Academic Publishers, 2005.

[27] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-resolution through neighbor embedding. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 275–282, 2004.

[28] K.-Y. Chang and J. Ghosh. Principal curves for nonlinear feature extraction and classification. In *Applications of Artificial Neural Networks in Image Processing III*, pages 120–129, Bellingham, WA, USA, 1998. SPIE.

[29] C. Chatfield and A.J. Collins. *Introduction to Multivariate Analysis*. Chapman and Hall, 1980.

[30] T.-J. Chin and D. Suter. Out-of-sample extrapolation of learned manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(9):1547–1556, 2008.

[31] H. Choi and S. Choi. Robust kernel Isomap. *Pattern Recognition*, 40(3):853–862, 2007.

[32] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similiarty metric discriminatively, with application to face verication. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 349–356, 2005.

[33] J.A. Costa and A.O. Hero. Classification constrained dimensionality reduction. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pages 1077–1080, 2005.

[34] T. Cox and M. Cox. *Multidimensional scaling*. Chapman & Hall, London, UK, 1994.

[35] D. de Ridder and V. Franc. Robust manifold learning. Technical Report CTU-CMP-2003-08, Department of Cybernetics, Czech Technical University, Prague, Czech Republic, April 2003.

[36] D. de Ridder and V. Franc. Robust subspace mixture models using t-distributions. In *Proceedings of the British Machine Vision Conference 2003*, pages 319–328, 2003.

[37] V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems*, volume 15, pages 721–728, Cambridge, MA, USA, 2003. The MIT Press.

[38] P. Demartines and J. Hérault. Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. *IEEE Transactions on Neural Networks*, 8(1):148–154, 1997.

[39] D. DeMers and G. Cottrell. Non-linear dimensionality reduction. In *Advances in Neural Information Processing Systems*, volume 5, pages 580–587, San Mateo, CA, USA, 1993. Morgan Kaufmann.

[40] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[41] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[42] D.L. Donoho and C. Grimes. Hessian eigenmaps: New locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 102(21):7426–7431, 2005.

[43] R. Duraiswami and V.C. Raykar. The manifolds of spatial hearing. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, volume 3, pages 285–288, 2005.

[44] S. Ekins, K.V. Balakin, N. Savchuk, and Y. Ivanenkov. Insights for human ether-a-go-go-related gene potassium channel inhibition using recursive partitioning and Kohonen and Sammon mapping techniques. *Journal of Medicinal Chemistry*, 49(17):5059–5071, 2006.

[45] C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, New York, NY, USA, 1995. ACM Press.

[46] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[47] R.W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962.

[48] D.R. Fokkema, G.L.G. Sleijpen, and H.A. van der Vorst. Jacobi–Davidson style QR and QZ algorithms for the reduction of matrix pencils. *SIAM Journal on Scientific Computing*, 20(1):94–125, 1999.

[49] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[50] Z. Ghahramani and G.E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, Department of Computer Science, University of Toronto, 1996.

[51] A. Globerson and S. Roweis. Metric learning by collapsing classes. In *Advances of Neural Information Processing Systems*, volume 18, pages 451–458, 2006.

[52] Y. Goldberg, A. Zakai, D. Kushnir, and Y. Ritov. Manifold learning: The price of normalization. *Journal of Machine Learning Research*, 9:1909–1939, 2008.

[53] J. Goldberger, S. Roweis, G.E. Hinton, and R.R. Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems*, volume 17, pages 513–520, Cambridge, MA, 2005. MIT Press.

[54] G. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21:215–224, 1979.

[55] T. Graepel. Kernel matrix completion by semidefinite programming. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 694–699, Berlin, Germany, 2002. Springer-Verlag.

[56] A.B.A. Graf and F.A. Wichmann. Gender classification of human faces. In *Biologically Motivated Computer Vision 2002, LNCS 2525*, pages 491–501, 2002.

[57] J. Hamm, D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. Technical Report TR-110, Max Planck Institute for Biological Cybernetics, Germany, 2003.

[58] X. He, D. Cai, S. Yan, and H.-J. Zhang. Neighborhood preserving embedding. In *Proceedings of the* 10^{th} *IEEE International Conference on Computer Vision*, pages 1208–1213, 2005.

[59] X. He and P. Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems*, volume 16, page 37, Cambridge, MA, USA, 2004. The MIT Press.

[60] G.E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[61] G.E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[62] G.E. Hinton and S.T. Roweis. Stochastic Neighbor Embedding. In *Advances in Neural Information Processing Systems*, volume 15, pages 833–840, Cambridge, MA, USA, 2002. The MIT Press.

[63] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[64] H. Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007.

[65] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[66] R. Huber, H. Ramoser, K. Mayer, H. Penz, and M. Rubik. Classification of coins using an eigenspace approach. *Pattern Recognition Letters*, 26(1):61–75, 2005.

[67] N.P. Hughes and L. Tarassenko. Novel signal shape descriptors through wavelet transforms and dimensionality reduction. In *Wavelet Applications in Signal and Image Processing X*, pages 763–773, 2003.

[68] O.C. Jenkins and M.J. Mataric. Deriving action and behavior primitives from human motion data. In *International Conference on Intelligent Robots and Systems*, volume 3, pages 2551–2556, 2002.

[69] L.O. Jimenez and D.A. Landgrebe. Supervised classification in high-dimensional space: geometrical,statistical, and asymptotical properties of multivariate data. *IEEE Transactions on Systems, Man and Cybernetics*, 28(1):39–54, 1997.

[70] N. Kambhatla and T.K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

[71] R. Kharal. Semidefinite embedding for the dimensionality reduction of DNA microarray data. Master's thesis, University of Waterloo, 2006.

[72] K.I. Kim, K. Jung, and H.J. Kim. Face recognition using kernel principal component analysis. *IEEE Signal Processing Letters*, 9(2):40–42, 2002.

[73] T. Kohonen. *Self-organization and associative memory: 3^{rd} edition*. Springer-Verlag New York, Inc., New York, NY, USA, 1989.

[74] E. Kokiopoulou and Y. Saad. Orthogonal Neighborhood Preserving Projections: A projection-based dimensionality reduction technique. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2134–2156, 2007.

[75] S.Y. Kung, K.I. Diamantaras, and J.S. Taur. Adaptive Principal component EXtraction (APEX) and applications. *IEEE Transactions on Signal Processing*, 42(5):1202–1217, 1994.

[76] S. Lafon and A.B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1393–1403, 2006.

[77] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, and L.E. Ghaouiand M.I. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[78] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. Exploring strategies for training deep neural networks. *Journal of Machine Learning Research*, 10(Jan):1–40, 2009.

[79] M.H. Law and A.K. Jain. Incremental nonlinear dimensionality reduction by manifold learning. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 28(3):377–391, 2006.

[80] N.D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6(Nov):1783–1816, 2005.

[81] J.A. Lee, A. Lendasse, N. Donckers, and M. Verleysen. A robust nonlinear projection method. In *Proceedings of the* 8^{th} *European Symposium on Artificial Neural Networks*, pages 13–20, 2000.

[82] J.A. Lee and M. Verleysen. Nonlinear dimensionality reduction of data manifolds with essential loops. *Neurocomputing*, 67:29–53, 2005.

[83] J.A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer, New York, NY, USA, 2007.

[84] E. Levina and P.J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems*, volume 17, Cambridge, MA, USA, 2004. The MIT Press.

[85] H. Li, L. Teng, W. Chen, and I.-F. Shen. Supervised learning on local tangent space. In *Lecture Notes on Computer Science*, volume 3496, pages 546–551, Berlin, Germany, 2005. Springer Verlag.

[86] I.S. Lim, P.H. Ciechomski, S. Sarni, and D. Thalmann. Planar arrangement of high-dimensional biomedical data sets by Isomap coordinates. In *Proceedings of the* 16^{th} *IEEE Symposium on Computer-Based Medical Systems*, pages 50–55, 2003.

[87] A. Lima, H. Zen, Y. Nankaku, C. Miyajima, K. Tokuda, and T. Kitamura. On the use of Kernel PCA for feature extraction in speech recognition. *IEICE Transactions on Information Systems*, E87-D(12):2802–2811, 2004.

[88] M. Martin-Merino and A. Munoz. A new Sammon algorithm for sparse data visualization. In *Proceedings of the* 17^{th} *International Conference on Pattern Recognition*, pages 477–481, 2004.

[89] N. Mekuz and J.K. Tsotsos. Parameterless Isomap with adaptive neighborhood selection. In *Proceedings of the* 28^{th} *DAGM Symposium*, pages 364–373, Berlin, Germany, 2006. Springer.

[90] M. Meytlis and L. Sirovich. On the dimensionality of face space. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 29(7):1262–1267, 2007.

[91] B. Nadler, S. Lafon, R.R. Coifman, and I.G. Kevrekidis. Diffusion maps, spectral clustering and the reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis: Special Issue on Diffusion Maps and Wavelets*, 21:113–127, 2006.

[92] K. Nam, H. Je, and S. Choi. Fast Stochastic Neighbor Embedding: A trust-region algorithm. In *Proceedings of the IEEE International Joint Conference on Neural Networks 2004*, volume 1, pages 123–128, Budapest, Hungary, 2004.

[93] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, volume 14, pages 849–856, Cambridge, MA, USA, 2001. The MIT Press.

[94] M. Niskanen and O. Silvén. Comparison of dimensionality reduction methods for wood surface inspection. In *Proceedings of the* 6^{th} *International Conference on Quality Control by Artificial Vision*, pages 178–188, Gatlinburg, TN, USA, 2003. International Society for Optical Engineering.

[95] J.-H. Park, Z. Zhang, H. Zha, and R. Kasturi. Local smoothing for manifold learning. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 452–459, 2004.

[96] M. Partridge and R. Calvo. Fast dimensionality reduction and Simple PCA. *Intelligent Data Analysis*, 2(3):292–298, 1997.

[97] N. Patwari and A.O. Hero. Manifold learning algorithms for localization in wireless sensor networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages 857–860, 2004.

[98] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philiosophical Magazine*, 2:559–572, 1901.

[99] J.C. Platt. FastMap, MetricMap, and Landmark MDS are all Nyström algorithms. In *Proceedings of the* 10^{th} *International Workshop on Artificial Intelligence and Statistics*, pages 261–268, 2005.

[100] A.M. Posadas, F. Vidal, F. de Miguel, G. Alguacil, J. Pena, J.M. Ibanez, and J. Morales. Spatial-temporal analysis of a seismic series using the principal components method. *Journal of Geophysical Research*, 98(B2):1923–1932, 1993.

[101] N.M. Rajpoot, M. Arif, and A.H. Bhalerao. Unsupervised learning of shape manifolds. In *Proceedings of the British Machine Vision Conference*, 2007.

[102] B. Raytchev, I. Yoda, and K. Sakaue. Head pose estimation by nonlinear manifold learning. In *Proceedings of the* 17^{th} *ICPR*, pages 462–466, 2004.

[103] S.T. Roweis. EM algorithms for PCA and SPCA. In *Advances in Neural Information Processing Systems*, volume 10, pages 626–632, 1997.

[104] S.T. Roweis, L. Saul, and G. Hinton. Global coordination of local linear models. In *Advances in Neural Information Processing Systems*, volume 14, pages 889–896, Cambridge, MA, USA, 2001. The MIT Press.

[105] S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by Locally Linear Embedding. *Science*, 290(5500):2323–2326, 2000.

[106] R.R. Salakhutdinov and G.E. Hinton. Learning a non-linear embedding by preserving class neighbourhood structure. In *Proceedings of the* 11^{th} *International Conference on Artificial Intelligence and Statistics*, volume 2, pages 412–419, 2007.

[107] O. Samko, A.D. Marshall, and P.L. Rosin. Selection of the optimal parameter value for the Isomap algorithm. *Pattern Recognition Letters*, 27(9):968–979, 2006.

[108] J.W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18(5):401–409, 1969.

[109] G. Sanguinetti. Dimensionality reduction of clustered datasets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):535–540, 2008.

[110] L.K. Saul, K.Q. Weinberger, J.H. Ham, F. Sha, and D.D. Lee. Spectral methods for dimensionality reduction. In *Semisupervised Learning*, Cambridge, MA, USA, 2006. The MIT Press.

[111] A. Saxena, A. Gupta, and A. Mukerjee. Non-linear dimensionality reduction by locally linear isomaps. *Lecture Notes in Computer Science*, 3316:1038–1043, 2004.

[112] B. Schölkopf, A.J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[113] F. Sha and L.K. Saul. Analysis and extension of spectral methods for nonlinear dimensionality reduction. In *Proceedings of the* 22^{nd} *International Conference on Machine Learning*, pages 785–792, 2005.

[114] B. Shaw and T. Jebara. Structure preserving embedding. In *Proceedings of the* 26^{th} *International Conference on Machine Learning*, 2009.

[115] J. Shawe-Taylor and N. Christianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, UK, 2004.

[116] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[117] C. Spearman. General intelligence objectively determined and measured. *American Journal of Psychology*, 15:206–221, 1904.

[118] J.A.K. Suykens. Data visualization and dimensionality reduction using kernel maps with a reference point. Technical Report 07-22, ESAT-SISTA, K.U. Leuven, 2007.

[119] M. Takatsuka. An application of the self-organizing map and interactive 3-D visualization to geospatial data. In *Proceedings of the* 6^{th} *International Conference on GeoComputation*, 2001.

[120] Y.W. Teh and S.T. Roweis. Automatic alignment of hidden representations. In *Advances in Neural Information Processing Systems*, volume 15, pages 841–848, Cambridge, MA, USA, 2002. The MIT Press.

[121] J.B. Tenenbaum. Mapping a manifold of perceptual observations. In *Advances in Neural Information Processing Systems*, volume 10, pages 682–688, Cambridge, MA, USA, 1998. The MIT Press.

[122] J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[123] L. Teng, H. Li, X. Fu, W. Chen, and I.-F. Shen. Dimension reduction of microarray data based on local tangent space alignment. In *Proceedings of the* 4^{th} *IEEE International Conference on Cognitive Informatics*, pages 154–159, 2005.

[124] M.E. Tipping. Sparse kernel principal component analysis. In *Advances in Neural Information Processing Systems*, volume 13, pages 633–639, Cambridge, MA, USA, 2000. The MIT Press.

[125] M.E. Tipping and C.M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

[126] W.S. Torgerson. Multidimensional scaling I: Theory and method. *Psychometrika*, 17:401–419, 1952.

[127] M.A. Turk and A.P. Pentland. Face recognition using eigenfaces. In *Proceedings of the Computer Vision and Pattern Recognition 1991*, pages 586–591, 1991.

[128] L.J.P. van der Maaten and G.E. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2431–2456, 2008.

[129] L.J.P. van der Maaten, E.O. Postma, and H.J. van den Herik. Dimensionality reduction: A comparative review, 2008.

[130] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[131] J. Venna. *Dimensionality reduction for visual exploration of similarity structures*. PhD thesis, Helsinki University of Technology, 2007.

[132] J. Venna and S. Kaski. Visualizing gene interaction graphs with local multidimensional scaling. In *Proceedings of the* 14^{th} *European Symposium on Artificial Neural Networks*, pages 557–562, 2006.

[133] J. Verbeek. Learning nonlinear image manifolds by global alignment of local linear models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1236–1250, 2006.

[134] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the International Conference on Machine Learning*, pages 1096–1103, 2008.

[135] J. Wang, Z. Zhang, and H. Zha. Adaptive manifold learning. In *Advances in Neural Information Processing Systems*, volume 17, pages 1473–1480, Cambridge, MA, USA, 2005. The MIT Press.

[136] K.Q. Weinberger, B.D. Packer, and L.K. Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In *Proceedings of the* 10^{th} *International Workshop on AI and Statistics*, Barbados, WI, 2005. Society for Artificial Intelligence and Statistics.

[137] K.Q. Weinberger and L.K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

[138] K.Q. Weinberger, F. Sha, and L.K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the* 21^{st} *International Confernence on Machine Learning*, 2004.

[139] K.Q. Weinberger, F. Sha, Q. Zhu, and L.K. Saul. Graph Laplacian regularization for large-scale semidefinite programming. In *Advances in Neural Information Processing Systems*, volume 19, 2007.

[140] Y. Weiss. Segmentation using eigenvectors: a unifying view. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, pages 975–982, Los Alamitos, CA, USA, 1999. IEEE Computer Society Press.

[141] M. Welling, F. Agakov, and C.K.I. Williams. Extreme components analysis. In S. Thrun, L.K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, Cambridge, MA, 2004. MIT Press.

[142] M. Welling, M. Rosen-Zvi, and G. Hinton. Exponential family harmoniums with an application to information retrieval. In *Advances in Neural Information Processing Systems*, volume 17, pages 1481–1488, 2004.

[143] C.K.I. Williams. On a connection between Kernel PCA and metric multidimensional scaling. *Machine Learning*, 46(1-3):11–19, 2002.

[144] L. Xiao, J. Sun, and S. Boyd. A duality view of spectral methods for dimensionality reduction. In *Proceedings of the* 23^{rd} *International Conference on Machine Learning*, pages 1041–1048, 2006.

[145] R. Xu, S. Damelin, and D.C. Wunsch. Applications of diffusion maps in gene expression data-based cancer diagnosis analysis. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4613–4616, 2007.

[146] L. Yang. Sammon's nonlinear mapping using geodesic distances. In *Proceedings of the* 17^{th} *International Conference on Pattern Recognition*, volume 2, pages 303–306, 2004.

[147] T. Zhang, J. Yang, D. Zhao, and X. Ge. Linear local tangent space alignment and application to face recognition. *Neurocomputing*, 70:1547–1533, 2007.

[148] Z. Zhang and H. Zha. Local linear smoothing for nonlinear manifold learning. Technical Report CSE-03-003, Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, USA, 2003.

[149] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimensionality reduction via local tangent space alignment. *SIAM Journal of Scientific Computing*, 26(1):313–338, 2004.