Dimensionality Reduction: A Comparative Review

L.J.P. van der Maaten *, E.O. Postma, H.J. van den Herik
MICC, Maastricht University, P.O. Box 616, 6200 MD Maastricht, The Netherlands.

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. 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 do not outperform the traditional PCA on 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.

Key words: Dimensionality reduction, manifold learning, feature extraction.

1. Introduction

Real-world data, such as speech signals, digital photographs, or fMRI scans, usually has a high dimensionality. In order to handle 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 [43]. Dimensionality reduction is important in many domains, since it mitigates the curse of dimensionality and other undesired properties of high-dimensional spaces [60]. 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) and factor analysis (see, e.g., [18]). However, these linear techniques cannot adequately handle complex nonlinear data.

Therefore, in the last decade, a large number of nonlinear techniques for dimensionality reduction have been proposed (see for an overview, e.g., [23,95,113]). 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 be highly nonlinear. 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 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 scarce. 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 [81], but this comparison is very limited in scope with respect to

* Corresponding author.
Email address: l.vandermaaten@micc.unimaas.nl (L.J.P. van der Maaten).

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 dimenisonality 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 the real-world datasets.

Next to PCA, the paper investigates the following twelve nonlinear techniques: (1) multidimensional scaling, (2) Isomap, (3) Maximum Variance Unfolding, (4) Kernel PCA, (5) diffusion maps, (6) multilayer autoencoders, (7) Locally Linear Embedding, (8) Laplacian Eigenmaps, (9) Hessian LLE, (10) Local Tangent Space Analysis, (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. Section 3 briefly discusses the most important linear technique for dimensionality reduction (PCA). Subsequently, Section 4 describes and discusses the selected twelve nonlinear techniques for dimensionality reduction. Section 5 lists all techniques by theoretical characteristics. Then, in Section 6, we present an empirical comparison of twelve 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. Our main conclusion is that the focus of the research community should shift towards nonlocal techniques for dimensionality reduction with objective functions that can be optimized well in practice (such as PCA, Kernel PCA, and autoencoders).

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 X consisting of n datavectors 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 X are lying on or near a manifold with dimensionality d that is embedded in the D-dimensional space. Dimensionality reduction techniques transform dataset X with dimensionality D into a new dataset 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 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 x_i, where x_i is the ith row of the D-dimensional data matrix X. The low-dimensional counterpart of x_i is denoted by y_i, where y_i is the ith row of the d-dimensional data matrix Y. In the remainder of the paper, we adopt the notation presented above.

Figure 1 shows a taxonomy of techniques for dimensionality reduction. The main distinction between techniques for dimensionality reduction is the distinction between linear and nonlinear techniques. Linear techniques assume that the data lie on or near a linear subspace of the high-dimensional space. Nonlinear techniques for dimensionality reduction do not rely on the linearity assumption as a result of which more complex embeddings of the data in the high-dimensional space can be identified. The further subdivisions in the taxonomy are discussed in the review in the following two sections.

3. Linear techniques for dimensionality reduction

Linear techniques perform 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 as a benchmark. We briefly discuss PCA below.

Principal Components Analysis (PCA) [56] constructs a low-dimensional representation of the data that describes as much of the variance in the data as possible.

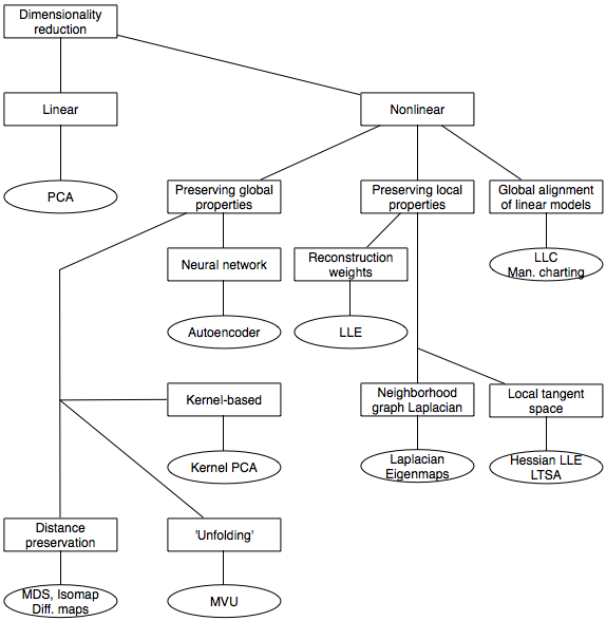


Fig. 1. Taxonomy of dimensionality reduction techniques.

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 M that maximizes $M^T \operatorname{cov}(X) M$, where $\operatorname{cov}(X)$ is the covariance matrix of the data X. It can be shown that this linear mapping is formed by the d principal eigenvectors (i.e., principal components) of the covariance matrix of the zero-mean data [1]. Hence, PCA solves the eigenproblem

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

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

PCA has been successfully applied in a large number of domains such as face recognition [110], coin classification [57], and seismic series analysis [87]. The main drawback of PCA is that 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 computing the eigenvectors of the squared Euclidean distance matrix $(X - \bar{X})(X - \bar{X})^T$ instead of the eigenvectors of the covariance matrix [2]. Alternatively, iterative techniques such as Simple PCA [83] or probabilistic PCA [90] may be employed.

[1] PCA maximizes $M^T \operatorname{cov}(X) M$ with respect to M, under the constraint that $|M| = 1$. This constraint can be enforced by introducing a Lagrange multiplier λ. Hence, an unconstrained maximization of $M^T \operatorname{cov}(X) M + \lambda(1 - M^T M)$ is performed. A stationary point of this quantity is to be found when $\operatorname{cov}(X) M = \lambda M$.

4. Nonlinear techniques for dimensionality reduction

In Section 3, we discussed the main linear technique for dimensionality reduction, which is established and well studied. In contrast, most nonlinear techniques for dimensionality reduction have been proposed more recently and are therefore less well studied. In this section, we discuss twelve nonlinear techniques for dimensionality reduction, as well as their weaknesses and applications as reported in the literature. Nonlinear techniques for dimensionality reduction can be subdivided into three main types [3]: (1) techniques that attempt to preserve global properties of the original data in the low-dimensional representation, (2) techniques that attempt to preserve local properties of the original data in the low-dimensional representation, and (3) techniques that perform global alignment of a mixture of linear models. In subsection 4.1, we discuss six global nonlinear techniques for dimensionality reduction. Subsection 4.2 presents four local nonlinear techniques for dimensionality reduction. Subsection 4.3 presents two techniques that perform a global alignment of linear models.

4.1. *Global techniques*

Global nonlinear techniques for dimensionality reduction are techniques that attempt to preserve global properties of the data. The subsection presents six global nonlinear techniques for dimensionality reduction: (1) MDS, (2) Isomap, (3) MVU, (4) diffusion maps, (5) Kernel PCA, and (6) multilayer autoencoders. The techniques are discussed in 4.1.1 to 4.1.6.

4.1.1. *MDS*

Multidimensional scaling (MDS) [29,66] represents a collection of nonlinear techniques that maps the high-dimensional data representation to a low-dimensional representation while retaining the pairwise distances between the datapoints as much as possible. The quality of the mapping is expressed in the stress function, a

[2] It can be shown that the eigenvectors u_i and v_i of the matrices $X^T X$ and $X X^T$ are related through $v_i = \frac{1}{\lambda_i} X u_i$, see, e.g., [26].

[3] Although diffusion maps and Kernel PCA are global methods, they may behave as local methods depending on the kernel selection.

measure of the error between the pairwise distances in the low-dimensional and high-dimensional representation of the data. Two important examples of stress functions (for metric MDS) are the raw stress function and the Sammon cost function. The raw stress function is defined by

$$\phi(Y) = \sum_{ij} (\|x_i - x_j\| - \|y_i - y_j\|)^2, \qquad (2)$$

in which $\|x_i - x_j\|$ is the Euclidean distance between the high-dimensional datapoints x_i and x_j, and $\|y_i - y_j\|$ is the Euclidean distance between the low-dimensional datapoints y_i and y_j. The Sammon cost function is given by

$$\phi(Y) = \frac{1}{\sum_{ij} \|x_i - x_j\|} \sum_{i \neq j} \frac{(\|x_i - x_j\| - \|y_i - y_j\|)^2}{\|x_i - x_j\|}. \qquad (3)$$

The Sammon cost function differs from the raw stress function in that it puts more emphasis on retaining distances that were originally small. The minimization of the stress function can be performed using various methods, such as the eigendecomposition of a pairwise dissimilarity matrix, the conjugate gradient method, or a pseudo-Newton method [29].

MDS is widely used for the visualization of data, e.g., in fMRI analysis [103] and in molecular modeling [112]. The popularity of MDS has led to the proposal of variants such as SPE [3], CCA [32], SNE [53,79], and FastMap [39]. In addition, there exist nonmetric variants of MDS, that aim to preserve ordinal relations in data, instead of pairwise distances [29].

4.1.2. *Isomap*

Multidimensional scaling has proven to be successful in many applications, but it suffers from the fact that it is based on 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 [105], MDS might consider two datapoints as near points, whereas their distance over the manifold is much larger than the typical interpoint distance. Isomap [105] 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 [105], the geodesic distances between the datapoints x_i ($i = 1, 2, \ldots, n$) are computed by constructing a neighborhood graph G, in which every datapoint x_i is connected with its k nearest neighbors x_{i_j} ($j = 1, 2, \ldots, k$) in the dataset X. The shortest path between two points in the graph forms a good (over)estimate of the geodesic distance between these two points, and can easily be computed using Dijkstra's or Floyd's shortest-path algorithm [35,41]. The geodesic distances between all datapoints in X are computed, thereby forming a pairwise geodesic distance matrix. The low-dimensional representations y_i of the datapoints x_i in the low-dimensional space Y are computed by applying multidimensional scaling (see 4.1.1) on the resulting 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 [71] 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 [27] or by removing nearest neighbors that violate local linearity of the neighborhood graph [96]. A second weakness is that Isomap may suffer from 'holes' in the manifold. This problem can be dealt with by tearing manifolds with holes [71]. A third weakness of Isomap is that it can fail if the manifold is nonconvex [106]. Despite these three weaknesses, Isomap was successfully applied on tasks such as wood inspection [81], visualization of biomedical data [74], and head pose estimation [89].

4.1.3. *MVU*

Because of its resemblance to Isomap, we discuss MVU before diffusion maps (see 4.1.4). Maximum Variance Unfolding (MVU) is very similar to Isomap in that it defines a neighborhood graph on the data and retains pairwise distances in the resulting graph [117]. 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 efficiently using semidefinite programming.

MVU starts with the construction of a neighborhood graph G, in which each datapoint x_i is connected to its k nearest neighbors 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.

Maximize $\sum_{ij} \| y_i - y_j \|^2$ subject to:

(1) $\| y_i - y_j \|^2 = \| x_i - x_j \|^2$ for $\forall (i,j) \in G$

MVU reformulates the optimization problem as a semidefinite programming problem (SDP) [111] by defining a matrix K that is the inner product of the low-dimensional data representation Y. The optimization problem is identical to the following SDP.

Maximize $\mathrm{trace}(K)$ subject to (1), (2), and (3), with:

(1) $k_{ii} - 2k_{ij} + k_{jj} = \| x_i - x_j \|^2$ for $\forall (i,j) \in G$

(2) $\sum_{ij} k_{ij} = 0$

(3) $K \geq 0$

From the solution K of the SDP, the low-dimensional data representation Y can be obtained by performing a singular value decomposition.

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 for, e.g., sensor localization [118] and DNA microarray data analysis [62].

4.1.4. *Diffusion maps*

The diffusion maps (DM) framework [68,78] 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.

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 W with entries

$$w_{ij} = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}, \tag{4}$$

where σ^2 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 $P^{(1)}$ is formed with entries

$$p_{ij}^{(1)} = \frac{w_{ij}}{\sum_k w_{ik}}. \tag{5}$$

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

$$D^{(t)}(x_i, x_j) = \sqrt{\sum_k \frac{(p_{ik}^{(t)} - p_{jk}^{(t)})^2}{\psi(x_k)^{(0)}}}. \tag{6}$$

In the equation, $\psi(x_i)^{(0)}$ is a term that attributes more weight to parts of the graph with high density. It is defined by $\psi(x_i)^{(0)} = \frac{m_i}{\sum_j m_j}$, where m_i is the degree of node x_i defined by $m_i = \sum_j p_{ij}$. From Equation 6, it can be observed that pairs of datapoints with a high forward transition probability have a small diffusion distance. The key idea behind the diffusion distance is that it is based on many paths through the graph. This makes the diffusion distance more robust to noise than, e.g., the geodesic distance that is employed in Isomap. In the low-dimensional representation of the data Y, diffusion maps attempt to retain the diffusion distances. Using spectral theory on the random walk, it has been shown (see, e.g., [68]) that the low-dimensional representation Y that retains the diffusion distances is formed by the d nontrivial principal eigenvectors of the eigenproblem

$$P^{(t)} v = \lambda v. \tag{7}$$

Because the graph is fully connected, the largest eigenvalue is trivial (viz. $\lambda_1 = 1$), and its eigenvector v_1 is thus discarded. The low-dimensional representation 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

$$Y = \{\lambda_2 v_2, \lambda_3 v_3, \ldots, \lambda_{d+1} v_{d+1}\}. \tag{8}$$

Diffusion maps have been successfully applied to, e.g., shape matching [88] and gene expression analysis [121].

4.1.5. *Kernel PCA*

Kernel PCA (KPCA) is the reformulation of traditional linear PCA in a high-dimensional space that is constructed using a kernel function [97]. 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 [99]. 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 non-linear mappings.
Kernel PCA computes the kernel matrix K of the datapoints x_i. The entries in the kernel matrix are defined by

$$k_{ij} = \kappa(x_i, x_j), \tag{9}$$

where κ is a kernel function [99]. Subsequently, the kernel matrix K is centered using the following modification of the entries

$$k_{ij} = k_{ij} - \frac{1}{n}\sum_l k_{il} - \frac{1}{n}\sum_l k_{jl} + \frac{1}{n^2}\sum_{lm} k_{lm}. \tag{10}$$

The centering operation corresponds to subtracting the mean of the features in traditional PCA. It makes sure that the features in the high-dimensional space defined by the kernel function are zero-mean. Subsequently, the principal d eigenvectors v_i of the centered kernel matrix are computed. The eigenvectors of the covariance matrix α_i (in the high-dimensional space constructed by κ) can now be computed, since they are related to the eigenvectors of the kernel matrix v_i (see, e.g., [26]) through

$$\alpha_i = \frac{1}{\sqrt{\lambda_i}} X v_i. \tag{11}$$

In order to obtain the low-dimensional data representation, the data is projected onto the eigenvectors of the covariance matrix α_i. The result of the projection (i.e., the low-dimensional data representation Y) is given by

$$y_i = \left\{ \sum_{j=1}^{n} \alpha_1^j \kappa(x_j, x_i), \ldots, \sum_{j=1}^{n} \alpha_d^j \kappa(x_j, x_i) \right\}, \tag{12}$$

where α_1^j indicates the jth value in the vector α_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 4 [99].
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 [108]. Kernel PCA has been successfully applied to, e.g., face recognition [63], speech recognition [75], and novelty detection [55].

4.1.6. *Multilayer autoencoders*

Multilayer encoders are feed-forward neural networks with an odd number of hidden layers [33,54]. 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 x_i leads to a network in which the middle hidden layer gives a d-dimensional representation of the datapoints that preserves as much information in X as possible. The low-dimensional representations y_i can be obtained by extracting the node values in the middle hidden layer, when datapoint x_i is used as input. If linear activation functions are used in the neural network, an autoencoder is very similar to PCA [67]. 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.
Multilayer autoencoders usually have a high number of connections. Therefore, backpropagation approaches converge slowly and are likely to get stuck in local minima. In [54], this drawback is overcome by a learning procedure that consists of three main stages.
First, the recognition layers of the network (i.e., the layers from X to Y) are trained one-by-one using Restricted Boltzmann Machines (RBMs). RBMs are two-layer neural networks with visual and hidden nodes that are binary and stochastic [4]. RBMs can be trained efficiently using an unsupervised learning procedure that minimizes the so-called contrastive divergence [52]. Second, the reconstruction layers of the network (i.e., the layers from Y to 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.
Autoencoders have succesfully been applied to problems such as missing data imputation [1] and HIV analysis [16].

4.2. *Local techniques*

Subsection 4.1 presented six techniques for dimensionality reduction that attempt to retain global properties of the data. In contrast, local nonlinear techniques

[4] For continuous data, the binary nodes may be replaced by mean-field logistic or exponential family nodes.

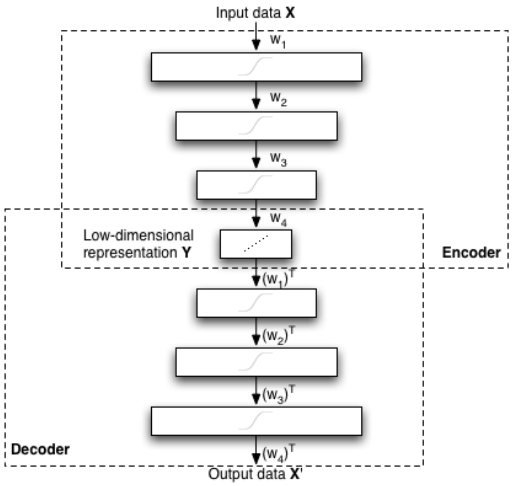


Fig. 2. Schematic structure of an autoencoder.

for dimensionality reduction are based on solely preserving properties of small neighborhoods around the datapoints. The central claim of these techniques is that by preservation of local properties of the data, the global layout of the data manifold is retained as well. This subsection presents four local nonlinear techniques for dimensionality reduction: (1) LLE, (2) Laplacian Eigenmaps, (3) Hessian LLE, and (4) LTSA in 4.2.1 to 4.2.4.

4.2.1. *LLE*

Local Linear Embedding (LLE) [92] is a local technique for dimensionality reduction that is similar to Isomap 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 properties are affected if short-circuiting occurs. Furthermore, the preservation of local properties allows for successful embedding of nonconvex manifolds. In LLE, the local properties of the data manifold are constructed by writing the 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 x_i by writing the datapoint as a linear combination W_i (the so-called reconstruction weights) of its k nearest neighbors x_{i_j}. Hence, LLE fits a hyperplane through the datapoint x_i and its nearest neighbors, thereby assuming that the manifold is locally linear. The local linearity assumption implies that the reconstruction weights W_i of the datapoints 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 W_i that reconstruct datapoint x_i from its neighbors in the high-dimensional data representation also reconstruct datapoint y_i from its neighbors in the low-dimensional data representation. As a consequence, finding the d-dimensional data representation Y amounts to minimizing the cost function

$$\phi(Y) = \sum_i (y_i - \sum_{j=1}^{k} w_{ij} y_{i_j})^2. \qquad (13)$$

Roweis and Saul [92] showed [5] that the coordinates of the low-dimensional representations y_i that minimize this cost function are found by computing the eigenvectors corresponding to the smallest d nonzero eigenvalues of the inproduct $(I-W)^T(I-W)$. In this formula, I is the $n \times n$ identity matrix.

The popularity of LLE has led to the proposal of linear variants of the algorithm [49,65], and to successful applications to, e.g., superresolution [24] and sound source localization [37]. However, there also exist experimental studies that report weak performance of LLE. In [74], LLE was reported to fail in the visualization of even simple synthetic biomedical datasets. In [59], 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 contains holes [92]. In addition, LLE tends to collapse large portions of the data very close together in the low-dimensional space.

4.2.2. *Laplacian Eigenmaps*

Similar to LLE, Laplacian Eigenmaps find a low-dimensional data representation by preserving local properties of the manifold [9]. 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

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

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. The Laplacian Eigenmap algorithm first constructs a neighborhood graph G in which every datapoint x_i is connected to its k nearest neighbors. For all points x_i and 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 4), leading to a sparse adjacency matrix W. In the computation of the low-dimensional representations y_i, the cost function that is minimized is given by

$$\phi(Y) = \sum_{ij}(y_i - y_j)^2 w_{ij}. \tag{14}$$

In the cost function, large weights w_{ij} correspond to small distances between the datapoints x_i and x_j. Hence, the difference between their low-dimensional representations y_i and y_j highly contributes to the cost function. As a consequence, nearby points in the high-dimensional space are brought closer together in the low-dimensional representation.

The computation of the degree matrix M and the graph Laplacian L of the graph W allows for formulating the minimization problem as an eigenproblem [4]. The degree matrix M of W is a diagonal matrix, of which the entries are the row sums of W (i.e., $m_{ii} = \sum_j w_{ij}$). The graph Laplacian L is computed by $L = M - W$. It can be shown that the following holds [6]

$$\phi(Y) = \sum_{ij}(y_i - y_j)^2 w_{ij} = 2Y^T LY. \tag{15}$$

Hence, minimizing $\phi(Y)$ is proportional to minimizing $Y^T LY$. The low-dimensional data representation Y can thus be found by solving the generalized eigenvalue problem

$$Lv = \lambda Mv \tag{16}$$

for the d smallest nonzero eigenvalues. The d eigenvectors v_i corresponding to the smallest nonzero eigenvalues form the low-dimensional data representation Y.

Laplacian Eigenmaps have been successfully applied to, e.g., clustering [80,100,119], face recognition [51], and the analysis of fMRI data [22]. In addition, variants of Laplacian Eigenmaps may be applied to supervised or semi-supervised learning problems [28,10]. A linear variant of Laplacian Eigenmaps is presented in [50].

[6] Note that $\phi(Y) = \sum_{ij}(y_i - y_j)^2 w_{ij} = \sum_{ij}(y_i^2 + y_j^2 - 2y_i y_j)w_{ij} = \sum_i y_i^2 m_{ii} + \sum_j y_j^2 m_{jj} - 2\sum_{ij} y_i y_j w_{ij} = 2Y^T MY - 2Y^T WY = 2Y^T LY$

4.2.3. *Hessian LLE*

Hessian LLE (HLLE) [36] 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 [7] that the coordinates of the low-dimensional representation can be found by performing an eigenanalysis of $\mathcal{H}$.

Hessian LLE starts with identifying the k nearest neighbors for each datapoint 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 x_i can be found by applying PCA on its k nearest neighbors x_{i_j}. In other words, for every datapoint x_i, a basis for the local tangent space at point x_i is determined by computing the d principal eigenvectors $M = \{m_1, m_2, \ldots, m_d\}$ of the covariance matrix $\mathrm{cov}(x_{i\cdot})$. Note that the above requires that $k \geq d$. Subsequently, an estimator for the Hessian of the manifold at point x_i in local tangent space coordinates is computed. In order to do this, a matrix Z_i is formed that contains (in the columns) all cross products of M up to the dth order (including a column with ones). The matrix Z_i is orthonormalized by applying Gram-Schmidt orthonormalization [2]. The estimation of the tangent Hessian H_i is now given by the transpose of the last $\frac{d(d+1)}{2}$ columns of the matrix Z_i. Using the Hessian estimators in local tangent coordinates, a matrix $\mathcal{H}$ is constructed with entries

$$\mathcal{H}_{lm} = \sum_i \sum_j ((H_i)_{jl} \times (H_i)_{jm}). \tag{17}$$

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 Y, which contains the low-dimensional representation of the data. A successful application of

[7] The derivation is too extensive for this paper; it can be found in [36].

Hessian LLE to sensor localization has been presented in [84].

4.2.4. *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 [124]. 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 [124]. 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 x_i. This is done by applying PCA on the k datapoints x_{i_j} that are neighbors of datapoint x_i. This results in a mapping M_i from the neighborhood of 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 L_i from the local tangent space coordinates θ_{i_j} to the low-dimensional representations y_{i_j}. Using this property of the local tangent space, LTSA performs the following minimization

$$\min_{Y_i, L_i} \sum_i \parallel Y_i J_k - L_i \Theta_i \parallel^2, \tag{18}$$

where J_k is the centering matrix of size k [99]. Zhang and Zha [124] have shown that the solution of the minimization is formed by the eigenvectors of an alignment matrix B, that correspond to the d smallest nonzero eigenvalues of B. The entries of the alignment matrix B are obtained by iterative summation (for all matrices V_i and starting from $b_{ij} = 0$ for $\forall ij$)

$$B_{N_i N_i} = B_{N_i N_i} + J_k(I - V_i V_i^T)J_k, \tag{19}$$

where N_i is a selection matrix that contains the indices of the nearest neighbors of datapoint x_i. Subsequently, the low-dimensional representation Y is obtained by computation of the eigenvectors corresponding to the d smallest nonzero eigenvectors of the symmetric matrix $\frac{1}{2}(B + B^T)$.

In [107], a successful application of LTSA to microarray data is reported. A linear variant of LTSA is proposed in [122].

4.3. *Global alignment of linear models*

In the previous subsections, we discussed techniques that compute a low-dimensional data representation by preserving global or local properties of the data. Techniques that perform global alignment of linear models combine these two types: they compute a number of locally linear models and perform a global alignment of these linear models. This subsection presents two such techniques, viz., LLC and manifold charting. The techniques are discussed separately in subsection 4.3.1 and 4.3.2.

4.3.1. *LLC*

Locally Linear Coordination (LLC) [104] 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 Expectation Maximization (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) using the EM algorithm [34,44,61]. Alternatively, a mixture of probabilistic PCA models (MoPPCA) model [109] could be employed. The local linear models in the mixture are used to construct m data representations z_{ij} and their corresponding responsibilities r_{ij} (where $j \in \{1, \ldots, m\}$) for every datapoint x_i. The responsibilities r_{ij} describe to what extent datapoint 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 $u_{ij} = r_{ij} z_{ij}$ are computed. The responsibility-weighted data representations u_{ij} are stored in a $n \times mD$ block matrix U. The alignment of the local models is performed based on U and on a matrix M that is given by $M = (I - W)^T(I - W)$. Herein, the matrix W contains the reconstruction weights computed by LLE (see 4.2.1), and I denotes the $n \times n$ identity matrix. LLC aligns the local models by solving the generalized eigenproblem

$$Av = \lambda Bv, \tag{20}$$

for the d smallest nonzero eigenvalues [8]. In the equation, A is the inproduct of $M^T U$ and B is the inproduct of U. The d eigenvectors v_i form a matrix L, that can be shown to define a linear mapping from the

[8] The derivation of this eigenproblem can be found in [104].

responsibility-weighted data representation U to the underlying low-dimensional data representation Y. The low-dimensional data representation is thus obtained by computing $Y = UL$.
LLC has been successfully applied to face images of a single person with variable pose and expression, and to handwritten digits [104].

4.3.2. *Manifold charting*

Similar to LLC, manifold charting constructs a low-dimensional data representation by aligning a MoFA or MoPPCA model [20]. 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 z_{ij} and corresponding responsibilities r_{ij} (where $j \in \{1, \ldots, m\}$) for all datapoints x_i. Manifold charting finds a linear mapping from the data representations z_{ij} to the global coordinates y_i that minimizes the cost function

$$\phi(Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} r_{ij} \parallel y_i - y_{ik} \parallel^2, \tag{21}$$

where $y_i = \sum_{k=1}^{m} r_{ik} y_{ik}$. 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

$$\phi(Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} r_{ij} r_{ik} \parallel y_{ij} - y_{ik} \parallel^2, \tag{22}$$

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 D with m blocks by

$$D = \begin{pmatrix} D_1 & \ldots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & D_m \end{pmatrix}, \tag{23}$$

where D_j is the sum of the weighted covariances of the data representations z_{ij}. Hence, D_j is given by

$$D_j = \sum_{i=1}^{n} r_{ij} \operatorname{cov}([Z_j \; 1]). \tag{24}$$

In Equation 24, the 1-column is added to the data representation Z_j in order to facilitate translations in the construction of y_i from the data representations z_{ij}. Using the definition of the matrix D and the $n \times mD$ block-diagonal matrix U with entries $u_{ij} = r_{ij}[z_{ij} \; 1]$, the manifold charting cost function can be rewritten as

$$\phi(Y) = L^T (D - U^T U) L, \tag{25}$$

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

$$(D - U^T U) v = \lambda U^T U v, \tag{26}$$

for the d smallest nonzero eigenvalues. The d eigenvectors v_i form the columns of the linear combination L from $[U \; 1]$ to Y.

5. Characterization of the techniques

In the Sections 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 in 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. Below, we discuss three types of relations between the techniques.
First, traditional PCA is identical to performing Kernel PCA with a linear kernel. Autoencoders in which only linear activation functions are employed are very similar to PCA as well [67]. Performing (metric) multidimensional scaling using the raw stress function with squared Euclidean distances is identical to performing PCA, due to the relation between the eigenvectors of the covariance matrix and the squared Euclidean distance matrix (see Section 3).
Second, performing MDS using geodesic distances is

identical to performing Isomap. Similarly, performing Isomap with the number of nearest neighbors k set to $n-1$ is identical to performing traditional MDS (and thus also to performing PCA). Diffusion maps are similar to MDS and Isomap, in that they attempt to preserve pairwise distances (the so-called diffusion distances). The main discerning property of diffusion maps is that it aims to retain a weighted sum of the distances of 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 [12,48]. As a result, Isomap, LLE, and Laplacian Eigenmaps can be considered as special cases of Kernel PCA (using a specific kernel function). For instance, this relation is visible in the out-of-sample extensions of Isomap, LLE, and Laplacian Eigenmaps [15]. The out-of-sample extension for these techniques is performed by means of a so-called Nyström approximation [6,86], which is known to be equivalent to the Kernel PCA projection [97]. 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 generally not essential in Kernel PCA [99]) and (2) diffusion maps do not employ the principal eigenvector of the Gaussian kernel, whereas Kernel PCA does. MVU can also be viewed upon as a special case of Kernel PCA, in which 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 [120]. Evaluation of the dual MVU problem has also shown that LLE and Laplacian Eigenmaps show great resemblance to MVU [120].

As a consequence of these relations between the techniques, our empirical comparative evaluation in Section 6 does not include (1) Kernel PCA using a linear kernel, (2) MDS, 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 convexity of the optimization problem, (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.

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

Table 1
Properties of techniques for dimensionality reduction.

For property 1, Table 1 shows that most techniques for dimensionality reduction optimize a convex cost function. This is advantageous, because it allows for finding the global optimum of the cost function. Because of their nonconvex cost functions, autoencoders, LLC, and manifold charting may suffer from getting stuck in local optima.

For property 2, Table 1 shows that 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 iterative techniques for dimensionality reduction have additional free parameters, such as the learning rate and the permitted maximum number of iterations. The presence of free parameters has advantages as well as 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 dimension-

ality 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, which is performed using a power method in $O(D^3)$. The corresponding memory complexity of PCA is $O(D^2)$ [9]. MDS, Isomap, diffusion maps, and Kernel PCA perform an eigenanalysis of an $n \times n$ matrix using a power method in $O(n^3)$. Because Isomap, diffusion maps, and Kernel PCA store a full $n \times n$ kernel matrix, the memory complexity of these techniques is $O(n^2)$.

In contrast to the spectral techniques discussed above, 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 [19]. 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 MoPPCA model, which has computational complexity $O(imd^3)$. The corresponding memory complexity is $O(nmd)$.

Similar to, e.g., Kernel PCA, local techniques perform an eigenanalysis of an $n \times n$ matrix. However, for local techniques the $n \times n$ matrix is sparse. The sparsity of the matrices is beneficial, because it lowers the computational complexity of the eigenanalysis. Eigenanalysis of a sparse matrix (using Arnoldi methods [5] or Jacobi-Davidson methods [42]) 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) some nonlinear techniques for dimensionality reduction may suffer from getting stuck in local optima, (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 the linear technique. Attempts to reduce the computational and/or memory complexities of nonlinear techniques have been proposed for, e.g., Isomap [31,70], MVU [116,118], and Kernel PCA [108].

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. For instance, in linear techniques such as PCA, this transformation is defined by the linear mapping M that was applied to the original data. For Kernel PCA, a similar transformation is available, although this transformation requires some additional kernel function computations [97]. 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, a nonparametric out-of-sample extension for Isomap, LLE, and Laplacian Eigenmaps has been presented in [15], in which the techniques are redefined as kernel methods. Subsequently, the out-of-

[9] 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).

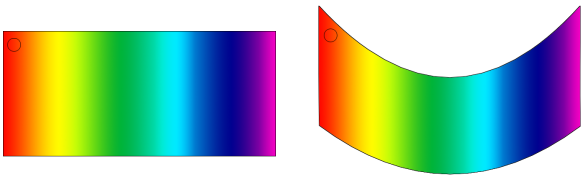
sample extension is performed using the Nyström approximation [86], which approximates the eigenvectors of a large $n \times n$ matrix based on the eigendecomposition of a smaller $m \times m$ submatrix of the large matrix. Similar nonparametric out-of-sample extensions for Isomap are proposed in [27,31]. 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 [116]. 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 (similar to Laplacian Eigenmaps) [118]. A nonparametric out-of-sample extension that can be applied to all nonlinear dimensionality reduction techniques is proposed in [73]. 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 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 linear and nonlinear 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.
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 twelve [10] techniques for dimensionality reduction on the high-dimensional representation of the data. Subsequently, we assess the quality of the resulting low-dimensional data representation by evaluating to what extent the local structure of the data is retained. The evaluation is performed by measuring the generalization errors of k-nearest neighbor classifiers that are trained on the low-dimensional data representation. A similar evaluation scheme is employed in [94]. We motivate our experimental setup below.
First, we opt for an evaluation of the local structure of the data, because for successful visualization or classification of data only its local structure needs to be retained. We evaluate how well the local structure of the data is retained by measuring the generalization error of k-nearest neighbor classifiers trained on the resulting data representations, because the high variance of this classifier (for small values of k). The high variance of the k-nearest neighbor classifier makes it very well suitable to assess the quality of the local structure of the data.
Second, we opt for an evaluation of the quality based on generalization errors instead of one based on reconstruction errors for two main reasons. The first reason is that 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). In other words, reconstruction errors measure the quality of the global structure of the low-dimensional data representation, and not the quality of the local structure. The second reason is that our main aim is to investigate the performance of the techniques on real-world datasets, in which the true underlying manifold of the data is usually unknown, and as a result, reconstruction errors cannot be computed.
For all dimensionality reduction techniques except for Isomap and MVU, we performed experiments without



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

Fig. 3. Two low-dimensional data representations.

[10] Please note that we do not consider MDS in the experiments, because of its similarity to PCA (see 5.1).

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 and MVU, we employ the respective out-of-sample extensions of these techniques (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 these techniques is necessary because the traditional implementations of Isomap and MVU can only embed the points that comprise the largest component of the neighborhood graph.

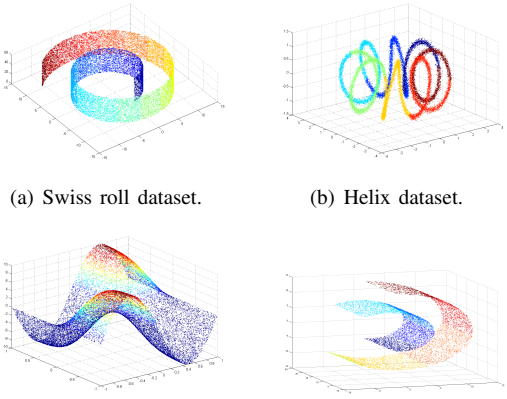
The parameter settings employed in our experiments are listed in Table 2. Most parameters were optimized using an exhaustive grid search within a reasonable range, which is shown in Table 2. For two parameters (σ in diffusion maps and Laplacian Eigenmaps), we employed fixed values in order to restrict 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 [72]. 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 results of the experiments were obtained using leave-one-out validation.

Technique	*Parameter settings*
PCA	None
Isomap	$5 \leq k \leq 15$
MVU	$5 \leq k \leq 15$
Diffusion maps	$10 \leq t \leq 100 \quad \sigma = 1$
Kernel PCA	$\kappa = (XX^T + 1)^5$
Autoencoders	Three hidden layers
LLE	$5 \leq k \leq 15$
Laplacian Eigenmaps	$5 \leq k \leq 15 \quad \sigma = 1$
Hessian LLE	$5 \leq k \leq 15$
LTSA	$5 \leq k \leq 15$
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.

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 discontinuous manifold, and (iv) data 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). 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.

Fig. 4. Four of the artificial datasets.

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 10,000 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 trained on the low-dimensional data representations obtained from the dimensionality reduction techniques. The table shows generalization errors measured using leave-one-out validation on the five artificial datasets. 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 (see subsection 6.1). The best performing technique for a dataset is shown in boldface. From the results in Table 3, we make five observations.

First, the results reveal that the nonlinear techniques employing neighborhood graphs (viz. Isomap, MVU, LLE, Laplacian Eigenmaps, Hessian LLE, LTSA, and LLC) outperform the other techniques on standard manifold learning problems, such as the Swiss roll dataset. Techniques that do not employ neighborhood graphs (viz. PCA, diffusion maps, Kernel PCA, autoencoders, and manifold charting) perform poorly on these datasets. The performances of LLC and manifold charting on the Swiss roll dataset are comparable to those of techniques that do not employ neighborhood graphs.

Second, from the results with the helix and twin peaks datasets, we observe that three local nonlinear dimensionality reduction techniques perform less well on manifolds that are not isometric to Euclidean space. The performance of Isomap, MVU, and LTSA on these datasets is still very strong. In addition, we observe that all neighborhood graph-based techniques outperform techniques that do not employ neighborhood graphs (including PCA).

Third, the results on the broken Swiss roll dataset indicate that most nonlinear techniques for dimensionality reduction cannot deal with discontinuous (i.e., non-smooth) manifolds. On the broken Swiss roll dataset, Hessian LLE is the only technique that does not suffer severely from the presence of a discontinuity in the manifold (when compared to the performance of the techniques on the original Swiss roll dataset).

Fourth, from the results on the HD dataset, we observe that most nonlinear techniques have major problems when faced with a dataset with a high intrinsic dimensionality. In particular, local dimensionality reduction techniques perform disappointing on a dataset with a high intrinsic dimensionality. On the HD dataset, PCA is only outperformed by Isomap and an autoencoder, which is the best performing technique on this dataset.

Fifth, we observe that Hessian LLE fails to find a solution on the helix dataset. The failure is the result of the inability of the eigensolver to solve the eigenproblem up to sufficient precision. Both Arnoldi [5] and Jacobi-Davidson eigendecomposition methods [42] suffer from this problem, that is caused by the nature of the eigenproblem that needs to be solved.

Taken together, the results show that although local techniques for dimensionality reduction perform strongly on a simple dataset such as the Swiss roll dataset, this strong performance does not generalize very well to more complex datasets (e.g., datasets with non-smooth manifolds, manifolds that are non-isometric to the Euclidean space, or manifolds with a high intrinsic dimensionality).

6.3. *Experiments on natural datasets*

Table 4 presents the generalization errors of 1-nearest neighbor classifiers that were trained on the low-dimensional data representations obtained from the dimensionality reduction techniques. From the results in Table 4, we make three observations.

First, we observe that the performance of nonlinear techniques for dimensionality reduction on the natural datasets is rather disappointing compared to the performance of these techniques on, e.g., the Swiss roll dataset. In particular, PCA outperforms all nonlinear techniques on three of the five datasets. Especially local nonlinear techniques for dimensionality reduction perform disappointingly. However, Kernel PCA and autoencoders perform strongly on almost all datasets.

Dataset (d)	*None*	*PCA*	*Isomap*	*MVU*	*KPCA*	*DM*	*Autoenc.*	*LLE*	*LEM*	*HLLE*	*LTSA*	*LLC*	*MC*
Swiss roll (2D)	3.68%	30.56%	3.28%	5.12%	29.30%	28.06%	30.58%	7.44%	10.16%	3.10%	**3.06%**	27.74%	42.74%
Helix (1D)	1.24%	38.56%	**1.22%**	3.76%	44.54%	36.18%	32.50%	20.38%	10.34%	failed	1.68%	26.68%	28.16%
Twinpeaks (2D)	0.40%	0.18%	0.30%	0.58%	0.08%	**0.06%**	0.12%	0.54%	0.52%	0.10%	0.36%	12.96%	**0.06%**
Broken Swiss (2D)	2.14%	27.62%	14.24%	36.28%	27.06%	23.92%	26.32%	37.06%	26.08%	**4.78%**	16.30%	26.96%	23.92%
HD (5D)	24.19%	22.14%	20.45%	23.62%	29.25%	34.75%	**16.29%**	35.81%	41.70%	47.97%	40.22%	38.69%	31.46%

Table 3
Generalization errors of 1-NN classifiers trained on artificial datasets.

Dataset (d)	*None*	*PCA*	*Isomap*	*MVU*	*KPCA*	*DM*	*Autoenc.*	*LLE*	*LEM*	*HLLE*	*LTSA*	*LLC*	*MC*
MNIST (20D)	5.11%	**5.06%**	28.54%	18.35%	65.48%	59.79%	14.10%	19.21%	19.45%	89.55%	32.52%	36.29%	38.22%
COIL20 (5D)	0.14%	3.82%	14.86%	21.88%	7.78%	4.51%	**1.39%**	9.86%	14.79%	43.40%	12.36%	6.74%	18.61%
ORL (8D)	2.50%	**4.75%**	44.20%	39.50%	5.50%	49.00%	69.00%	9.00%	12.50%	56.00%	12.75%	50.00%	62.25%
NiSIS (15D)	8.24%	**8.73%**	20.57%	19.40%	11.70%	22.94%	9.82%	28.71%	43.08%	45.00%	failed	26.86%	20.41%
HIVA (15D)	4.63%	5.05%	4.97%	4.89%	5.07%	3.51%	4.84%	5.23%	5.23%	failed	6.09%	**3.43%**	5.20%

Table 4
Generalization errors of 1-NN classifiers trained on natural datasets.

Second, the results in Table 4 reveal that two techniques (viz., Hessian LLE and LTSA) fail on one dataset. The failures of Hessian LLE and LTSA are the result of the inability of the eigensolver to identify the smallest eigenvalues (due to numerical problems). Both Arnoldi [5] and Jacobi-Davidson eigendecomposition methods [42] suffer from this limitation.
Third, the results show that on some natural datasets, the classification performance of our classifiers was not improved by performing dimensionality reduction. Most likely, this is due to errors in the intrinsic dimensionality estimator we employed. As a result, the target dimensionalities may not be optimal (in the sense that they minimize the generalization error of the trained classifier). 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.

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 three parts. Subsection 7.1 discusses five weaknesses of local techniques for dimensionality reduction. In subsection 7.2, weaknesses of global techniques and techniques that globally align a collection of linear models are discussed. Subsection 7.3 summarizes the main weaknesses of current nonlinear techniques for dimensionality reduction and present recommendations for the development of future dimensionality reduction techniques.

7.1. *Local techniques*

The results of our experiments show that the performance of popular techniques based on neighborhood graphs is rather disappointing on many datasets. Most likely, the poor performance of these techniques is due to one or more of the following five weaknesses.
First, local dimensionality reduction techniques suffer from the curse of dimensionality of the embedded manifold (i.e., the intrinsic dimension of the data) [13,14,116], 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 diffusion maps, Support Vector Machines, and Gaussian processes). 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 [77]. As a result, the performance of local techniques is poor on many real-world datasets, which explains the results of our experiments with the HD dataset and the natural datasets.
Second, the inferior performance of local nonlinear 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 results of Isomap (that searches for the largest eigenvalues) compared to local techniques (that search for the smallest eigenvalues) may be explained by the difficulty of solving eigenproblems.
Third, local properties of a manifold do not necessarily follow the global structure of the manifold (as noted in, e.g., [20,91]) in the presence of noise around the manifold. In other words, local methods suffer from overfitting on the manifold. Furthermore, local techniques for dimensionality reduction are sensitive to the presence of outliers in the data [25]. 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 [82,123].
Fourth, the local linearity assumption of local techniques for dimensionality reduction implies that the techniques assume that the manifold contains no discontinuities (i.e., the manifold is non-smooth). The results of our experiments with the broken Swiss dataset illustrate the incapability of current dimensionality reduction techniques to model non-smooth manifolds. In real-world datasets, the underlying manifold is not likely to be smooth. For instance, objects in real-world images undergo translations, which gives rise to discontinuities in the underlying manifold representation. In addition, most local nonlinear techniques cannot deal with manifolds that are not isometric to the Euclidean space. This is most significantly revealed by the results of our experiments with the helix and twinpeaks datasets. It 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).
Fifth, local techniques for dimensionality reduction suffer from folding [21]. 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 is not necessarily equal over the manifold (i.e., because the prior 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., [76,93,115].
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.

7.2. *Global techniques*

Our discussion on the results of global techniques for dimensionality reduction is subdivided into three parts. First, we discuss the results of the 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. Third, we discuss the results of the three techniques that optimize nonconvex objective functions, viz., autoencoders, LLC, and manifold charting.
For the first part, we remark that global techniques for dimensionality reduction that employ neighborhood graphs, such as Isomap and MVU, are subject to many of the weaknesses that are mentioned in subsection 7.1, because the construction of the neighborhood graph is susceptible to (1) the curse of dimensionality, (2) overfitting, and the (3) presence of outliers. The results of our experiments on natural datasets show that global techniques for dimensionality reduction based on neighborhood graphs are often outperformed by PCA. However, the results also reveal that Isomap and MVU outperform local techniques such as LLE on natural datasets. The relatively good results of Isomap may be explained from the presence of a large number of 'short-circuits' in the neighborhood graph, as a result of which the neighborhood graph shows large resem-

blance to a random network. In such a network, the distances through the graph are strongly correlated to the (squared) Euclidean distances [11], and Isomap thus reduces to MDS and PCA. The relatively good results of MVU may be explained by the maximization of the variance in the low-dimensional data representation that MVU performs. This maximization causes originally distant points to be far apart in the low-dimensional data representation as well, even when the constraints on the maximization do not follow the local structure of the manifold.

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 certain complex 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 [45], 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 using semidefinite programming [46,69]. 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 local techniques (e.g., when a Gaussian kernel with a small value of σ is employed). The poor performance of diffusion maps in our experiments supports this claim.

For the third part, we remark that techniques that optimize nonconvex objective functions, such as autoencoders, LLC, and manifold charting, 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, these techniques 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 [30]. 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). For autoencoders, the presence of local optima in the objective function has largely been overcome by the pretraining of the network using RBMs. 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. Hence, the inferior performance of autoencoders on the ORL dataset is the result of the 10,304 dimensions in this dataset. This limitation of autoencoders may be addressed by preprocessing the data using PCA.

[11] In the Erdös-Rényi random network [38] on a Euclidean space, the shortest path between two points through the network is correlated with the Euclidean distance between the two points [85].

7.3. *Main weaknesses*

Taken together, the results of our experiments indicate that nonlinear techniques for dimensionality reduction do not yet clearly outperform traditional PCA. This result agrees with the results of studies reported in the literature. On selected datasets, nonlinear techniques for dimensionality reduction outperform linear techniques [81,107], but nonlinear techniques perform poorly on various other natural datasets [47,58,59,74]. In particular, our results establish two main weaknesses of the popular local techniques for dimensionality reduction: (1) the susceptibility to the curse of dimensionality and (2) the problems in finding the smallest eigenvalues in an eigenproblem.

From the first weakness, we may infer that a requirement for future techniques for dimensionality reduction is that they do not rely 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 [14]. However, the poor results of LLC and manifold charting in our experiments do not support this suggestion. The main added value of local techniques for dimensionality reduction techniques is that they can be applied in domains where information on the global structure of the data is not available, such as in sensor localization problems [118]. The second weakness leads to the suggestion that in dimensionality reduction, it is not so important which property of the data is retained in the low-dimensional data representation, but it is more important how well it is retained. Therefore, we suggest that the focus of the research community should shift towards the development of techniques that have objective functions that can be optimized well in practice. The strong per-

formance of autoencoders indicates that these objective functions are not necessarily convex.

8. Conclusions

The paper presents 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, not yet capable of outperforming traditional PCA. In the future, we foresee the development of new nonlinear techniques for dimensionality reduction that do not rely on local properties of the data manifold. In addition, we foresee a shift in focus towards the development of nonlocal techniques for dimensionality reduction with objective functions that can be optimized well, such as (Kernel) PCA and autoencoders.

Acknowledgements

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

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 [64] and their probabilistic extension GTM [17], because we consider these techniques to be clustering techniques. Techniques for Independent Component Analysis [11] are not included in our review, because they were mainly designed for blind-source separation. Linear Discriminant Analysis [40] and Generalized Discriminant Analysis [8] 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 [101], principal curves [25], kernel maps [102], conformal eigenmaps [98], Geodesic Nullspace Analysis [21], variants of multidimensional scaling [3,32,39,53,79], techniques that (similarly to LLC and manifold charting) globally align a mixture of linear models [91,94,114], and linear variants of LLE [49,65], Laplacian Eigenmaps [50], and LTSA [122].

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

[9] 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.

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

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

[12] 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.

[13] 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.

[14] 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.

[15] 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.

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

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

[18] C.M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[19] 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.

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

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

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

[23] 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.

[24] 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.

[25] 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.

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

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

[28] 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.

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

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

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

[36] 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.

[37] 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.

[38] P. Erdös and A. Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.

[39] 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.

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

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

[42] 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.

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

[44] 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.

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

[46] 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.

[47] 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.

[48] J. Ham, 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.

[49] 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.

[50] 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.

[51] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang. Face recognition using Laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

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

[53] 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.

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

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

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

[57] 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.

[58] 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.

[59] 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.

[60] 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.

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

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

[63] 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.

[64] T. Kohonen. *Self-organization and associative memory*. Springer-Verlag, Berlin, Germany, 1988.

[65] 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.

[66] J.B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1–27, 1964.

[67] 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.

[68] 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.

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

[70] 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.

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

[72] 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.

[73] 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.

[74] 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.

[75] 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.

[76] 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.

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

[78] 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.

[79] 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.

[80] 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.

[81] 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.

[82] 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.

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

[84] 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.

[85] M. Penrose. *Random Geometric Graphs*. Oxford University Press, 2003.

[86] 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.

[87] 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.

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

[89] 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.

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

[91] 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.

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

[93] 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.

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

[95] 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.

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

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

[98] 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.

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

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

[101] C. Spearman. General intelligence objectively determined and measured. *American Journal of Psychology*, 15:201–293, 1904.

[102] 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.

[103] G.A. Tagaris, W. Richter, S.G. Kim, G. Pellizzer, P. Andersen, K. Ugurbil, and A.P. Georgopoulos. Functional magnetic resonance imaging of mental rotation and memory scanning: a multidimensional scaling analysis of brain activation patterns. *Brain Res.*, 26(2-3):106–12, 1998.

[104] 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.

[105] 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.

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

[107] 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.

[108] 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.

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

[110] 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.

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

[112] M.S. Venkatarajan and W. Braun. New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physicalchemical properties. *Journal of Molecular Modeling*, 7(12):445–453, 2004.

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

[114] 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.

[115] 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.

[116] 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.

[117] 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.

[118] 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 (to appear), 2007.

[119] 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.

[120] 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.

[121] 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.

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

[123] 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.

[124] 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.