A Survey of Dimension Reduction Methods for High-dimensional Data Analysis and Visualization*

Daniel Engel[1], Lars Hüttenberger[1], and Bernd Hamann[2]

1 **Computer Graphics and HCI Group**
University of Kaiserslautern, Germany
{d_engel, l_huette}@cs.uni-kl.de

3 **Institute for Data Analysis and Visualization & CS Department**
University of California, Davis, USA
hamann@cs.ucdavis.edu

Abstract

Dimension reduction is commonly defined as the process of mapping high-dimensional data to a lower-dimensional embedding. Applications of dimension reduction include, but are not limited to, filtering, compression, regression, classification, feature analysis, and visualization. We review methods that compute a point-based visual representation of high-dimensional data sets to aid in exploratory data analysis. The aim is not to be exhaustive but to provide an overview of basic approaches, as well as to review select state-of-the-art methods. Our survey paper is an introduction to dimension reduction from a visualization point of view. Subsequently, a comparison of state-of-the-art methods outlines relations and shared research foci.

1998 ACM Subject Classification G.3 Multivariate Statistics, I.2.6 Learning, G.1.2 Approximation

Keywords and phrases high-dimensional, multivariate data, dimension reduction, manifold learning

Digital Object Identifier 10.4230/OASIcs.VLUDS.2011.135

1 Introduction

Contemporary simulation and experimental data acquisition technologies enable scientists and engineers to generate massive amounts of data. Thereby, more and more application domains are producing progressively larger and inherently more complex (multivariate) data sets. These data sets are collections of samples that consist of multiple measured (or simulated) observations of a variable set. Expressed in a space that requires many degrees of freedom, multivariate data present severe problems for data analysis and especially for visualization. Visualization is the integral part of exploratory data analysis, the first stage of data analysis where the goal is to make sense of the data before proceeding with more goal-directed modeling and analyses. Since human perception (and output devices) is limited to three-dimensional space, the challenge of visualizing multivariate data is converting the data to a space of lower dimensionality that is depictable and comprehensible to the user while preserving as much information as possible. This process is called dimension reduction and visualization of multivariate data is one of its traditional applications.

This survey reviews methods of dimension reduction that focus on visualizing multivariate data. That is, they are suitable for a depictable target space. Our aim is not to be exhaustive

* This work was supported by the German Science Foundation (DFG).

Proceedings of IRTG 1131 – Visualization of Large and Unstructured Data Sets Workshop 2011.
Editors: Christoph Garth, Ariane Middel, Hans Hagen; pp. 135–149

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

but to provide an overview of basic approaches, as well as to review select state-of-the-art methods. Thereby, we describe the mathematical concepts and ideas underlying the algorithms. Implementation details, although important, are not discussed. The reader should be aware that there are numerous dimension reduction methods that focus on the various aspects of data analysis. For example, methods for feature reconstruction or classification are closely related to those considered here, but are not discussed because their focus is not visualization. The reader will find that, due to its long history, there are numerous surveys on dimension reduction. For example, authors focus on a specific subset of techniques [23] or investigations [25], provide a broad overview [4], or historical background [16]. This survey provides an introduction to the concepts of visualizing high-dimensional data using dimension reduction and reviews select state-of-the-art methods that share this focus.

The remainder of the paper is structured as follows. Section 2 represents the core of the survey - a detailed introduction to the concepts of dimension reduction. After a formal problem statement is given, we divide the basic approaches in two classes: projection (Section 2.1) and manifold learning (Section 2.2). We also provide a taxonomy for these methods that can act as a classifier for which data the methods are most suited. Section 3 reviews two recently developed but fundamentally different approaches to non-linear multivariate data visualization and offers a qualitative comparison between them. The object of this investigation is to infer common trends between different concepts of dimension reduction. Finally, concluding remarks are provided in Section 4.

2 Dimension reduction

Methods for dimension reduction compute a mapping from high- to low-dimensional space. The formal problem setting can be described as follows. Let $X \in \mathbb{R}^{(n\times m)}$, a set of n points in m-dimensional data space, and two metric distance (or dissimilarity) functions, $\delta_m : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ and $\delta_t : \mathbb{R}^t \times \mathbb{R}^t \to \mathbb{R}$, over data space $\mathbb{R}^m$ and target space $\mathbb{R}^t$ respectively, with $m, t \in \mathbb{N}^*, t \ll m$, be given. A mapping function ϕ that maps the m-dimensional data points ($x_i \in X$) to t-dimensional target points ($y_i \in Y$), i.e.,

$$\begin{aligned} \phi : \quad \mathbb{R}^m &\to \mathbb{R}^t \\ x_i &\mapsto y_i, \text{ for } \; 1 \leq i \leq n, \end{aligned} \tag{1}$$

is defined s.t. ϕ "faithfully" approximates pairwise distance relationships of X by those of $Y \in \mathbb{R}^{(n\times t)}$, thereby mapping close (similar) points in data space to equally close points in target space, i.e., $\delta_m(x_i, x_j) \approx \delta_t(y_i, y_j)$, for $1 \leq i, j \leq n$. In particular, an adequate mapping is designed to ensure that remote data points are mapped to remote target points.

Since the target space usually has lower degrees of freedom than those required to model distance relationships in multi-dimensional space, the mapping ϕ adheres to an inherent error that is to be minimized by its definition. Thereby, ϕ is commonly defined to minimize the least squares error $\mathcal{E}_\phi$:

$$\mathcal{E}_\phi = \sum_{1\leq i,j\leq n} W_{i,j} \; (\delta_m(x_i, x_j) - \delta_t(y_i, y_j))^2, \; \textit{for } W \in \mathbb{R}^{(n\times n)}, \tag{2}$$

where W is a weight matrix that can be used to define the importance of certain data relationships or dimensions. For example, this may be used to disregard outliers by defining $W_{i,j} = 1/\delta_m(x_i, x_j)$ (for $\delta_m(x_i, x_j) \neq 0$).

Formally, the above definitions require both data and target distance functions to be metric. That is, both functions must adhere to the properties of positive definiteness,

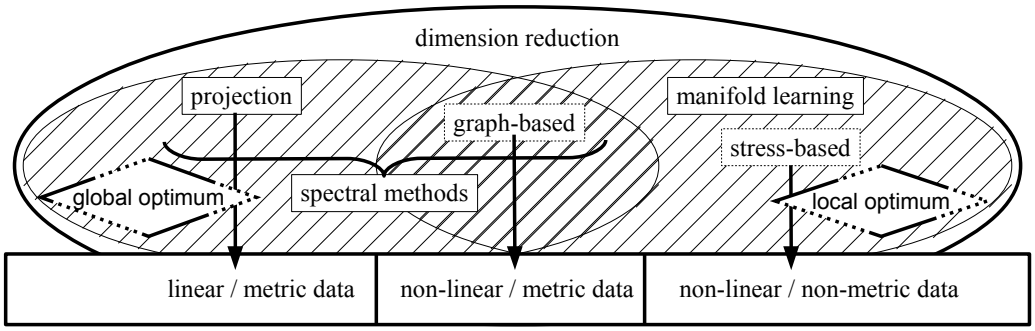

Figure 1 Concepts of dimension reduction.

symmetry, and the triangular inequality. Based on human perception, the most intuitive distance metric is the Euclidean distance, $L_2(p,p') = \sum_{1\leq i\leq q}\sqrt{(p_i - {p'}_i)^2}$ for $p,p' \in \mathbb{R}^q$. Due to its intuitiveness, the Euclidean distance is often chosen as the metric for the target space, $\delta_t = L_2$. However, the distance (or dissimilarity) measure of the application domain, δ_m, is in most cases not Euclidean and may in some cases not even be metric. For example, psychometric dissimilarities can be non-metric. In practice, this formal prerequisite can be relaxed since even an optimal mapping is, at any rate, an approximation of multivariate relationships.

In the following, we review and discuss several algorithms that realize a suitable mapping ϕ as defined above. We divide them into two basic approaches of the following underlying principal geometric ideas. If the data lie within a linear subspace of lower dimensionality, then they can be re-expressed by a linear basis transformation without loss of information. These bases can be ordered according to their contribution to the mapping error $\mathcal{E}_\phi$ and the t bases are used that minimize this error. However, if the data are non-linear and lie on an unknown manifold of lower dimensionality, then distance relationships along this manifold can be learned in an unsupervised manner and used for data mapping.

A careful taxonomy of the methods considered here is formulated in the following and illustrated in Figure 1. Methods that are solely based on linear inner product transformations are defined as projection techniques, while those that are able to ascertain distance relationships in a non-linear data structure are defined as manifold learning techniques. These techniques can be further grouped in two basic approaches. Focusing on metric data spaces, the first approach is graph-based. These methods model the data as a graph and utilize optimizations of graph theory to learn manifold distances in data space. The second approach is stress-based and focuses on the embedding directly, i.e., learning the mapping that minimizes the mapping error in target space. These methods are based on iterative optimizations of the mapping error (stress) and can learn the embedding of non-metric distances.

2.1 Projection-based Methods

Projective techniques display multi-dimensional data by projecting points onto a lower-dimensional space such that distance relationships between points in the projection space reflect specific relationships between the data points in multi-dimensional space. Since these relationships may be too complex to be completely conveyed in lower-dimensional space, projections (and all mappings considered here) are in general ambiguous. We define

a projection by the use of a projection in the geometric sense - projecting the data based on a (linear) inner product transformation. The geometric idea behind this approach is to express the data by a set of "condensed" variables that approximately model the (unknown) underlying factors and reduce redundancies. The two main approaches are to project based on variance or inner product relations and both are, in an Euclidean setting, interchangeable.

2.1.1 Principal Components Analysis (PCA)

As one of the first dimension reduction techniques discussed in the literature, Principal Components Analysis (PCA) [20] conveys distance relationships of the data by orthogonally projecting the data on a linear subspace of target dimensionality. In this specific subspace, the orthogonally projected data have maximal variance. Thereby, PCA defines a "faithful" approximation as one that captures the data's variance in an optimal way. It has been shown [13] that by the maximization of variance, PCA also minimizes the least squares error (2) for Euclidean distances in data and target space, $\delta_m = \delta_t = L_2$, under the constraint of orthogonally projecting the data:

$$\mathcal{E}_{\mathrm{PCA}} = \sum_{1 \leq i,j \leq n} (L_2(x_i, x_j) - L_2(y_i, y_j))^2. \tag{3}$$

Remarkably, PCA achieves this through a computationally efficient linear transformation. The resulting projection is a genuine view that does not distort the data. The only major drawback of PCA is that, due to its linear nature, it does not capture non-linear data well.

For the following considerations, we assume without loss of generality that $X \in \mathbb{R}^{(n \times m)}$ is centered, i.e., the mean of all given data points has been subtracted from all data points. The PCA projection is defined as

$$\begin{aligned} \mathrm{PCA}: \quad \mathbb{R}^m &\to \mathbb{R}^t \\ x_i &\mapsto x_i \, \widehat{\Gamma}, \ \textit{for } 1 \leq i \leq n, \end{aligned} \tag{4}$$

with $\widehat{\Gamma} = (\gamma^{(1)}, ..., \gamma^{(t)}) \in \mathbb{R}^{(m \times t)}$ being the matrix storing columnwise the eigenvectors of the corresponding t largest eigenvalues of the data's covariance matrix $S = n^{-1} X^T X$. The largest eigenvalue of S, λ_1, holds the variance of the data orthogonally projected in the direction of γ_1. $\widehat{\Gamma}$, storing the t mutually orthogonal vectors in which directions the data have the largest variance, define a partial orthonormal basis in data space $\mathbb{R}^m$. The orthogonal projection onto the corresponding rank-t subspace in $\mathbb{R}^m$ is defined by $\widehat{X} = X \, \widehat{\Gamma}\widehat{\Gamma}^T$. Thereby, $\widehat{X} \in \mathbb{R}^{(n \times m)}$ is the best rank-t-approximation of X (under L_2). Using the basis $\widehat{\Gamma}$, data points x_i are projected onto this subspace such that $\widehat{x}_i = \sum_{1 \leq k \leq t} \gamma^{(k)} \mathrm{PCA}(x_i)_k$, for $1 \leq i \leq n$.

Besides its broad applicability to visualization, PCA may be used for many more tasks. For example, a prominent gap in the eigenvalue spectra gives an upper bound for the intrinsic dimensionality of the data. Therefore, it is often used for filtering Gaussian noise or for reducing data size and computation time. PCA is a well-established technique with an extensive history. As such, many variants exist and more information can be found, for example, in [11] or [17].

2.1.2 Metric Multidimensional Scaling (MDS)

Metric Multidimensional Scaling (MDS) [28], also known as classical MDS, is a well-established approach that uses projection to map high-dimensional points to a linear subspace of lower dimensionality. The technique is often motivated by its goal to preserve pairwise distances

in this mapping. As such, metric MDS defines a faithful approximation as one that captures pairwise distance relationships in an optimal way; more precisely, inner product relations.

Metric MDS finds an optimal (least squares) linear fit to the given pairwise distances, assuming the distance used is metric. If Euclidean distances are given, $\delta = L_2$, metric MDS is equivalent to PCA up to scaling and rotation. However, metric MDS finds the best linear fit to any metric dissimilarities. This makes the technique more flexible to use compared to PCA. Its performance is also independent of data dimensionality, however, the method scales poorly with the number of data points.

By the method's design, the mapping error preserves inner product relations:

$$\mathcal{E}_{\text{mMDS}} = \sum_{1 \leq i,j \leq n} (x_i x_j^T - y_i y_j^T)^2. \tag{5}$$

Let a matrix of pairwise metric distances (or dissimilarities), $(\Delta)_{i,j} = \delta_{i,j}$, be given. From these metric distances, the data's Gram matrix of inner products is given by $G = HAH^T$, where $A = -1/2\delta_{i,j}^2$ and H is a centering matrix. The complete eigendecomposition of G requires $O(n^3)$ time which is, in most cases, too expensive for practical problems. However, variants of the method achieve an approximation in $O(nlogn)$ time based on a divide and conquer approach of the eigendecomposition [30]. In addition, increasingly faster solvers are being developed [14].

Metric MDS is defined as

$$\begin{aligned} \text{mMDS}: \quad \mathbb{R}^m &\to \mathbb{R}^t \\ x_i &\mapsto \widehat{\Gamma}_i \, \widehat{\Lambda}, \ for\ 1 \leq i \leq n \end{aligned} \tag{6}$$

with $\widehat{\Gamma} = (\gamma^{(1)}, ..., \gamma^{(t)}) \in \mathbb{R}^{(n \times t)}$ being the matrix storing columnwise the eigenvectors of the corresponding t largest eigenvalues of the Gram matrix of inner products, $G = XX^T$, $G_{i,j} = x_i x_j^T$. $\widehat{\Lambda}$ is the diagonal matrix storing the roots of the t largest eigenvalues of G, $\widehat{\Lambda} = diag(\sqrt{\lambda_1}, ..., \sqrt{\lambda_t})$.

Although metric MDS works in the inner product space, the geometric intuition behind the method is very similar to that of PCA. As such, points are projected into the linear subspace of largest variance. However, this subspace is defined by metric MDS based on the eigenvalue decomposition of an $n \times n$ matrix of inner products. The duality between PCA and MDS becomes clear when considering that G has the same rank and eigenvalues (up to a constant factor) as the covariance matrix $S = n^{-1}X^T X = Cov(X)$ and $G = n^{-1}Cov(X^T)$. Therefore, the Gram matrix is a covariance matrix in $\mathbb{R}^n$ that reflects the same principal relationships of the data as the covariance matrix in $\mathbb{R}^m$, although, expressed in a basis system that reflects linear combinations of data points (instead of dimensions). For more information on metric MDS, the reader is referred to [9] or [5].

Although being both powerful and flexible, Metric MDS leaves two questions unanswered: (1) What if the data are samples from a non-linear manifold and its proximity relationships are unknown? (2) What if these dissimilarities are not metric? In the following, we discuss the essential concepts that solve these two major issues. In particular, metric MDS has brought forth the variants Kernel PCA, Isomap, and non-metric MDS.

2.1.3 Kernel PCA

Kernel PCA [24] is considered a variant of PCA and metric MDS (due to their duality) that is capable of depicting non-linear data. Although distance relationships along a non-linear pattern are unknown, Kernel PCA is based on two assumptions that make the application

of (linear) PCA to non-linear data possible. The first assumption is that in the space of the data's underlying features, the data are linear. The second assumption is that there is a function that approximates the inner product of data points in this feature space. This function is called a kernel and the utilization of a non-linear kernel in a linear setting to capture non-linear data structure is known as the "kernel trick". Formally, this setting is described as follows. Let a kernel k be given that approximates inner product relations of non-linear data in their feature space, such that

$$\begin{aligned} k: \quad \mathbb{R}^m \times \mathbb{R}^m &\to \mathbb{R} \\ (x_i, x_j) &\mapsto \Phi(x_i)\Phi(x_j)^T, \ for\ 1 \leq i \leq n, \end{aligned} \tag{7}$$

where Φ is the mapping to feature space. Kernel PCA is defined as

$$\begin{aligned} \mathrm{K-PCA}: \quad \mathbb{R}^m &\to \mathbb{R}^t \\ x_i &\mapsto \widehat{\Gamma}_i\, \widehat{\Lambda}, \ for\ 1 \leq i \leq n \end{aligned} \tag{8}$$

with $\widehat{\Gamma} = (\gamma^{(1)}, ..., \gamma^{(t)}) \in \mathbb{R}^{(n \times t)}$ being the matrix storing columnwise the eigenvectors of the corresponding t largest eigenvalues of the Gram matrix of inner products *in feature space*, $G^{(k)}_{i,j} = k(x_i, x_j)$. $\widehat{\Lambda}$ is the diagonal matrix storing the roots of the t largest eigenvalues of $G^{(k)}$, $\widehat{\Lambda} = diag(\sqrt{\lambda_1}, ..., \sqrt{\lambda_t})$.

Thereby, Kernel PCA computes the eigenvectors of the covariance matrix of the data in feature space. Although this space, as well as the data coordinates therein, is unknown, the kernel maps to the data's Gram matrix of inner products in feature space. Based on the assumption of the correctness of a kernel k, the eigendecomposition of $G^{(k)}$ captures the non-linear relationships in the data by maximizing variance in feature space. As such, Kernel PCA can be viewed as a generalization of the method metric (classical) MDS by substituting the utilization of Euclidean dot products to generalized dot products.

It is not surprising that the bottleneck of Kernel PCA is finding the "right" kernel. Since distance relationships along the possibly non-linear sub-structures of the data are, in general, a-priori unknown, the definition of a suitable kernel requires explicit knowledge about the data. If this knowledge is not given, methods are better suited that determine distance relationships along non-linear data structures in an unsupervised data-driven manner. This is the concept of manifold learning.

2.2 Manifold Learning

Projection-based methods work well for data that fit approximately to a linear subspace. When this is not the case, the hope for dimension reduction is that the data follow at least a non-linear pattern, i.e., they lie on a manifold. The methods considered in this section are able to learn (and depict) proximity relationships of data points on (non-linear) manifolds in an unsupervised manner. While mappings from projection-based methods can be described by linear transformations that capture known proximity relationships, this is not the case for manifold learning techniques. In particular, these techniques abstract from Euclidean distance relationships and capture distances along a manifold. Figure 2 illustrates the difference between projection and manifold learning based mappings.

There are two distinct approaches to learn unknown proximity relationships. These approaches are based on the data being of metric or non-metric dissimilarity. To model metric distances on a manifold, graph-based techniques are often used that retrieve local distance relationships in a data-driven way and project the data based on these metric distances. However, there are various applications that require the display of non-metric

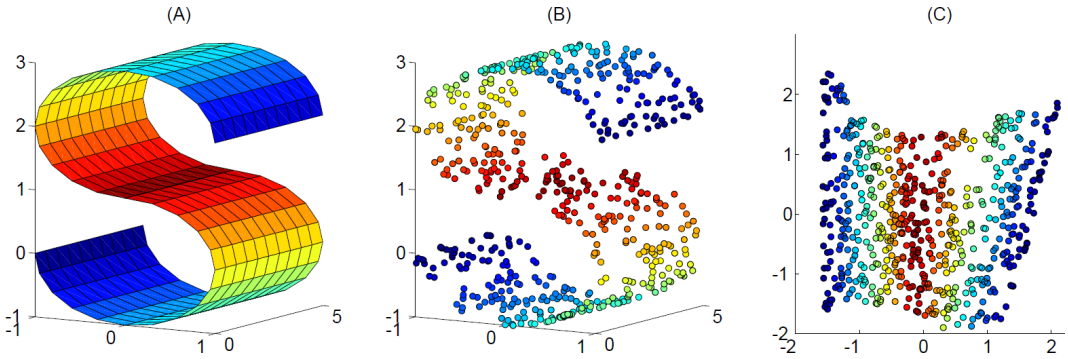

Figure 2 In this example, data sampled from a non-linear three-dimensional manifold (A) are mapped by a projection-based method (B) and by a manifold learning technique (C). In (B), the projection of the data is a linear transformation that optimally captures Euclidean distances. In (C), distance relationships along the manifold are captured by a non-linear mapping of the data. This figure derives from [21].

dissimilarity relationships. This problem cannot be solved by graph-based methods but only through a direct minimization of the mapping error in the embedding. This leads to the optimization of a non-convex stress function. Consequently, stress-based methods are prone to local minima and often slow convergence.

Graph-based methods can be divided into two classes: global and local modeling. Global approaches first learn proximity relationships on a locally low-dimensional sub-manifold and, second, depict these relationships using, for example, projection-based methods like metric MDS. Local graph-based modeling follows a divide and conquer approach. The idea is to divide the data into small groups and to solve this embedding locally. Local systems are then "pieced together" based on overlapping or fixation points. Although the projection step finds the global optimum for the embedding, the initial retrieval of distance relationships is based on optimization problems such as shortest path problems, least squares fits, or semidefinite programming. In this regard, graph-based methods are also prone to local minima or higher computational cost.

2.2.1 Non-metric MDS

The ability of metric MDS to map data relationships from a dissimilarity matrix is based on the key assumption that dissimilarities are approximate squared metric distances. As for all spectral methods, this allows for the computation of a global optimal projection. However, this also limits its application and prohibits non-metric scenarios, for example, stemming from psychometric research where metric postulates do not hold. Instead of this eigendecomposition approach, the idea of non-metric Multidimensional Scaling is to directly minimize the mapping error (2) with respect to a given non-metric dissimilarity matrix and possibly some weighting thereof. Unfortunately, due to the non-metric nature, the resulting stress function is non-convex and optimization thereof is prone to local minima.

For a perfect projection, it holds that $\mathcal{E}_{\mathrm{MDS}}(\Delta, Y, W) = 0$, where Δ is the input, Y the output, and W an optional (arbitrary) weighting. One way to approximate the solution is through a steepest descent approach, for example, with the Euler method [1]. Thereby, a step-wise iteration towards zero, where the $(k+1)^{th}$ iteration has the form $Y^{(k+1)} = Y^{(k)} + \alpha^{(k)} \nabla \mathcal{E}_{\mathrm{MDS}}(\Delta, Y^{(k)}, W)$, converges to a local minimum. The step size $\alpha^{(k)}$ can be constant or can be computed by means of line search. A disadvantage of this

method is its slow convergence near a minimum. An approach to avoid this is to use higher-level, gradient-descent-type methods, for example, Newton's methods [12]. These methods converge more quickly at a higher computational cost.

The exact embedding of non-metric dissimilarities in a metric target space is impossible. However, in non-metric MDS, the rank-order of dissimilarities is assumed to contain the most significant information, and the main goal of the approach is to depict the rank-order in its output configuration. A well-known approach to non-metric MDS is the Shepard-Kruskal algorithm [15]. At its core is a twofold optimization process that optimizes the goodness of fit with regard to the non-metric input. First, an optimal monotonic transformation of the non-metric dissimilarities to metric distances is found that preserves the rank-order of non-metric inputs. After the optimization of the rank-order distances, the output configuration is further improved iteratively, balancing both stress and monotonicity.

MDS is in all respects a hard non-convex optimization problem. Using a good initialization is therefore important. Numerous variants of MDS exist and many other methods are closely related, like Sammon's mapping [22]. Especially multi-level approaches have substantially increased performance [10]. For an overview, reference [2] is helpful.

2.2.2 Isomap

Instead of learning the embedding directly in target space, Isomap [27] attempts to explicitly model non-linear proximity relationships in terms of geodesic distances. As such, it can be viewed as a variant of metric MDS to model non-linear data using its (metric) geodesic distances. In order to retrieve these distances, a global graph-based optimization approach is utilized.

Geodesic distances are learned by linearly approximating the non-linear manifold. Thereby, a network of undirected neighborhood graphs is constructed in which each data point is a node and has edges to its neighbors that are weighted by the points' dissimilarity. The weights represent the local approximation of geodesic distances on the manifold. From these graphs, a square geodesic distance matrix is computed which is used for the metric MDS projection. The essential steps can be summarized as follows:

1. For each data point x_i compute an undirected k-neighborhood graph based on the k points of smallest dissimilarity to x_i and assign this dissimilarity as the edge's weight.[1]
2. The $(n \times n)$ matrix of geodesic distances $\tilde{\Delta}$ is found by computing the shortest paths through the network of neighborhood graphs.[2]
3. Project the data using $\tilde{\Delta}$ and metric MDS, as described in Section 2.1.2.

One problem of Isomap is that after double-centering of the geodesic distances, the Gram matrix of inner products is not guaranteed to be positive semidefinite. One variant that solves this issue is Maximum Variance Unfolding (MVU) [29]. The underlying idea behind MVU is to unfold the manifold under the constraint that local distances between neighboring points are preserved. This is optimized with respect to maximum variance.

Note that the lower-dimensional embedding of geodesic distances by Isomap involves the eigendecomposition of a dense $(n \times n)$ matrix. Like with metric MDS, this leads to significant computational effort. Further variants exist that tackle this problem, for example, by integrating a local approach [25].

[1] Often a threshold is used to model disconnected sub-manifolds.
[2] This can be computed, for example, using Dijkstra's algorithm[7].

2.2.3 Locally Linear Embedding (LLE)

In contrast to modeling a manifold by global geodesic distance relationships, LLE [21] models the manifold by extracting its local intrinsic geometry. Thereby, LLE follows a local graph-based approach. The basic idea of LLE is based on the linear approximation of all data points (in complex non-linear structures) by a convex linear combination of its neighborhood. Formally, this assumption can be described by the following equation which has to hold for all data points $x_i \in X$ and their surrounding neighbors N_i,

$$x_i = \sum_{x_j \in N_i} W_{i,j} x_j \tag{9}$$

with $0 \leq W_{i,j} \leq 1$, $\sum_{x_j \in N_i} W_{i,j} = 1$, and $W_{i,i} = 0$, for $1 \leq i, j \leq n$. The local intrinsic geometry has the appealing property that it stays unchanged under transformations like translation, rotation or scaling. Hence, the local linear relationships of points in data space directly define the intrinsic geometry for the output points to target space. The weights $W_{i,j}$ are approximated by solving a least squares problem based on a k-neighborhood graph. In contrast to Isomap, LLE models nearest neighbors by directed graphs which leads to a more suitable approximation. With these local relationships, LLE constructs a set of global equations for the projection to target space. The method is summarized as follows:

1. For each data point x_i, compute the k neighbors N_i that are nearest to x_i with respect to the distance function δ_m.
2. Compute the weights $W_{i,j}$ that minimize the equation $\sum_{i=1}^{n} |x_i - \sum_{j=1}^{n} W_{i,j} x_j|^2$ and satisfy the constraints, $W_{i,j} = 0$ if x_j is not a neighbor of x_i, $W_{i,i} = 0$ and $\sum_{j=1}^{n} W_{i,j} = 1$ for all $1 \leq i \leq n$.
3. Compute the output points y_i that minimize the equation $\sum_{i=1}^{n} |y_i - \sum_{j=1}^{n} W_{i,j} y_j|^2$.

As with Isomap, the data projection step is done by solving an $n \times n$ eigenproblem that is based on the global weight matrix W. Due to the locality of LLE, this weight matrix is sparse which leads to a significant advantage in terms of computation speed. The projection is defined by the bottom $t+1$ [3] eigenvectors of the matrix $(I-W)^T(I-W)$ that can be computed without a full matrix diagonalization [6].

3 Current State of Research

Having introduced the main concepts of dimension reduction that can be utilized for visualization, this section reviews more recent work. We compare the two dominant and distinct approaches to non-linear dimensionality reduction, namely graph- and stress-based methods. We review one representative paper of each approach, each one being both state-of-the-art and comparable in terms of similar goals and assumptions. Because both methods stem from a different background, it is likely that they have been developed independently from each other. Our goal is to infer common trends, relations, and solutions of these independent research streams that both solve the problem of finding optimal lower-dimensional embeddings for non-linear multivariate data.

[3] The bottom eigenvector is a unit vector and is discarded to enforce the constraint that the embeddings have zero mean. Here, bottom refers to the ordering imposed by largest to lowest corresponding eigenvalues.

3.1 Piecewise Laplacian-based Projection (PLP)

Similar to LLE, PLP [19] makes the assumption that every data point x_i can be approximated by a convex combination of its neighbors $x_j \in N_i$ based on weights $W_{i,j}$. While LLE finds those weights through optimization, PLP uses pre-defined weights according to:

$$W_{i,j} = \frac{1}{\delta_m(x_i, x_j)} \Bigg/ \sum_{x_k \in N_i} \frac{1}{\delta_m(x_i, x_k)} \tag{10}$$

with δ_m being the metric distance function of the data space. Due to those pre-defined weights, the projection has no unique solution. Therefore, a set of global control points is added on a divide-and-conquer basis to solve this problem. PLP divides the data in smaller subsets, each contributing a number of control points that are globally projected to preserve global relationships among subsets. This procedure allows for corrections based on user input, which makes this method interactive. PLP is defined by the following steps:

1. Separate X into $s = \sqrt{n}$ different samples S_j for $1 \leq j \leq s$. [4]
2. For each sample S_j define the neighborhoods $N_i \subseteq S_j$ for each $x_i \in S_j$ and a set of control points $C_j \subseteq S_j$.
3. Globally project all control points $C = C_1 \cup \ldots \cup C_s$ from $\mathbb{R}^m$ to $\mathbb{R}^t$.
4. For each sample S_j, construct and solve a separate local linear system but based only on the local variables C_j and the neighborhoods $N_i \subseteq S_j$.
5. Present the resulting projected data points Y to the user who can redefine the neighborhoods. Based on the new neighborhoods, repeat the method from step three.

Paulovich et. al. [19] set the number of neighbors k to ten and the number of control points in each sample S_i to $\sqrt{|S_i|}$ [5], which ensures that the number of control points of a sample corresponds to its sample size. The set of global control points C can be embedded by any appropriate mapping, for example, Paulovich et. al. use the stress-based *Force Scheme* [26].

After the local linear systems have been solved for each sample, the user can interact with the projected data set through its representation as a k-nearest neighbor graph and adjust neighborhoods or samples by simply moving data points within the embedding. Due to the used multi-level approach, only the linear systems of samples have to be recomputed in which the neighborhoods have been changed. Consequently, PLP can learn the embedding of large high-dimensional data sets in a semi-supervised manner.

If data do not come in a tagged format, partitioning them into samples is done by clustering methods. On the one hand, the multi-level approach leads to significantly smaller total computational cost since the linear systems, which are solved at step four, are now smaller. On the other hand, important global features may be missed due to this approach. Since the control points (randomly chosen) set the frame for global relation of local patches, there is no guaranty that global features can be preserved in all cases. However, the novel option of user interaction likely compensates for this scenario.

3.2 Multigrid Multidimensional Scaling (MG-MDS)

As a variant of multidimensional scaling, MG-MDS [3] is based on the direct optimization of the *weighted* mapping error as a stress function $\mathcal{E}_\phi$ given by (2), although, the method

[4] Note that $\sqrt{n}$ is an upper bound for the number of groups in a data set of size n [18]. More sophisticated estimation schemes may also be used.

[5] Note that the total number of control points amounts to $n^{3/4}$

requires distances to be metric. In contrast to PLP, weights in MG-MDS can be arbitrary and do not represent a convex combination. The basic idea is to re-state the problem of finding $\phi = \arg\min_\phi \mathcal{E}(\phi(X); \Delta, W)$ with respect to the gradient-descent-type method as a problem of finding ϕ with $\nabla \mathcal{E}(\phi(X); \Delta, W) = 0$ and to embed this problem into a multigrid approach, through which substantial performance improvements can be achieved. A simplified view of MG-MDS is that the re-stated problem is first solved for a *core*, a small subset of all data points. But instead of a one-step projection of the remaining data points, each of the remaining data points is projected separately, in a step-by-step projection. Hence, to project one of the remaining data points, not only the projected core but all the so far projected points are used. Obviously, this increases the computational cost, but approximation errors, which occur during a big, one-step projection, can be counteracted.

MG-MDS constructs a hierarchy of grids from the data set X such that for $X = \{x_{i_1}, \ldots, x_{i_n}\}$, the hierarchy is defined by choosing x_{i_n} randomly chosen from X and picking x_{i_k} from $k = n-1$ to $k = 1$ so that the following equation holds:

$$x_{i_k} = \arg\max_{x \in X} \min_{l=k+1,\ldots,n} \delta(x, x_{i_l}) \text{ for } 1 \leq k \leq n-1.$$

In other words, x_{i_k} is a data point with maximal distance to all data points with higher hierarchy level. Each grid level k holds the set of all data points of the hierarchy level equal or higher than k, i.e., $X_k = \{x_{i_k}, \ldots, x_{i_n}\}$. To transfer between grid levels, multi-grid approaches offer *restriction* P_k^{k+1} and *interpolation matrices* P_k^{k-1}, such that $N_{k-1} = P_k^{k+1} N_k$ and $N_{k-1} = P_k^{k-1} N_k$. Additionally, a corresponding stress function $\mathcal{E}_\mathbf{k}$, based on X_k, Δ_k, and W_k, determines the error.

Choosing a maximal grid level R, MG-MDS is summarized by the following steps:

1. If $r = R$, solve $\min_{X_R} s_R(X_R, T_R)$ by using Euler's or Newton's methods which are based on the gradient of s_R.
2. Otherwise, go from grid r to $r+1$, using P_r^{r+1} and ∇s_r, changing also W and Δ.
3. Apply recursively the MG-MDS method to X_{r+1} and use P_{r+1}^r to get from grid $r+1$ back to grid r.
4. During each movement from one grid to the next, a relaxation using an SMACOF-type method [2] is needed to smooth the errors which occur during the movement.

Note that the existence of P_r^{r+1} and P_r^{r-1} for all $R \leq r \leq n$ is a weaker form of the convex neighborhood assumption of LLE or PLP. P_r^{r+1} and P_r^{r-1} can be found if the data points in grid level r belong to the convex combination of the points in $r+1$, and $r-1$ respectively.

3.3 Comparison

Both approaches of stress optimization and spectral decomposition solve the problem of visualizing non-linear multivariate data. However, they achieve this in completely different ways. A comparison between them is difficult because stress optimization solves the much harder problem of embedding non-metric distance relationships, while spectral methods are restricted to metric ones. Nevertheless, such a comparison has the potential of inferring valuable insights on what generic ideas and solutions help with the problem at hand. For this, MG-MDS was chosen as a representative over numerous other state-of-the-art methods that follow the stress optimization approach, because its unique advantages are also restricted to the input being metric dissimilarities. Here, the relations between both methods are qualitatively discussed and their suitability for different scenarios is assessed. This comparison is based on the crucial factors that may delimit their application: online behavior, parametrization, and computational cost.

Assumptions

PLP (like LLE) makes the assumption that each data point can be represented by a convex combination of its nearest neighbors. Thereby, the data are approximated by a set of linear patches. MG-MDS, on the other hand, is based on the minimization of the stress function through gradient methods and uses only a weak form of this assumption.[6] Hence, for data sets where the convex combination property does not hold, no suitable neighborhood can be found, or when the computation of the neighborhoods is too costly, MG-MDS is better suited to solve the problem.

Relations

PLP and MG-MDS are similar in the sense that they do not use the whole data set at once. Instead, they use a small subset for the costly core projection[7] and then project the rest of the data with a faster method which uses the core projection. This is a definite trend and saves a significant amount of time. However, this approach requires the data set to be of sufficient size in order for a good initial core projection to be possible. Hence, for smaller data sets, methods like LLE are preferable.

Online behavior

Considering online scenarios where an existing solution is to be adjusted with regard to new data, PLP is better suited for such purpose than MG-MDS. [8] With PLP, new data points do not chance the global projection but only the local linear system within the sample which can be computed with comparably low computational cost. In this regard, MG-MDS has to be redone for grid levels in which the new data points occur. Although, most likely, the maximal grid level $r = R$ stays unchanged, the overall computational cost is higher. Both methods, however, are based on the dimension of the data points. For online scenarios where, instead of new points, new dimensions are added to the already existing data, methods solely based on local intrinsic geometry (like LLE) are advantageous. In any case, local methods are preferable for online scenarios.

Parameterization

When little is known of a data set, an extensive list of parameters often represents a burden for the analyst. However, in a visual analytics environment, the ability to tweak the mapping based on knowledge and interaction is a definite advantage. Additionally, expert knowledge is utilized that simplifies the problem of embedding. PLP requires knowledge of the "right" clustering technique, the number of clusters in the data set, the number of control points, as well as knowledge for defining the "right" neighborhood. This requires the user to have a good initial assessment on the data's structure and their global features. Therefore, when no expert is available, MG-MDS is the safer choice because it requires less user parameters (maximal grid level and core gradient method). On the other hand, PLP's ability to iteratively refine the mapping based on user interaction makes the method more suitable for visual exploration and allows one to infer this knowledge over time.

---

[6] In MG-MDS, the convex combination is only a sufficient condition but does not have to hold for all data points and also does not include neighborhood relations.

[7] Either the projection of the control points or the calculation at the maximal grid level $r = R$.

[8] It is assumed that the new data points are not taken as control points.

Computational cost

Another limiting factor for the suitability of dimension reduction methods with regard to many applications are their computational costs. The cost for computing a neighborhood graph depends on the form the data are given in. In case of a distance matrix, the cost to construct a k-neighborhood graph amounts to $O(k \cdot n)$ for each of the n data points. If the data are given as an m-dimensional point set, the computational cost to define the neighborhood for each data point is $O(m \cdot n^2)$. Although, in some cases, space partitioning data structures like *K-D trees* [8] can reduce this cost to $O(n \cdot \log n)$, their suitability for higher-dimensional spaces is an open research question. We therefore denote the cost to compute the distance matrix by $O(\text{Distance})$, while the cost to compute a k-neighborhood graph is denoted by $O(\text{Neighbors})$. With these considerations in mind, the computational costs of these two methods are:

PLP $O(\text{Distances}) + O(n^{3/2}) + O(n^{9/4}) + O(s{\cdot}\text{Sample}_{n/s})$ with $s = \sqrt{n}$ being the number of samples and $O(\text{Sample}_{n/s})$ the computational cost for each sample with size n/s. For this, a uniform size over all samples is assumed. $O(\text{Sample}_k)$ is defined as $O(\text{Sample}_k) = O(k^{3/2}) + O(k^{3/2}) +$ the computational costs to solve a linear system of size $k \times (k + \sqrt{k})$, with $\sqrt{k}$ being the number of control points in the sample. The two other terms are the cost to find the samples using a clustering method and the global projection of all $n^{3/4}$ control points using any $O(n^3)$ projection method.

MG-MDS $O((n - R)n^2) + O(2Rn^2) + O(\text{Distances})$, with R being the maximal grid level. The first term is for the core projection of grid level $r = R$ using Euler's Method. The second term is for movement between these r many grid levels. By using more complex methods than Euler's method, the computational cost increases while the value of the stress function decreases. Based on the same considerations as those made by PLP, it seems that $R = n - \sqrt{n}$ is a fair initial guess for the maximal grid level.

Note that these terms are all upper bounds. The actual computational cost can be far smaller. For example, in PLP, much effort is saved since the computation of the samples and control points uses the clustering results for the computation of the neighborhoods. Also, data may already come in a gridded or tagged form that these algorithms can use and take advantage of.

4 Conclusion

Research on dimension reduction continues at a rapid pace. This survey provides an introduction to the main concepts of dimension reduction for visualization: from linear data projection to graph- and stress-based manifold learning. Although being non-exhaustive, the comparison of state-of-the-art methods that follow the graph- or stress-based approach shows that no single method can be preferred over another. On the contrary, the effectiveness of state-of-the-art methods mainly depends on the data and application. However, the comparison also shows that there are similar research directions. At present, especially multi-level approaches show great potential and form one of the dominant research directions in both graph- and stress-based manifold learning.

Motivation for ongoing work includes manifolds of complex non-linear geometry, more flexible and interactive embeddings, better encoding of information, and scalability to data sets of peta-scale sizes. We believe that only through the incorporation of multiple concepts from different research fields, can methods for dimension reduction keep pace with future problems. Due to the increasing complexity of high-dimensional data sets, a two-dimensional

target space is not sufficient for the embedding. There is a major gap to close to the concepts of information visualization that may be used to gain additional degrees of freedom in an embedding. Furthermore, these concepts can help with better interpretability and interactivity in adjusting both view and model of the lower-dimensional mapping. Data analysis also requires the incorporation of level-of-detail approaches for data abstraction and new concepts for visual verification that evaluate the error and ambiguity of a mapping. As we have discussed, the focus of state-of-the-art methods has already changed towards semi-supervised learning that incorporates user knowledge into mapping and visualization, thereby allowing an effective visual exploration. It is likely that these knowledge-based algorithms will continue to evolve and gain in importance.

Acknowledgements The authors gratefully acknowledge the support of the German Science Foundation (DFG) for funding this research (grant #1131).

References

1 U.M. Ascher and L.R. Petzhold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, 1998. 314 pages.

2 I. Borg and P.J.F. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 2005.

3 M. M. Bronstein, A. M. Bronstein, R. Kimmel, and I. Yavneh. Multigrid multidimensional scaling. *Numerical Linear Algebra with Applications (NLAA)*, 13:149–171, March-April 2006.

4 Christopher J. C. Burges. Dimension reduction: A guided tour. *Foundations and Trends in Machine Learning*, 2(4), 2010.

5 T. Cox and M. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 1994.

6 James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

7 Edsger. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

8 J.H. Friedman, J.L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):290–226, 1977.

9 W. Härdle and L. Simar. *Applied Multivariate Statistical Analysis*. Springer, 2nd edition edition, 2007.

10 Stephen Ingram, Tamara Munzner, and Marc Olano. Glimmer: Multilevel mds on the gpu. *IEEE Transactions on Visualization and Computer Graphics*, 15:249–261, 2009.

11 I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, 2002.

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

13 Y. Koren and L. Carmel. Robust linear dimensionality reduction. *Visualization and Computer Graphics, IEEE Transactions on*, 10(4):459 –470, jul. 2004.

14 Ioannis Koutis, Gary L. Miller, and Richard Peng. Approaching optimality for solving sdd linear systems. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 235–244, Washington, DC, USA, 2010. IEEE Computer Society.

15 J. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29, 1964.

16 John Aldo Lee and Michel Verleysen. Unsupervised dimensionality reduction: Overview and recent advances. In *IJCNN*, pages 1–8. IEEE, 2010.

17 B. F. Manly. *Multivariate statistical methods: a primer*. Chapman & Hall, Ltd., London, UK, UK, 1986.

18 N.R. Pal and J.C. Bezdek. On cluster validity for the fuzzy c-means model. *IEEE TFS*, 3(3):370–379, 1995.

19 F.V. Paulovich, D.M. Eler, J. Poco, C.P. Botha, R. Minghim, and L.G. Nonato. Piece wise laplacian-based projection for interactive data exploration and organization. *Computer Graphics Forum*, 30(3):1091–1100, 2011.

20 K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(6):559–572, 1901.

21 Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

22 J. W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Trans. Comput.*, 18(5):401–409, 1969.

23 L. K. Saul, K. Q. Weinberger, J. H. Ham, F. Sha, and D. D. Lee. Spectral methods for dimensionality reduction. *Semisupervised Learning. MIT Press: Cambridge, MA*, 2006.

24 Bernhard Schölkopf, Alex J. Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

25 Vin De Silva and Joshua B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, 2003.

26 E. Tejada, R. Minghim, and L.G. Nonato. On improved projection techniques to support visual exploration of multidimensional data sets. *Information Visualization*, 2(4):218–231, 2003.

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

28 W.S. Torgerson. *Theory and methods of scaling*. Wiley, 1958.

29 Kilian Q. Weinberger and Lawrence K. Saul. An introduction to nonlinear dimensionality reduction by maximum variance unfolding. In *AAAI*. AAAI Press, 2006.

30 Tynia Yang, Jinze Liu, Leonard Mcmillan, and Wei Wang. A fast approximation to multidimensional scaling, by. In *Proceedings of the ECCV Workshop on Computation Intensive Methods for Computer Vision (CIMCV)*, 2006.