Local multidimensional scaling

Jarkko Venna and Samuel Kaski

Adaptive Informatics Research Centre

Helsinki University of Technology

P.O. Box 5400, FI-02015 TKK, Finland

Phone: +358 451 4335,+358 451 8203

Fax: +358 451 3277

{jarkko.venna, samuel.kaski}@tkk.fi

Acknowledgments

This work was supported by the Academy of Finland, decision numbers 79017 and 207467 and by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-202-506778.

Local multidimensional scaling

Abstract

In a visualization task, every nonlinear projection method needs to make a compromise between trustworthiness and continuity. In a trustworthy projection the visualized proximities hold in the original data as well, whereas a continuous projection visualizes all proximities of the original data. We show experimentally that one of the multidimensional scaling methods, curvilinear components analysis, is good at maximizing trustworthiness. We then extend it to focus on local proximities both in the input and output space, and to explicitly make a user-tunable parameterized compromise between trustworthiness and continuity. The new method compares favorably to alternative nonlinear projection methods.

Keywords: Information visualization, manifold extraction, multidimensional scaling (MDS), nonlinear dimensionality reduction, nonlinear projection, gene expression.

Symbol	Description
i, j	indices of data points
k	number of nearest neighbors
N	number of data points
$\mathbf{x}_i$	data point in the input space
$\mathbf{y}_i$	representation of point i in output space
$\mathbf{C}_x$	covariance matrix of the data
$d(\mathbf{x}_i, \mathbf{x}_j)$	distance in the input space
$d(\mathbf{y}_i, \mathbf{y}_j)$	distance in the output space
$\boldsymbol{\lambda}$	vector of eigenvalues
σ_y	radius of the area of influence
p_{ij}, q_{ij}	probability of i being a neighbor of j
$\mathbf{m}_i$	model vector
$\mathbf{r}_i$	location on the SOM grid
$U_{k(i)}$	the set of data points that are in the neighborhood of point i in the output space but not in the input space
$r(i, j)$	the rank of point j in the ordering according to the distance from i in the input space
$V_{k(i)}$	the set of data points that are in the neighborhood of point i in the input space but not in the output space
$\hat{r}(i, j)$	the rank of point j in the ordering according to the distance from i in the output space
λ	tradeoff parameter on local MDS

1 Introduction

In information visualization, one of the main tasks is to reduce the dimensionality of data to two or three, to visualize proximities within a data set. In general, it is not possible to reduce the dimensionality without losing some of the proximities in the process. Two kinds of errors can occur. First, data points originally farther away may enter the neighborhood of a sample in the projection. These errors decrease the *trustworthiness* of the visualization, as they create neighborhood relationships that are not present in the data. Second, data points that are originally in the neighborhood can be pushed farther away in the visualization. This stems from the *discontinuity* of the mapping. Because of the discontinuities some of the original neighborhood relationships are missing from the visualization. Each dimensionality reduction method necessarily makes a tradeoff between these two kinds of errors. This setting is analogous to the precision—recall tradeoff in information retrieval.

We have earlier (Kaski et al., 2003) argued that trustworthiness is often more important than continuity since the visualized proximities are particularly salient. It has turned out in our experiments (see Section 3.3) that one of the multidimensional scaling methods, curvilinear component analysis (CCA: Demartines & Hérault, 1997), is particularly good in this respect. It aims at preserving pairwise distances but not all of them; only distances between points close-by on the visualization are preserved. The formulation of neighborhoods in

the projection plane shares some motivation with the Self-Organizing Map (Kohonen, 2001). It would be even better to let the user decide about the compromise between trustworthiness and continuity, and in this work we will extend CCA to make a parameterized compromise between trustworthiness and continuity. We call the new method *local multidimensional scaling* (local MDS).

New methods for estimation of data manifolds or embeddings have been presented in recent years. So far the methods, isomap (Tenenbaum et al., 2000), locally linear embedding (LLE: Roweis & Saul, 2000), Laplacian eigenmap (Belkin & Niyogi, 2002a) and stochastic neighbor embedding (SNE: Hinton & Roweis, 2002), have not been compared in the task of *visualization* where the dimensionality of the representation is not selected based on the manifold but constrained by the display instead. We compare these methods with CCA and SOM, two older methods that are known to perform well in visualization, and with the new local MDS proposed here.

Many nonlinear projection methods operate on a distance matrix of the data. Typically Euclidean distance matrix has been used but there has recently been a trend, started by isomap, of creating new variants of older methods by replacing the Euclidean distance matrix with approximation of the geodesic distance matrix. In this paper we will, in addition to isomap, compare such variants of CCA and SNE to their Euclidean counterparts. The goal is to see whether the use of the geodesic distances enhances visualization performance.

2 Information visualization methods

2.1 Multidimensional scaling

We did not include traditional multidimensional scaling (MDS) to the comparisons, but a short description helps to understand the more recent methods below.

There are several different variants of MDS (Borg & Groenen, 1997), but they all have a common goal: to find a configuration of points that preserves the pairwise distance matrix. The simplest version is the linear MDS (Torgerson, 1952; Gower, 1966), also called classical scaling. The solution to Linear MDS can be found by solving an eigenvalue problem.

A slightly more complex version is metric MDS. Its cost function is

$$E = \sum_{ij} (d(\mathbf{x}_i, \mathbf{x}_j) - d(\mathbf{y}_i, \mathbf{y}_j))^2, \tag{1}$$

where $d(\mathbf{x}_i, \mathbf{x}_j)$ is the distance in the input space and $d(\mathbf{y}_i, \mathbf{y}_j)$ the distance in the output space, between the representations $\mathbf{y}_i$ and $\mathbf{y}_j$ of the points i and j. The cost function is minimized with respect to the representations $\mathbf{y}_i$.

Most versions of MDS use a variant of this cost function. Sammon's mapping (Sammon Jr., 1969) gives small distances a larger weight. In non-metric MDS (Kruskal, 1964) the distances are modified by a monotonic function. There exists a huge number of different variants, but all have basically the same form.

2.2 Principal component analysis (PCA)

The goal of PCA (Hotelling, 1933) is to find linear projections that maximally preserve the variance in the data. The projection directions can be found by solving the eigenvalue problem

$$\mathbf{C}_x \mathbf{a} = \lambda \mathbf{a} \ , \tag{2}$$

where $\mathbf{C}_x$ is the covariance matrix of the data $\mathbf{x}$. The data points $\mathbf{x}_i$ can then be visualized by projecting them with

$$\mathbf{y}_i = \mathbf{A}\mathbf{x}_i, \tag{3}$$

where $\mathbf{A}$ is the matrix containing the eigenvectors corresponding to the two or three largest eigenvalues, and $\mathbf{y}_i$ is the obtained low-dimensional representation of $\mathbf{x}_i$.

PCA is very closely related to linear MDS. Gower (1966) has shown that when the dimensionality of the solutions is the same, the projection of the original data to the PCA subspace equals the configuration of points found by linear MDS that is calculated from the Euclidean distance matrix of the data. Thus the cost function of PCA tries to preserve the squared distances between data points.

2.3 Locally linear embedding (LLE)

The LLE algorithm developed by Roweis and Saul (2000) is based on the assumption that we can make a locally linear approximation of the data manifold. It assumes that a point and its neighbors lie in or close to a locally linear subspace on the manifold. The geometry of

this subspace can be captured by calculating the linear coefficients that reconstruct each data point from its neighbors. Here the neighbors are the k nearest neighbors of the data point. The reconstruction error is defined as

$$E(\mathbf{W}) = \sum_i \|\mathbf{x}_i - \sum_j W_{ij}\mathbf{x}_j\|^2. \tag{4}$$

To find the optimal weight matrix $\mathbf{W}$ the reconstruction error is minimized subject to the constraints that $W_{ij} = 0$ if i and j are not neighbors, and $\sum_j W_{ij} = 1$.

For visualization we want to reduce the dimensionality of the data to two or three. To achieve this we have to solve another optimization problem,

$$E(\mathbf{Y}) = \sum_i \|\mathbf{y}_i - \sum_j \mathbf{W}_{ij}\mathbf{y}_j\|^2 \ , \tag{5}$$

for $\mathbf{y}_i$, the low-dimensional representation of the data point i. The problem can be solved by finding the $p+1$ smallest eigenvalues of the matrix $(\mathbf{I}-\mathbf{W})^T(\mathbf{I}-\mathbf{W})$ (details in Roweis & Saul, 2000), where p is the dimensionality of the output. The smallest eigenvalue corresponds to a constant eigenvector and the next p give the coordinates of the data points within the manifold space.

The LLE implementation at http://www.cs.toronto.edu/∼roweis/lle/ was used in the experiments.

2.4 Laplacian eigenmap

The Laplacian eigenmap algorithm of Belkin and Niyogi (2002a) is similar to the LLE algorithm. The first step is to form the k-nearest-neighbor graph. Each data point is a vertex in the graph. There is

an undirected edge from point i to point j if j is among the k nearest neighbors of i. The definition of a neighbor differs from that used in LLE in that the neighborhood relation here is symmetric. If the data point i is a neighbor of j then j is also always a neighbor of i.

After the graph has been formed the edges have to be given weights. The simple method of assigning $W_{ij} = 1$ if the points i and j are neighbors and zero otherwise has been found to work well in practice (Belkin & Niyogi, 2002b).

The configuration of points in the low-dimensional space can be found by solving the generalized eigenvalue problem

$$\mathbf{L}\mathbf{y} = \lambda \mathbf{D}\mathbf{y}, \tag{6}$$

where $\mathbf{D}$ is the diagonal matrix with elements $D_{ii} = \sum_j W_{ij}$, and $\mathbf{L} = \mathbf{D} - \mathbf{W}$. The embedding of the data points is given by the eigenvectors having the p smallest eigenvalues, after discarding the smallest (always zero) eigenvalue.

2.5 Isomap

The isomap (Tenenbaum et al., 2000) is a variant of linear MDS. It finds a configuration of points that matches the given distance matrix. The difference from traditional MDS is in how the distances are defined. Isomap uses geodesic distances instead of direct pairwise distances. The geodesic distances are approximated with the shortest path distances calculated along the k-nearest-neighbor graph. The graph is defined in the same way as in the Laplacian eigenmap, ex-

cept that the weights of the edges are set to the Euclidean distances between the connected points.

The actual embedding of points is found by standard linear MDS, applied to the shortest-path distance matrix. It has been shown (Bernstein et al., 2000) that this algorithm is asymptotically able to recover certain types of manifolds.

The isomap implementation available at http://isomap.stanford.edu/ was used in the experiments.

2.6 Curvilinear component analysis (CCA)

CCA by Demartines and Hérault (1997) is also a variant of MDS. The starting point is a random initialization of points in the reduced-dimensional output space, and a pairwise distance matrix between the original data points. It differs from the standard formulation of MDS in that it concentrates on preserving the distances of points that are proximate in the *output space*. The cost function measures preservation of the original pairwise distances, weighted by a coefficient F that depends on the distance between the points in the output space:

$$E = \frac{1}{2} \sum_i \sum_{j \neq i} (d(\mathbf{x}_i, \mathbf{x}_j) - d(\mathbf{y}_i, \mathbf{y}_j))^2 F(d(\mathbf{y}_i, \mathbf{y}_j), \sigma_y) \ . \tag{7}$$

The coefficient F is usually defined as an area of influence around a data point in the output space:

$$F(d(\mathbf{y}_i, \mathbf{y}_j), \sigma_y)) = \begin{cases} 1 & \text{if } d(\mathbf{y}_i, \mathbf{y}_j) \leq \sigma_y \\ 0 & \text{if } d(\mathbf{y}_i, \mathbf{y}_j) > \sigma_y \ . \end{cases} \tag{8}$$

The cost function is optimized using a form of stochastic gradient descent algorithm. In the beginning of optimization the radius of the

area of influence, σ_y, is kept large enough to cover all or at least most of the data points. During the optimization it is slowly decreased to zero.

An extension of CCA, curvilinear distance analysis (CDA), was recently introduced by Lee et al. (2000, 2004). The main idea of CDA is to replace the Euclidean distances in the original space with geodesic distances in the same manner as in the isomap algorithm. Otherwise the algorithm stays the same.

2.7 Stochastic Neighbor embedding (SNE)

The SNE algorithm developed by Hinton and Roweis (2002) does not try to preserve pairwise distances as such, but instead *probabilities* of points being neighbors. The pairwise distances in the input and output space are used to define probability distributions on how probable it is that the point i is a neighbor of point j. The goal then is to find a configuration of points in the output space where those probabilities are the same, for each pair of points, as in the input space.

More formally, the probability p_{ij} of the point i being a neighbor of point j in the input space is defined to be

$$p_{ij} = \frac{\exp\left(-d(\mathbf{x}_i, \mathbf{x}_j)\right)}{\sum_{k \neq i} \exp\left(-d(\mathbf{x}_i, \mathbf{x}_k)\right)}, \tag{9}$$

where $d(\mathbf{x}_i, \mathbf{x}_j)$ is the pairwise distance between the data points. In this paper $d(\mathbf{x}_i, \mathbf{x}_j)$ is either the Euclidean distance between the data points, or the geodesic distance that is calculated in the isomap. The version of SNE that uses geodesic distances will be referred to as SNEG

to distinguish between the two variants.

Similarly, the probability of the point i being a neighbor of point j in the output space is defined to be

$$q_{ij} = \frac{\exp\left(-\|\mathbf{y}_i - \mathbf{y}_j\|^2\right)}{\sum_{k \neq i} \exp\left(-\|\mathbf{y}_i - \mathbf{y}_k\|^2\right)}. \tag{10}$$

The configuration of points $\mathbf{y}_i$ that minimizes the Kullback-Leibler divergence between the probability distributions in the input and output spaces is the solution for the problem. The cost function is thus

$$E = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{11}$$

In the experiments we used the SNE implementation that is part of the Matlab package available at http://www.kyb.tuebingen.mpg.de/bs/people/zien/Mbed/. It uses conjugate gradients for optimizing the cost function.

2.8 The self-organizing map (SOM)

The SOM (Kohonen, 2001) consists of a regular grid of *units*. Each unit contains a model vector $\mathbf{m}_i \in \mathbb{R}^n$, where n is the dimensionality of the data. The sequential SOM algorithm iterates two steps. First, for data point $x(t)$ chosen randomly at iteration step $t = 0, 1, 2, \ldots$, the best matching model vector $\mathbf{m}_{c(t)}$ is sought using the equation

$$c(t) = \operatorname*{argmin}_j \{d(\mathbf{x}(t), \mathbf{m}_j)\}, \tag{12}$$

where $d(\mathbf{x}(t), \mathbf{m}_j)$ is the distance between the data point $\mathbf{x}(t)$ and the model vector $\mathbf{m}_j$. When the best matching model vector has been

found, the model vectors are updated with

$$\mathbf{m}_j(t+1) = \mathbf{m}_j(t) + h_{c(t),j}(t)[\mathbf{x}(t) - \mathbf{m}_j(t)]. \tag{13}$$

The function $h_{c(t),j}(t) = h(\|\mathbf{r}_{c(t)} - \mathbf{r}_j\|; t)$, where $\mathbf{r}_j$ and $\mathbf{r}_{c(t)}$ are the location vectors on the SOM grid for the units, is the neighborhood function. It makes the map ordered.

The visualizations produced by the SOM differ from the other methods presented here. Instead of each data point having its own location, on the SOM each data point is placed at the location of the best matching unit on the fixed SOM grid. Typically SOMs are visualized using the U-matrix (Ultsch, 1993, see Figure 3) or its variants. A gray-shade code is used to display distances between neighboring units. The light areas contain units that are mutually more similar than on the dark areas. Technically, space is added in between each pair of SOM units and shaded according to the distance between their model vectors. The shade of the map units is proportional to the median of distances to the neighboring units.

On a SOM display, the similarity can be defined simply as the distance on the display plane. This measure does not, however, take into account the density of the model vectors that is visualized by the U-matrix. Hence, we have used distances along minimal paths on the map lattice, with weights equal to the distances between the model vectors. On light areas, such distances are shorter and on dark areas they are longer.

3 Comparison of the methods

We start by comparing the nonlinear projection methods on toy data sets to illustrate their properties in a visualization task, and on real world high-dimensional data sets. After this, we extend one of the methods, the best performing (CCA), to local MDS in the next session.

3.1 Data Sets

Thick S-curve. A simple data set having a folded low-dimensional manifold, a two-dimensional S-shaped curve in a three-dimensional space, was constructed as follows. First, the data was uniformly sampled from a two-dimensional S-shaped sheet. Then, to give the manifold a thickness, a spherical normally distributed displacement was added to each point. The data set consists of 1000 data points.

Clusters. The data set consists of six clusters that are located symmetrically in a three-dimensional space. Five of the clusters are spherical Gaussians and one is a two-dimensional S-shaped manifold. The data set has 1000 data points divided in equal proportions to the clusters.

This data set poses an interesting problem for the methods that use k-nearest neighbor information. On typical values of k the k-nearest neighbor graph separates into several unconnected sections. Because of this, it is impossible to utilize these methods directly. There are two possible solutions. We can either increase k to make the graph connected ($k > 63$ on this data set) or add tailored connections to

the graph. The downside of increasing the number of neighbors is that the shortest path distances become more like Euclidean distances and some of the possible benefits of using geodesic distances are lost. Adding tailored connections is not trivial either.

We suggest using a simple practical procedure as an alternative to increasing k. We first create the (symmetric) k-nearest-neighbor graph and then recursively connect two separate sections by finding the smallest Euclidean distance between any two points in different unconnected sections. This procedure adds as few and as short edges to the graph as possible, to make it connected.

We tested both approaches. On LLE using a small number of neighbors and adding connections to the graph was the best solution based on the measures introduced in Sec. 3.2. On Laplacian eigenmap the results were very similar for both approaches with an increase in k producing slightly better results. On the isomap increasing k to make the graph connected was clearly the better method on this data set. On CDA and SNEG keeping k small and adding edges to make the graph connected produced the best results. On each method we utilized the best approach.

Gene expression compendium. We used the large collection of human gene expression arrays collected by Segal et al. (Segal et al., 2004). (The normalized expression compendium is available from http://dags.stanford.edu/cancer.)

For visualization purposes we removed samples with missing values

from the data. First we removed genes that were missing from more than 300 arrays. Then we removed the arrays for which values were still missing. This resulted in a data set containing 1278 arrays and 1339 genes (dimensions).

This is a very hard data set to visualize. The data is very high dimensional and there do not seem to be any low dimensional manifold structures that the methods could take advantage of.

Mouse gene expression. The fourth data set is a collection of gene expression profiles from different mouse tissues (Su et al., 2002). Expression of over 13000 mouse genes had been measured in 45 tissues. We selected an extremely simple filtering method, similar to that originally used in (Su et al., 2002). Of the mouse genes clearly (average difference in Affymetrix chips, AD $>$ 200) expressed in at least one of the 45 tissues, a random sample of 1600 genes was selected for visualization. The variance in each tissue was normalized to unity. For more details of the data set and of preprocessing see the paper by Kaski et al. (2003).

3.2 Measuring trustworthiness and continuity of a visualization

We consider a projection onto a display *trustworthy* if the k closest neighbors of a point on the display are also neighbors in the original space. We will use the following trustworthiness measure to compare the different visualization methods, and to quantify the compromise

made by the new method. (See Kaski et al., 2003; Venna & Kaski, 2001, for details.)

Let N be the number of data samples and $r(i,j)$ be the rank of the data sample j in the ordering according to the distance from i in the original data space. Denote by $U_k(i)$ the set of those data samples that are in the neighborhood of size k of the sample i in the visualization display but not in the original data space. Our measure of trustworthiness of the visualization is

$$M_1(k) = 1 - \frac{2}{Nk(2N-3k-1)} \sum_{i=1}^{N} \sum_{j \in U_k(i)} (r(i,j) - k) \; . \qquad (14)$$

The errors caused by discontinuities may be quantified analogously to the errors in trustworthiness. Let $V_k(i)$ be the set of those data samples that are in the neighborhood of the data sample i in the original space but not in the visualization, and let $\hat{r}(i,j)$ be the rank of the data sample j in the ordering according to the distance from i in the visualization display. The effects of discontinuities of the projection are measured by

$$M_2(k) = 1 - \frac{2}{Nk(2N-3k-1)} \sum_{i=1}^{N} \sum_{j \in V_k(i)} (\hat{r}(i,j) - k) \; . \qquad (15)$$

In case of ties in rank ordering, all compatible rank orders are assumed equally likely. We calculate the best and worst case values for the error measures and report the average of them.

The worst attainable values of both measures may, at least in principle, vary with k, and were estimated in the results (Figs. 1 and 2) with random projections and with random neighborhoods.

3.3 Results

We compared CCA, SOM and the new nonlinear projection methods mentioned in the Introduction.

The methods having a number of neighbors parameter k were run several times with values of k going from 4 to 20. Methods that do not have a global optimum were run ten times on each data set, starting from a different random initialization each time. The SOM size was set such that the average number of data points in each unit was about 2.7 on the mouse data and 5 on the other data sets. The SOM neighborhood was decreased to one during the optimization. In each case the result with the best trustworthiness was selected.

Trustworthiness and continuity. When trying to get insights on a data point a human analyst usually looks at a handful (say 10) data points around it. Thus it is very important that the visualization preserves small neighborhoods well, that is, that the visualization is trustworthy. It is clear from Figures 1 and 2 that in terms of trustworthiness the CCA with either Euclidean distance or geodesic distance (CDA) is the best or second best method on all data sets. SOM has the best trustworthiness on the two bioinformatics data sets. On the gene expression compendium SNE with graph distances is among the best methods as well. SNE is very good at preserving the continuity of the original neighborhoods. Both variants were the best or among the best in this respect on all data sets except the gene expression compendium data set. SNE with Euclidean distances was not able to

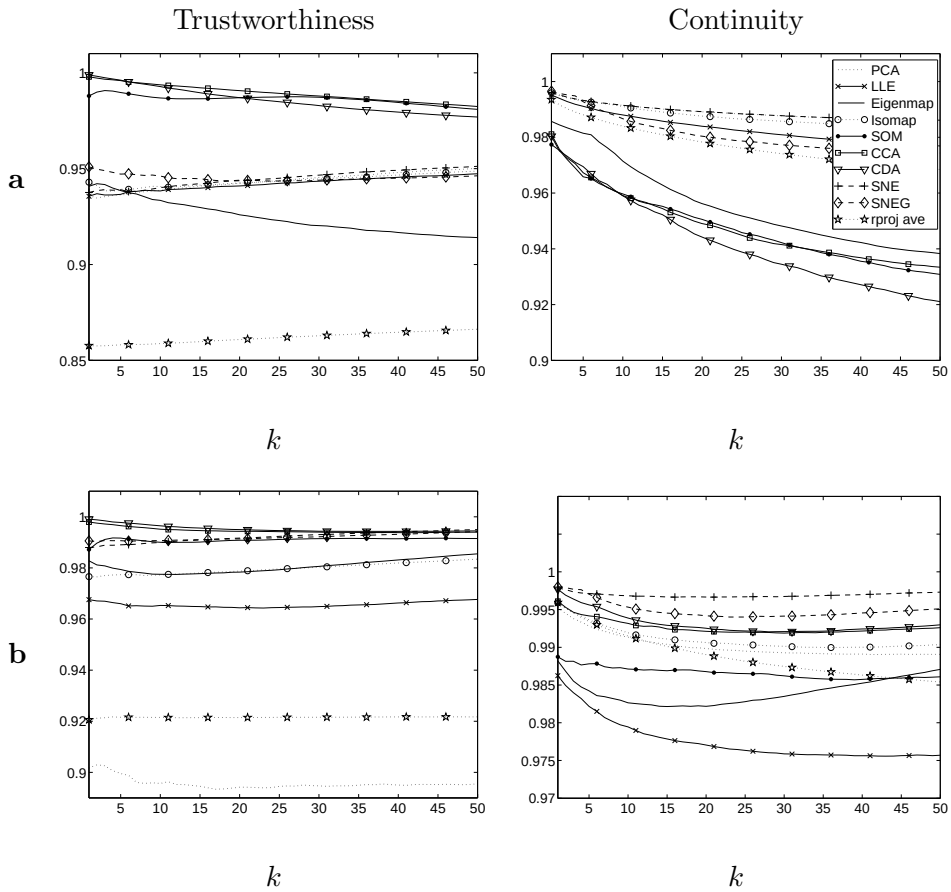

Figure 1: Trustworthiness and continuity of the mapping as a function of k, the size of the neighborhood used in measuring them. Small neighborhoods are the most important ones. **a**) Thick S-curve manifold, **b**) Cluster data set. Rproj is the average value of 100 linear random projections. The trustworthiness and continuity values of random neighborhoods are approximately 0.5. PCA: Principal component analysis, LLE: locally linear embedding, Eigenmap: Laplacian eigenmap, CCA: Curvilinear component analysis, CDA: CCA using geodesic distances, SOM: Self-organizing map, SNE: stochastic neighbor embedding, SNEG: SNE using geodesic distances.

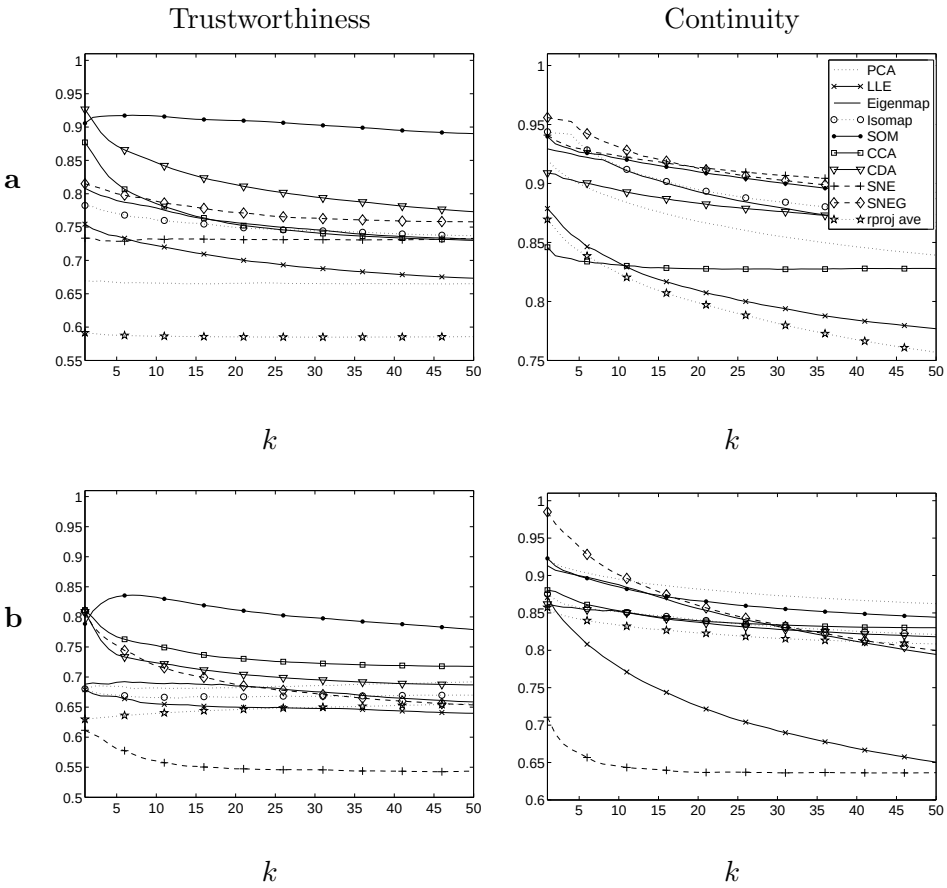

Figure 2: Trustworthiness and continuity of the mapping as a function of k, the size of the neighborhood used in measuring them. Small neighborhoods are the most important ones. **a**) Mouse gene expression, **b**) Gene expression compendium. Key: see Fig. 1.

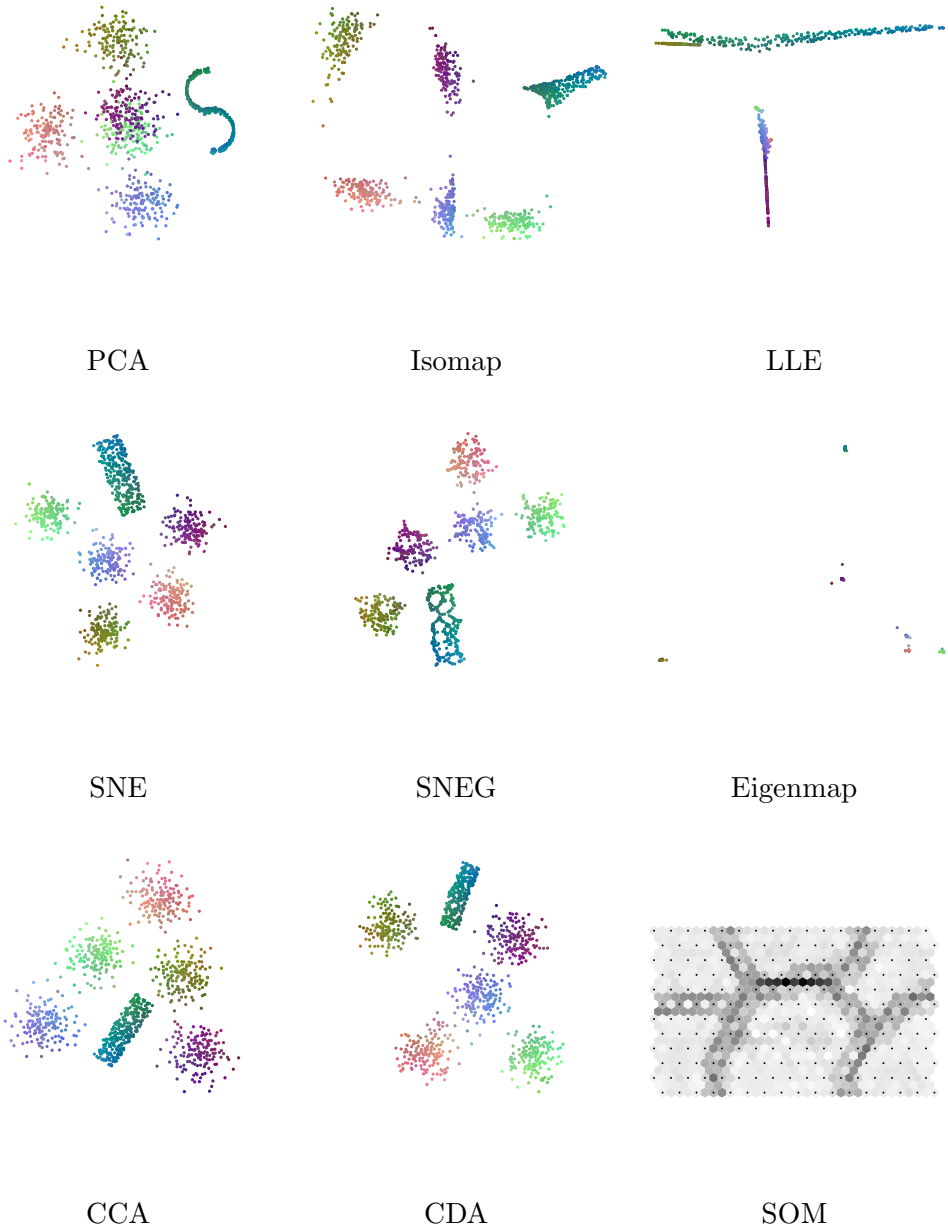

Figure 3: Sample visualizations produced with different methods on the cluster data set. Key for the methods: see Fig. 1.

produce any meaningful results on this data set, due to always getting stuck in a local minimum in the early stages of the optimization. PCA is also consistently quite good at preserving the continuity of original neighborhoods. Both versions of CCA are typically among the worst methods in this respect.

Euclidean vs. geodesic distance. There are three methods with two variants: the other uses Euclidean distances and the other a graph approximation of the geodesic distances. Which distance measure produces better results depends highly on the data set in question. Of the data sets used here the mouse gene expression data sets a clear case for the use of geodesic distances. On this data set the methods using geodesic distances outperformed their Euclidean counterparts with a clear margin in both trustworthiness and continuity. On the other hand, on the gene expression compendium the use of geodesic distances seems to give slightly worse results than Euclidean distances. The exception here is SNE which failed to produce any meaningful results with the Euclidean distance.

Quality of the visualizations. While the trustworthiness and continuity measures give a good idea on how well the methods preserve the local similarity structure in the data, they do not give the whole picture on the quality of the visualizations. Examples of the visualizations produced by the different methods on the cluster data set are shown in Figure 3. What one would expect to see in these visualizations is a set of six separate clusters, and hopefully the two-dimensional

(S-shaped manifold) structure of one of the clusters would also be evident. By looking at the results of LLE and Laplacian eigenmap, it is clear that the visualizations do not perform as well as expected. It is very hard to identify the separate clusters from the LLE visualization. Moreover, all clusters have been stretched to form mostly linear structures. On the visualization produced by Laplacian eigenmap the differences in the scales of distances are so large that it is not possible to discern any structure within the clusters. Only small blobs are visible.

The visualizations produced by SNEG illustrate an artifact that is typical for methods that utilize nearest neighbor information. The graph distances overestimate distances within the manifold and produce clear "holes." These are very clear on the SNEG visualization of the S-curve cluster. This effect can be lessened in two ways. The first is to select a method like CCA that relies mostly on local distances and the second is to increase the number of neighbors. The latter means has fixed the problem for isomap, where we had already used a very large number (k=67 in comparison to k=7 (CDA) and k=4 (SNEG)) of neighbors to make the graph connected.

4 Controlling the tradeoff: Local MDS

Every visualization method has to make a tradeoff between gaining a good trustworthiness and preserving the continuity of the mapping. Some methods like SOM and CCA are typically good at finding solu-

tions with a high trustworthiness, and others like SNE are very good at preserving the continuity. We propose a new method, *local MDS*, which is a derivative of CCA with the ability to control the tradeoff between trustworthiness and continuity of the mapping.

The CCA cost function (7) penalizes errors in preserving distances for points that are neighbors in the output space. This tends to produce solutions with a high trustworthiness. The basic idea of the extension is to add a similar term that penalizes errors for points that are proximate in the *input space*. The tradeoff between these two terms, tunable by a parameter λ, governs the tradeoff between trustworthiness and continuity. The cost function of local MDS is

$$\begin{aligned} E = \frac{1}{2}\sum_i \sum_{j\neq i} [(1-\lambda)(d(\mathbf{x}_i,\mathbf{x}_j) - d(\mathbf{y}_i,\mathbf{y}_j))^2 F(d(\mathbf{y}_i,\mathbf{y}_j),\sigma_i) + \\ + \lambda(d(\mathbf{x}_i,\mathbf{x}_j) - d(\mathbf{y}_i,\mathbf{y}_j))^2 F(d(\mathbf{x}_i,\mathbf{x}_j),\sigma_i)] \\ = \frac{1}{2}\sum_i \sum_{j\neq i} (d(\mathbf{x}_i,\mathbf{x}_j) - d(\mathbf{y}_i,\mathbf{y}_j))^2 \times \\ \times [(1-\lambda)F(d(\mathbf{y}_i,\mathbf{y}_j),\sigma_i) + \lambda F(d(\mathbf{x}_i,\mathbf{x}_j),\sigma_i)] \ , \quad (16) \end{aligned}$$

We optimize the cost function with the stochastic gradient descent introduced for CCA in (Demartines & Hérault, 1997). During the optimization the radius of the area of influence around data point i, σ_i, is slowly brought down. The final radius is set equal to the distance of the Kth nearest neighbor of the data point i in the original space. A small value of K will usually produce better values of trustworthiness on small neighborhoods but at the same time the effectivity of λ in controlling the tradeoff is reduced. The results shown here were pro-

duced with $K = 20$. Setting $\lambda = 0$ results in a normal CCA projection (with the difference that the end radius of the area of influence σ_i is larger than zero and different for each data point; for CCA the end radius of each data point is customarily reduced to zero).

We additionally tested a radius of influence which was the same for each data point and was brought down to zero at the end of optimization. The behavior was quite similar but a nonzero end neighborhood makes controlling of the compromise more robust (reduces fluctuations as a function of λ).

It is also possible to extend local MDS to use geodesic distances in a similar manner as has earlier been done with CCA: Simply replace the Euclidean distances with geodesic distances in the cost function. In the next section we will present results for both versions.

4.1 Results with local MDS

The number of neighbors used in the graph approximation of the geodesic distance was selected to be the same that produced the best result on CDA. For each value of λ we ran the algorithm five times, starting from different initial positions, and selected the result that produced the best value of (16).

The effect of varying λ is illustrated in Fig. 4 where trustworthiness and continuity (of a neighborhood of size 10) are plotted as a function of λ. When λ is increased there is an overall tendency for trustworthiness to decrease and continuity of original neighborhoods to increase. Typical behavior of local MDS on different neighborhood

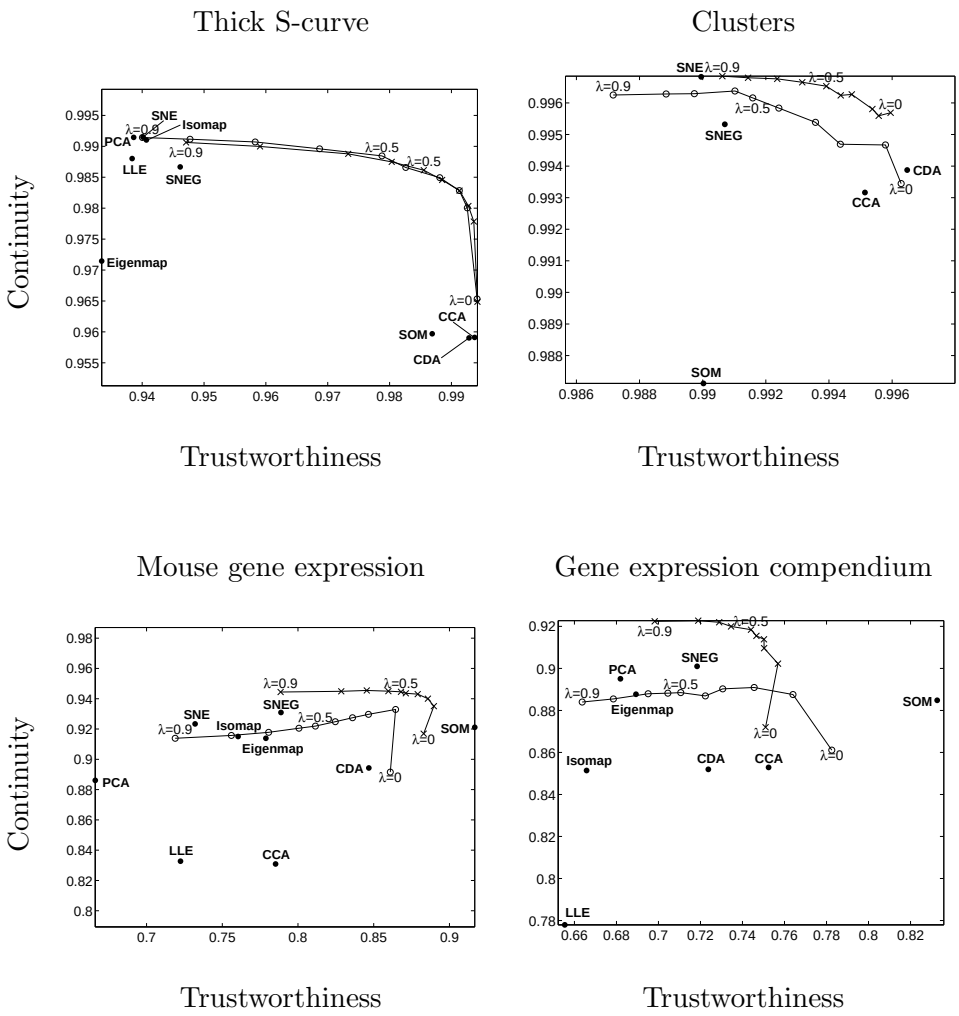

Figure 4: The relationship between trustworthiness and continuity of the mapping as a function of λ, for a neighborhood of size 10. *Line with open circles*: Local MDS, *line with* $\times s$: Local MDS with geodesic distances. Other methods are included (black dots with a name attached) for reference. Methods not shown are too far down or to the left to fit in the image.

sizes and with different values of λ is illustrated in Figure 5.

The performance of local MDS compares quite favorably to the other methods tested in this paper. Although the main point of local MDS is its ability to control the tradeoff between trustworthiness and continuity, it is able to produce results that are very close to or better than any of the other methods in either trustworthiness or continuity. The exception is the SOM on the bioinformatics data sets. Even there local MDS with geodesic distances outperforms all other methods.

Fig. 6 gives three examples of local MDS projections. Data that lies on the surface of a sphere is projected first with $\lambda = 0$ and then with $\lambda = 0.1$ and finally with $\lambda = 0.9$. When λ is zero local MDS splits the sphere open, roughly into two discs. When λ is increased, the edges, where continuity is violated the worst, get pulled closer together to minimize the number of neighborhoods that become split, and to reduce the distance between those neighborhoods that cannot be connected.

There are two peculiarities in the results of local MDS in Figure 4. First, when using Euclidean distances there is a point (usually at around $\lambda = 0.2 \ldots 0.5$) after which continuity of the mapping may start to decrease. This happens because the second part of the cost function does not optimize continuity directly, but only indirectly. If λ is too large, the unfolding effect of the first part of the cost function may not be enough to keep the projection from folding on top of itself. This is evident in the Figure 4 where on the mouse gene expression data continuity first increases sharply and then starts to decline. Thus,

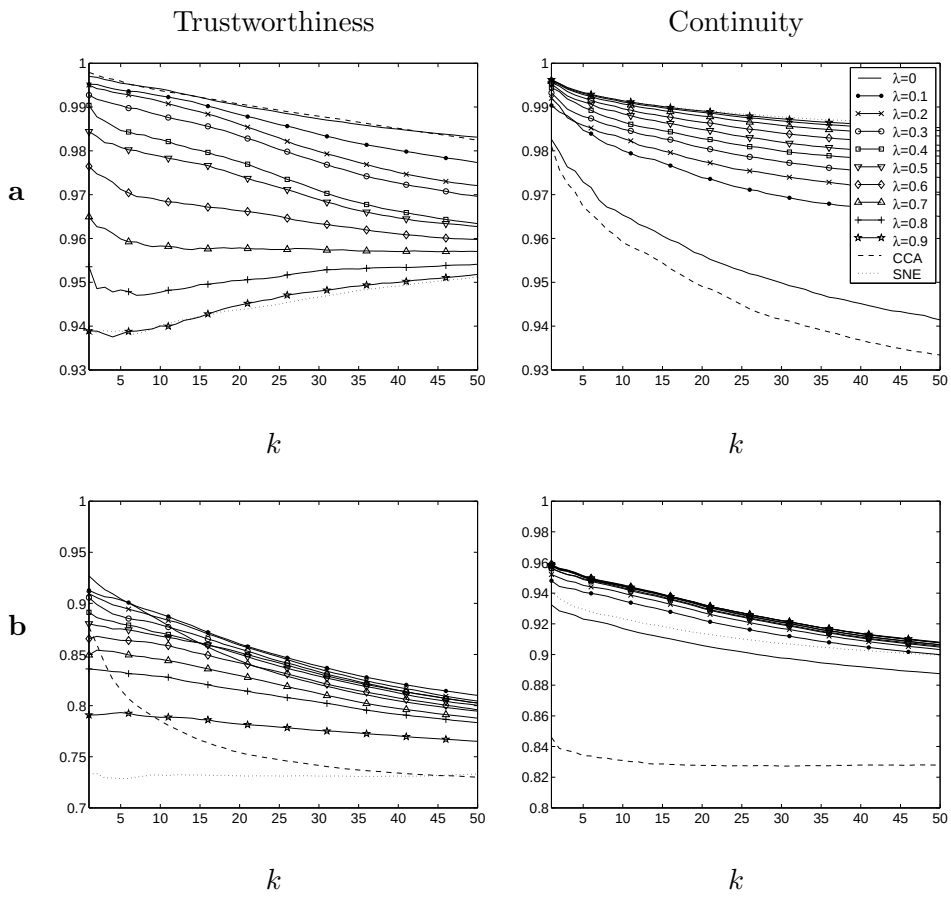

Figure 5: Typical behavior of local MDS when λ is changed. **a**) Local MDS illustrated on the S-curve manifold **b**) Local MDS with geodesic distances illustrated on the Mouse gene expression data. Results from stochastic neighbor embedding (SNE) and Curvilinear component analysis (CCA) are included for reference.

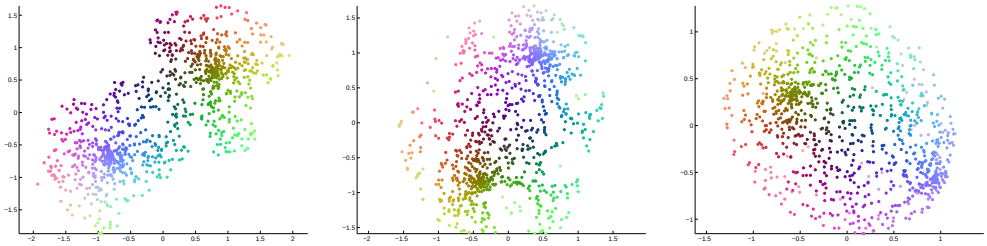

Figure 6: Three projections of a three-dimensional spherical cell with local MDS. On the left, trustworthiness of the projection is maximized by selecting $\lambda = 0$. In the middle and right, discontinuity of the projection is penalized as well, by setting $\lambda = 0.1$ and $\lambda = 0.9$, respectively.

based on empirical findings, we recommend that λ should be kept within the range $[0,\ 0.5]$ when Euclidean distances are used. Geodesic distances seem to alleviate the problem. Overall, geodesic distances seem to allow higher values of continuity to be reached.

Second, in Figure 4 on the Mouse gene expression data set both versions of local MDS increase in trustworthiness when λ is changed from 0 to 0.1. This behavior is easy to understand if we look at the full trustworthiness curves in Figure 5**b** (curves for the Euclidean case are similar). On this data set, when the parameter λ is zero, trustworthiness is first high at very small neighborhoods but then drops relatively fast. On the other hand, when $\lambda = 0.1$ trustworthiness is slightly lower at the very small neighborhoods but the curve has a gentler slope. At a neighborhood of size 10 the trustworthiness of local MDS with $\lambda = 0$ has already dropped below that produced when $\lambda = 0.1$.

5 Discussion

In this paper we have utilized geodesic distances as a way to help unfold the data for better visualization. The end goal has been to visualize the Euclidean proximities. It can also alternatively be argued that the geodesic distance, instead of the Euclidean distance, is the true metric of the data. This line of thought can be followed by replicating the experiments of this paper, but this time calculating the trustworthiness and continuity values using geodesic distances. To be consistent, we would not then talk about isomap, for instance, but of linear MDS in the geodesic distance measure instead.

Utilizing geodesic distances can lead to better visualization results. The downside is that the size of the neighborhood in the graph approximation should be selected correctly for best results. Thus far there is no simple way of doing the selection, and the methods have to be run several times with different neighborhood sizes. This can be slow especially if the method has to be run several times from different starting positions to avoid local minima.

6 Conclusions

We tested several different nonlinear dimensionality reduction methods. Of these, isomap, Laplacian eigenmap, and LLE are designed to extract manifolds while SNE, CCA, and SOM are more generally targeted for dimensionality reduction. One of the main tasks that these methods are used for is visualization. Thus it is important to under-

stand how they perform in typical visualization situations, and what kinds of tradeoffs they make. Of the methods tested here only SOM and CCA can be recommended for general visualization tasks where high trustworthiness is required. If preservation of original neighborhoods is required the linear method PCA is a good first choice followed by SNE which can produce better results but is computationally heavy, and prone to problems caused by local minima.

We introduced an extension of CCA called local MDS, that according to the experimental results is capable of controlling the tradeoff between trustworthiness and continuity of the projection.

References

Belkin, M., & Niyogi, P. (2002a). Laplacian eigenmaps and spectral techniques for embedding and clustering. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems 14* (p. 585-591). Cambridge, MA: MIT Press.

Belkin, M., & Niyogi, P. (2002b). *Laplacian eigenmaps for dimensionality reduction and data representation.* (Tech. Rep. No. TR-2002-01), Department of Computer Science, The University of Chicago.

Bernstein, M., de Silva, V., Langford, J. C., & Tenenbaum, J. B. (2000). *Graph approximations to geodesics on embedded*

manifolds. (Tech. Rep.), Department of Psychology, Stanford University.

Borg, I., & Groenen, P. (1997). *Modern multidimensional scaling.* New York: Springer.

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

Gower, J. C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, *53*, 325–338.

Hinton, G., & Roweis, S. (2002). Stochastic neighbor embedding. In *Advances in neural information processing systems 15* (pp. 833–840). MIT Press.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, *24*, 417–441, 498–520.

Kaski, S., Nikkilä, J., Oja, M., Venna, J., Törönen, P., & Castrén, E. (2003). Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics*, *4*, 48.

Kohonen, T. (2001). *Self-Organizing Maps* (3rd ed.). Berlin:

Springer.

Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrica*, *29*, 1–26.

Lee, J. A., Lendasse, A., Donckers, N., & Verleysen, M. (2000). A robust nonlinear projection method. In M. Verleysen (Ed.), *ESANN'2000, Eighth European Symposium on Artificial Neural Networks* (pp. 13–20). Bruges, Belgium: D-Facto Publications.

Lee, J. A., Lendasse, A., & Verleysen, M. (2004). Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis. *Neurocomputing*, *57*, 49–76.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*, 2323–2326.

Sammon Jr., J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, *C-18*, 401-409.

Segal, E., Koller, N. F. A. D., & Regev, A. (2004). A module map showing conditional activity of expression modules in cancer. *Nature Genetics*, *36*, 1090-1098.

Su, A. I., Cooke, M. P., Ching, K. A., Hakak, Y., Walker, J. R.,

Wiltshire, T., et al. (2002). Large-scale analysis of the human and mouse transcriptomes. *Proceedings of the National Academy of Sciences*, *99*, 4465-4470.

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

Torgerson, W. S. (1952). Multidimensional scaling: I. theory and method. *Psychometrika*, *17*, 401–419.

Ultsch, A. (1993). Self-organization neural networks for visualization and classification. In O. Opitz, B. Lausen, & R. Klar (Eds.), *Information and classification* (pp. 307–313). Berlin: Springer-Verlag.

Venna, J., & Kaski, S. (2001). Neighborhood preservation in nonlinear projection methods: An experimental study. In G. Dorffner, H. Bischof, & K. Hornik (Eds.), *Proceedings of ICANN 2001, international conference on artificial neural networks* (pp. 485–491). Berlin: Springer.