Publication 4

Jarkko Venna, and Samuel Kaski. Comparison of visualization methods for an atlas of gene expression data sets. *Information Visualization*. To Appear.

4

A paper

Comparison of visualization methods for an atlas of gene expression data sets

Jarkko Venna and Samuel Kaski

Address:

Laboratory of Computer and Information Science

Helsinki University of Technology

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

Telephone: +358 9 4514335

Fax: +358 9 4513277

Email: {jarkko.venna, samuel.kaski}@hut.fi

Running title: Visualizations for an expression atlas

Total number of pages: 47

	mentioned in text	attached
Number of figures	8	8
Number of tables	4	4

This paper (or a similar version) is not currently under review by a journal or conference, nor will it be submitted to such within the next three months.

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.

Comparison of visualization methods for an atlas of gene expression data sets

Abstract

This paper has two intertwined goals: (i) to study the feasibility of an atlas of gene expression *data sets* as a visual interface to expression databanks, and (ii) to study which dimensionality reduction methods would be suitable for visualizing very high-dimensional data sets. Several new methods have been recently proposed for the estimation of data manifolds or embeddings, but they have so far not been compared in the task of *visualization*. In visualizations the dimensionality is constrained, in addition to the data itself, by the presentation medium. It turns out that an older method, curvilinear components analysis, outperforms the new ones in terms of trustworthiness of the projections. In a sample databank on gene expression, the main sources of variation were the differences between data sets, different labs, and different measurement methods. This hints at a need for better methods for making the data sets commensurable, in accordance with earlier studies. The good news is that the visualized overview, *expression atlas*, reveals many of these subsets. Hence, we conclude that dimensionality reduction even from 1339 to 2 can produce a useful interface to gene expression databanks.

Keywords

Gene expression, manifold extraction, nonlinear dimensionality reduction, visualization

1 Introduction

The next challenge following the invention of high-throughput microarray techniques for genome-wide expression and other measurements, and collecting the measurement data into community-resource databanks, is in creating methods for searching and browsing the data. In this paper we study a technical subproblem of this very general task which benefits from information visualization. While the problem is motivated by visualization of gene expression data, the methods are more generally applicable to other high-dimensional data as well.

Assume a user searching for a gene expression data set or sets. Typical examples are modelers searching for data needed for estimating their models, or experimental biologists searching for relevant data to compare with their own findings. In analogy to textual information retrieval, the user needs tools for two different kinds of tasks: (i) Searching for a specific data set. This task is analogous to (textual) information retrieval, and feasible if the user knows the relevant search terms or keywords. A relevant "keyword" for a data set search could be a sample data set, when the task is to find the most similar existing sets. (ii) Browse the databanks to find interesting or potentially relevant data sets. This is analogous to text mining and browsing using visual interfaces. The goal of the user is tacit, and hence suitable search "terms" are either not known or insufficient. The best an automated system can do is to help the user in browsing the databank. Visualizations can be of help here. It is also possible to combine visual exploration with a query interface. The results of the query are shown dynamically on the visualization. This way the user can use the metadata in the database to direct the exploration to specific domains of interest.

The first big problem in visualizing gene expression data sets stems from their dimensionality, which may be thousands or even tens of thousands, equaling the number of genes on a microarray. Traditional dimensionality reduction methods include the linear principal components analysis (PCA) [1] which tries to pre-

serve variance in the data, and multidimensional scaling (MDS) which tries to preserve pairwise distances between data points. There exist lots of variants of MDS for different types of distances, and for emphasizing preservation of different properties in them [2]. We have earlier [3] compared these methods together with hierarchical clustering and the Self-Organizing Map (SOM) [4], a neurally-inspired dimensionality reduction method, for a related task: visualization of similarity relationships between genes, based on their expression profiles in a set of treatments. The result of the comparison was that the SOM-visualizations were more trustworthy, in the sense that a set of genes found close-by each other on a SOM display was more likely to be similar in terms of the original data as well. In other words, the proximities visible on the displays were more trustworthy. The other side of the coin is whether the visualization is able to show all of the proximities present in the original data. It turned out that the SOM was among the best methods here as well. There is later evidence [5] that a related method, curvilinear components analysis (CCA) [6], may outperform the SOM in this task.

There has recently been a surge of interest in methods for finding latent lower-dimensional manifolds of data, or nonlinear embeddings of low-dimensional data manifolds in a higher-dimensional data space. Most of the new methods have been described in Section 2. Although it is sometimes implied that these methods might be suitable for *visualizing* the manifold as well, we are not aware of any benchmarking studies. The key difference between manifold estimation and visualization is that a visualization can be only two- or at most three-dimensional. In this work we will study what happens to the results of the manifold estimation methods when the end dimensionality is fixed to two for visualization purposes, and compare them with the method found to be the best in the earlier studies (CCA).

Another main problem in visualizing gene expression data sets, and in fact in all comparisons of the sets, is how to make them commensurable. The measurement results depend on experimental and measurement procedures, the specifics

of the organism and its biological state, biological sampling, measurement devices, and normalization and postprocessing procedures. Although the problem seems hopeless, even very simple normalization procedures have resulted in promising data analysis results in a recent study which combined data from a variety of human cancer studies [7]. This prompted us to study the feasibility of a gene expression data atlas, where only very simple procedures have been applied to make the sets commensurable.

2 Methods

In this section we describe briefly the main classical methods for visualizing similarity relationships in data, and the recent ones that focus on finding data manifolds or embeddings. Lastly, we describe the measures for the goodness of visualizations we will use in the experiments. Many of the visualization methods rely on theoretical ideas which are hard to describe briefly in an intuitively appealing manner; we will complement the presentation in Section 3 with demonstrations on toy data sets selected to highlight differences in the performance of the methods.

All methods we will discuss take the original data, which can be considered as points in a high-dimensional data space, and represent each point as a point in a lower-dimensional space. The input data will be denoted by the vectors $\mathbf{x}_i$, indexed by i, and their low-dimensional representations by $\mathbf{y}_i$. For visualization purposes the dimensionality of the *output space* needs to be two or at most three, whereas the original or *input space* may have thousands of dimensions. The task of a visualization method is to construct the representations $\mathbf{y}_i$ in such a way that essential properties of the similarity relationships of the $\mathbf{x}_i$ are preserved; it is clear that lower-dimensional representations cannot in general preserve everything of a higher-dimensional data set, and all methods need to make a compromise on what to preserve. Computation of the representation $\mathbf{y}_i$ of $\mathbf{x}_i$ is often called *projection*; when the projection is linear it can be expressed as a

matrix product (see Principal component analysis below), and the visualization task is to find a good matrix. Alternatively, the representations $\mathbf{y}_i$ can be optimized directly (see Multidimensional scaling below) which implicitly defines a "nonlinear projection."

The traditional way of visualizing multidimensional data is to project the data to the plane spanned by two of the axes of the data space at a time. This representation has two drawbacks, however. First, as the number of dimensions grows the number of scatter plots increases rapidly, and there is no way of *a priori* knowing which plots are the most relevant ones. The second problem is that linear projections to the original axes might not bring out the interesting possibly nonlinear relationships in the data. To solve these two problems more sophisticated visualization methods have been developed.

2.1 Principal component analysis (PCA)

The goal of PCA [1] is to find directions, or components, where the data has maximal variance. When data is projected to a PCA component the variance in the data is preserved maximally. Technically, the components can be found by solving the eigenvalue problem

$$\mathbf{C}_x\mathbf{a} = \lambda\mathbf{a} \ , \tag{1}$$

where $\mathbf{C}_x$ is the covariance matrix of the vectorial data $\mathbf{x}$. Here $\mathbf{a}$ is an eigenvector and λ the corresponding eigenvalue. The problem has several solutions, of which the ones with the largest λ are the most interesting in PCA. For visualization the data points need to be projected onto a two-dimensional plane defined by the two main components. This is done by

$$\mathbf{y}_i = \mathbf{A}\mathbf{x}_i, \tag{2}$$

where $\mathbf{A}$ is the matrix containing the eigenvectors corresponding to the two largest eigenvalues, and $\mathbf{y}_i$ is the two-dimensional representation of $\mathbf{x}_i$. In effect, this produces an image where the data is spread as widely as is possible by using a linear projection.

2.2 Multidimensional scaling (MDS)

Traditional multidimensional scaling is not included among the methods that we test in this paper, but a short description helps to explain the more complex methods below.

There are several different variants of MDS [2], but they all have a common goal: to find a configuration of points in space that preserves the pairwise distance matrix as well as possible. The simplest version is the linear MDS [8, 9], also called classical scaling. The solution to linear MDS can found by solving an eigenvalue problem, in fact, linear MDS is very closely related to PCA. It can be shown [9] that when the dimensionality of the sought solutions is the same and the distance measure is Euclidean, the projection of the original data to the PCA subspace equals the configuration of points found by linear MDS. This results implies the interpretation that PCA tries to preserve the squared distances between data points, and linear MDS finds a solution that is a linear projection of the original data.

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

$$E = \sum_{ij} (d_{i,j} - d(\mathbf{y}_i, \mathbf{y}_j))^2, \tag{3}$$

where $d_{i,j}$ is the distance between points i and j in the input space (entry in the distance matrix) and $d(\mathbf{y}_i, \mathbf{y}_j)$ the distance in the output space of the selected dimensionality, between the representations $\mathbf{y}_i$ and $\mathbf{y}_j$ of the points. The representations are simply the coordinates of the points in the output space. The cost function is typically optimized using iterative methods. In this case the resulting configuration of points is not the result of a linear transformation

of the data but a result of a nonlinear transformation defined implicitly by the optimized configuration. Thus it is usually better able to preserve the distances than a purely linear method would be.

Most versions of MDS use some variant of this cost function. For instance, Sammon's mapping [10] gives small distances a larger weight. Non-metric MDS [11] on the other hand, allows distances to be modified by any monotonic function. There exists a huge number of different variants, all sharing the task of preserving pairwise distances, in one way or the other.

2.3 Isomap

The Isomap [15] is a variant of MDS. It too finds a configuration of points that matches the given distance matrix. The difference from traditional MDS is in how the distances in the input space are defined. Isomap uses geodesic distances along the manifold the data forms, instead of direct pairwise Euclidean distances. When the MDS algorithm then reproduces the geodesic distances with the pairwise Euclidean distances in the output space, the manifold structure in the original data becomes unfolded. This can be illustrated with the following example. Take a string and tie knots in it to represent the data points. Then set it on a table top, say, in the form of the letter S. This string is the manifold (here one-dimensional) and it has been embedded to the two-dimensional space of the table top. Next measure all the pairwise distances between the knots *along the string*. The task of the visualization is to represent these distances with normal Euclidean distances in the output space, that is, only straight lines between representations of the knots. This can only be done by pulling the string straight. Thus the S shaped manifold, the string, was unfolded by representing the geodesic distances with Euclidean distances.

Technically, the geodesic distances are approximated with the shortest path distances calculated along the k-nearest-neighbor graph formed of the data. Each data point is a vertex in the graph. There is an edge between points i and j if j is among the k nearest neighbors of i or i among the k nearest neighbors

of j. The weight of each edge is the Euclidean distance between the two points. The actual embedding of points is found by standard linear MDS, applied to the shortest-path distance matrix. It has been shown [16] 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.

2.4 Curvilinear component analysis (CCA)

Like Isomap, CCA [6] has similarities with MDS. Where Isomap changes the definition of distances, CCA chooses to look for a configuration of points that preserves only a subset of the distances. The starting point is a random initialization of points ($\mathbf{y}_i$) in the reduced-dimensional output space, and a pairwise distance matrix between the original data points ($\mathbf{x}_i$). The cost function measures preservation of the original pairwise Euclidean distances, but now weighted by a coefficient F that depends on the distance between the points in the *output space*. Here CCA differs from traditional MDS methods. The idea is to concentrate on preserving distances between points that are near each other in the visualization. The cost function is

$$E = \frac{1}{2} \sum_i \sum_{j \neq i} (d(\mathbf{x}_i, \mathbf{x}_j) - d(\mathbf{y}_i, \mathbf{y}_j))^2 F(d(\mathbf{y}_i, \mathbf{y}_j), \lambda_y). \tag{4}$$

The term $F(d(\mathbf{y}_i, \mathbf{y}_j), \lambda_y)$ determines how strongly errors in reproducing the distance between the points i and j contribute to the cost function. It is usually defined as an area of influence around a data point in the output space:

$$F(d(\mathbf{y}_i, \mathbf{y}_j), \lambda_y) = \begin{cases} 1 & \text{if } d(\mathbf{y}_i, \mathbf{y}_j) \leq \lambda_y \\ 0 & \text{if } d(\mathbf{y}_i, \mathbf{y}_j) > \lambda_y. \end{cases} \tag{5}$$

The cost function is optimized using a kind of a stochastic gradient descent algorithm. In the beginning of optimization the radius of the area of influence, λ_y, is kept large enough to cover all or at least most of the data points. During

the optimization it is slowly reduced to zero. Thus at initial stages CCA works exactly as standard metric MDS where all distances are treated equally. The dynamic reduction of the area of influence around data points during the optimization results in an unfolding effect similar to the one in the Isomap. The differences of the methods will be demonstrated in Section 3.

Contrary to the other methods applied in this paper, the cost function of CCA can have several local optima. Although this can potentially cause problems, the solutions found by CCA have been quite good in practice, even starting from only one initialization.

2.5 Locally linear embedding (LLE)

The LLE algorithm [12] is based on the assumption that the data lies on or close to a low-dimensional manifold in the high-dimensional space. If this is the case then we can make a locally linear approximation of the manifold, and assume that a point and its neighbors lie in a locally linear subspace on the manifold. The geometry of this subspace can be estimated by calculating the linear coefficients that reconstruct each data point from its k nearest neighbors. In effect the k nearest neighbors define a (possibly over-complete) basis and the position of the data point is given as coordinates (coefficients) in this basis. If the data lies in a low-dimensional linear subspace of the high dimensional input space then all these local coordinate systems are inherently aligned and a simple linear transformation of the data can be used to find a low dimensional representation that exactly reproduces the local coordinates of each data point. In the case of a nonlinear manifold the local coordinate systems are inherently incompatible, and a cost function has to be defined to optimize the representation.

The LLE algorithm works in two phases. First the local "coordinates" are calculated; a coordinate system represents a point, here $\mathbf{x}_i$, as a linear combination of "basis vectors", here the neighbors of $\mathbf{x}_i$. In the first stage the coordinates W_{ij} are optimized to best represent x_i. The total reconstruction

error to be optimized is

$$E(\mathbf{W}) = \sum_i |\mathbf{x}_i - \sum_j W_{ij}\mathbf{x}_j|^2. \quad (6)$$

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 $\sum_j W_{ij} = 1$.

In the second stage the task is to find such low-dimensional representations $\mathbf{y}_i$ that the local coordinate systems are kept as compatible as possible. Now the representation error is

$$E(\mathbf{Y}) = \sum_i |\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j|^2 \, . \quad (7)$$

This time the coordinates W_{ij} are kept constant at the values produced by stage one, and the positions of the data points $\mathbf{y}_i$ are changed. This optimization can be solve by an eigenvalue analysis (details in [12]).

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

2.6 Laplacian eigenmap

The Laplacian eigenmap [13] algorithm is related to the LLE algorithm but motivated graph-theoretically. The algorithm is the hardest to describe intuitively, and we have to resort to a brief and more formal description here.

The first step is to form the k-nearest-neighbor graph as in the Isomap algorithm, but here often the edges are simply assigned a weight $W_{ij} = 1$ if the points i and j are neighbors, and zero otherwise. This has been found to work well in practice [14].

The neighborhood graph defines the similarity structure that we would like to see in the visualization. Laplacian eigenmap tries to represent this structure

by optimizing the cost function

$$\frac{1}{2}\sum_{i,j}\|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij} = tr(\mathbf{Y}^T\mathbf{L}\mathbf{Y}), \tag{8}$$

where $\mathbf{L} = \mathbf{D} - \mathbf{W}$, $\mathbf{D}$ is the diagonal matrix with elements $D_{ii} = \sum_j W_{ij}$, and $\mathbf{Y}$ is the matrix holding the coordinates of all data points. Minimizing the cost function tries to put points that are connected in the graph as close by as possible. There is a trivial solution to the cost function. That is to put all the representations in a single location. This can be prevented by adding the constraint $\mathbf{Y}^T\mathbf{D}\mathbf{Y} = I$.

In practice the configuration of points in the two-dimensional visualization can be found by solving the generalized eigenvalue problem

$$\mathbf{L}\mathbf{y} = \lambda\mathbf{D}\mathbf{y}. \tag{9}$$

The embedding of the data points is given by the eigenvectors having the 2 smallest eigenvalues, after discarding the smallest (always zero) eigenvalue. The zero eigenvalue corresponds to the case where all points are represented by a single location.

According to [18] the Laplacian eigenmap can also be seen as a variant of linear MDS that tries to preserve the expected commute time distance matrix. Commute time distance is defined to be the expected time a Markov chain random walker takes when traveling from one vertex of the neighborhood graph to another and back. The expectation is taken over all possible paths.

2.7 Measuring trustworthiness of a visualization

When visualizing similarities of data points, the local similarities are the most salient: when looking at a point the first perceptions are which other points are proximate, and which close-by points form groups. We have developed a way to measure how trustworthy the proximities presented by the visualization are

[3, 17].

We consider a projection onto a display *trustworthy* if the set of k closest neighbors of a point on the display is also close-by in the original space. The setting is illustrated in Figure 1. The measure is very closely related to the *precision* measure in information retrieval, where the task is to find a set of documents that are relevant to a query. We can consider the neighborhood in the input space as the set of relevant items and the neighborhood in the visualization as the result of the query. If we calculate the proportion of points that are in the neighborhood in the visualization but not in the input space we get a number that quantifies the loss of precision; in fact the standard measure of precision equals one minus this number. The pure number of errors is not very informative, however, and we have selected to quantify the magnitude of the error by ranking the data points based on their distance instead of just counting the number of errors. In information retrieval this would be analogous to ordering each item in the data base according to their non-binary relevance to the query. Now to have perfect precision, the set of retrieved items (the neighbors in the visualization) should contain those points that have a rank $\leq k$. Those points that are in the retrieved set but have a rank $> k$ cause the precision to fall and we define the amount of the error to increase proportionally to the increase in the rank, that is, $r - k$, for a point with rank r. To get the trustworthiness for the whole visualization this is summed over all data points.

More formally, let N be the number of data samples and $r(i, j)$ be the rank of the 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 the sample i in the visualization display but not in the original data space. Our measure of trustworthiness of the visualization, M_1, is defined by

$$M_1(k) = 1 - A(k) \sum_{i=1}^{N} \sum_{j \in U_k(i)} (r(i, j) - k) \, , \tag{10}$$

where $A(k) = 2/(Nk(2N-3k-1))$ scales the values between zero and one. The worst attainable values of M_1 may, at least in principle, vary with k, and will be approximated in our experiments with random projections and with random neighborhoods.

While the trustworthiness measure shows how well the points in a neighborhood on the display match the neighborhood in the original space, it is also of interest to know whether neighbors in the original space remain neighbors. If points become pushed out of the neighborhood in the visualization process, *discontinuities* arise in the projection. As a result of the latter kinds of errors, not all proximities existing in the original data are visible in the visualization. As in the case with trustworthiness an analogy can be done with information retrieval, but in this case continuity is connected to the *recall* measure.

The errors caused by discontinuities may be quantified analogously to the errors in trustworthiness. Let $V_k(i)$ be the set of those 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 on the display. The effects of discontinuities of the projection are quantified by how well the continuity of the original neighborhoods are preserved, measured by

$$M_2(k) = 1 - A(k) \sum_{i=1}^{N} \sum_{j \in V_k(i)} (\hat{r}(i,j) - k) \; . \tag{11}$$

In case of ties in rank ordering, all compatible rank orders are assumed equally likely, and averages of the error measures are computed. This is very rare, however.

In practice we usually study the trustworthiness and continuity measures with a range of values of k; small k are more important for the quality of visualization but a very rapid drop of trustworthiness for slightly larger neighborhoods would result in a bad visualization. Note that the k in the measures 10 and 11, and the k in th k-neighbor graphs of many of the visualization methods are

different symbols with no direct connection.

3 Comparisons of the methods on toy data sets

Although the cost function of a visualization method describes exactly what the method tries to do, it is often very hard to understand, based on the abstract definition, how the method behaves in practice. This is especially true for the nonlinear methods. Because of this we start by testing the methods on data sets which are simple, representative of the kinds of data the methods need to handle (manifolds, clusters), and for which we at least approximatively know what a good result would be like. The purpose of these experiments is to try to understand the behavior and limitations of the different methods in situations that are likely to appear when visualizing real-world data. Three groups of data sets are studied. In the first group there are three sets that are designed to test how well the methods find a simple nonlinear manifold in the data, and what happens when the manifold gets higher-dimensional than the visualization. This is a task many of the methods were developed for. The second group consist of two data sets that test what happens when the topology of the manifold is spherical instead of flat. This can be the case, for example, when viewing an object from several angles, or when the lengths of centered data vectors are normalized to be equal as is common for bioinformatics data. In the third group there is only one data set that contains clusters instead of a single nice manifold. All data sets are three-dimensional and contain 1000 data points.

3.1 Data sets

S-curve. Three of the data sets contain a two-dimensional S-shaped curve. The first (Figure 2**a**) can be thought of as a paper strip bended to an S-shape. The data are uniformly distributed on the two-dimensional manifold. In the second and third data sets in Figures 2**b** and 2**c**, respectively, the manifold has a non-zero thickness. The data was constructed by adding a spherical normally

distributed displacement to each point. In the second data set the standard deviation was 0.3 and in the third it was 0.6.

The purpose of these three data sets is to investigate how well the methods recover the manifold and what happens as the thickness of the manifold, and hence its dimensionality, increases. The S-curve with zero thickness can safely be assumed to be a two-dimensional manifold, but the curve with moderate thickness already raises some doubt, and the thickest "curve" is clearly three-dimensional. For the local proximity relationships to be trustworthy the two-dimensional projection should "fold" inside the thicker manifold instead of just pressing it flat.

Sphere. Two of the data sets contain a sphere in a three-dimensional space. In the first set (Figure 2**d**) the data lie on the surface of a unit sphere. In the second set (Figure 2**e**) the spherical shell has a thickness of one, symmetrically around the unit sphere. (In practice the data was formed by perturbing samples in the radial direction from the unit sphere, and hence the density within the sphere is not uniform.)

A spherical data set is very hard for many types of projection methods. A typical result is that the sphere becomes squashed flat so that the opposing sides get pressed together. A good solution in terms of the trustworthiness would be to cut the sphere open and spread it in a manner similar to printed maps of the Earth.

Cluster. In this set the data (Figure 2**f**) are distributed in six clusters placed in a symmetric configuration. Five of the clusters are normally distributed and one is a S-curved manifold.

Several of the tested methods have been noted to have problems with clustered data. Yet, in an explorative visualization task any clusters and their structure are usually of great interest. We included the cluster data set to benchmark the methods on this very different visualization task, and to find out whether they can find the manifold structure in one of the clusters.

3.2 Results

The methods having parameters were run for a range of parameter values, and the best result in terms of trustworthiness was selected. This was done for all data sets.

The methods having a number of neighbors parameter (for the nearest neighbors graph) were run several times with values of k going from 4 to 20. On the cluster data set the methods were additionally run with k ranging from 64 to 80. The neighborhood graph becomes connected only for $k \geq 64$ in this data set. CCA was run ten times on each data set with different randomly chosen initial conditions. In each case the result with the best trustworthiness was selected.

The methods were compared using the trustworthiness and continuity measures (Figures 4 and 5) and qualitatively by looking at the visualizations produced by the methods (samples in Figure 6). The results are summarized in Figure 3 and a more thorough description of the findings for each method is given below.

PCA. The trustworthiness of PCA projections was never among the best on the data sets tested (Figures 4 and 5). An interesting finding is that, on the average, PCA is worse than a random projection on the cluster data set. On this data set the PCA projects two clusters on top of each other while a random projection usually manages to separate them.

In contrast, PCA is among the best in preserving the original neighborhoods. It never projects two near-by points far from each other. PCA produces easily interpretable visualizations, but they are mainly capable of reproducing the global structure instead of the local proximity relationships.

LLE performs well on the S-curve (Figures 4 and 6**a**), although its performance starts to detoriate as the thickness of the manifold increases. On this data set the performances of LLE and Isomap are similar. On the cluster data set the LLE-visualization consists of a small set of isolated points (for $k < 64$),

and it is not possible to perceive any structure inside the clusters. When k is large enough, implying that the neighborhood graph becomes connected, LLE is able to separate the S-manifold from the other clusters, but it becomes stretched to a one-dimensional straight line. The number of neighbors is then clearly too large, and hence the projection is qualitatively poor even tough the trustworthiness is better than with a PCA projection.

In summary, LLE seems sensitive to the choice of the parameter that selects the number of neighbors.

Laplacian eigenmap. The trustworthiness of projections created by the Laplacian eigenmap is the second best on all data sets except the S-curve sets (Figures 4 and 5). For those sets it is among the two worst methods in preserving the original neighborhoods as well. For the S-curve with zero thickness the Laplacian eigenmap collapses the two-dimensional manifold to a very narrow curve (Figure 6**b**).

On the cluster data set the Laplacian eigenmap performs as badly as the LLE when the number of neighbors is not large enough to form a connected neighbor graph. The resulting visualization consists of a small set of points, their number equaling the number of connected components in the neighborhood graph. When the number of neighbors is large enough the Laplacian eigenmap produces a fairly trustworthy projection (Figure 5**c**). The projection is problematic qualitatively, however (Figure 6**c**). It consist of mostly small groups that do not seem to have any structure when studied visually. Actually the small clusters in the display do contain structure, but the scale differences are too large to visualize them in one image.

As was mentioned in section 2.6 the Laplacian eigenmap can be seen as a method that tries to represent the commute time distance matrix of the neighborhood graph with Euclidean distances. This leads to a tendency to magnify pairwise distances. The less likely a short path between two points is going to be on a random walk the larger the magnification of the pairwise distance is.

This distance magnification results in several kinds of distortions in the visualization. Two cases are common. First, as happened with the cluster data above, the scaling of distances within clusters can get very small compared to distances between clusters.

The second kind of common distortion is that "holes" emerge in the visualizations, as illustrated with the sphere data in Figure 6**d**. (The image is an Isomap projection, but the holes are similar to what Laplacian eigenmap produces). If the number of neighbors used to form the neighborhood graph is small, due to random fluctuations some proximate points will not be connected in the graph and the travel time along the shortest path is longer than the real distance on the manifold would indicate. The expectation taken on the travel time magnifies this effect even more, unless the shortest path is the only one that can be taken.

Isomap. The trustworthiness of Isomap projections is slightly worse than of Laplacian eigenmap on all but the S-curve data sets (Figures 4 and 5), on which the Laplacian eigenmap has problems. On the S-curve with zero thickness the Isomap and CCA were the only methods that managed to recover the neighborhoods almost perfectly. The Isomap seems to favor continuity relative to trustworthiness. Qualitatively Isomap projections are better than those produced by Laplacian eigenmap and LLE (it produces less distortions of the kinds shown in Figure 6).

When the number of neighbors parameter k is selected to be small Isomap has a tendency to create holes in the projection (see Figure 6**d**) like the Laplacian eigenmap discussed above. The effect is not as strong as in Laplacian eigenmap because Isomap does not take expectations over all paths but always uses the shortest one.

On the thickest S-curve Isomap produces an image that looks very similar to the one produced by PCA. This is expected as the shortest path distances start to approach the Euclidean distances as the data gets closer to a three-

dimensional data cloud instead of a two-dimensional manifold.

CCA. Figures 4 and 5 show that CCA is consistently the best method in terms of trustworthiness. The same does not hold for the preservation of original neighborhoods, but we should note that all methods, including random projection, are almost perfect for these simple data sets in this respect. In general, CCA gains its high trustworthiness by "cutting the data open." If the manifold is thin this results in a nice unfolding. If the manifold is thicker the cutting is more forceful. An example of such cutting can be seen in Figure 6**e** where the thick S-curve is unfolded. Some neighborhoods have been split so that they appear in two separate locations in the projection (visible as separate patches having similar color), but most of the local neighborhood relationships are still preserved. Another example of cutting is presented in Figure 6**f** where the sphere has been split in two parts that are connected on one side. This cutting results in gains in trustworthiness and reductions in continuity.

A disadvantage of CCA is that sometimes data points "get dropped off" from the optimization as the area of influence is decreased. As the area of influence decreases it is possible that a data point or a group of points is no longer connected to any other points in the data, because their distance from other data points is larger than the area of influence. An example of this can be seen in Figure 6**f**. There are three small groups of points far away from most of the other data points.

4 A gene expression atlas

We applied the methods to constructing an atlas of gene expression data sets, aimed at revealing proximity relationships between and within the data sets. The atlas is computed of a collection of cancer expression sets which have been preprocessed only lightly, to make them somewhat more commensurable. This case study serves both as a test bench for comparing the different visualization methods, and as a feasibility study on whether the current simple preprocessing

methods are sufficient for this task.

4.1 Data and preprocessing

We used the large collection of human gene expression arrays collected by Segal et al. [7]. (The normalized expression compendium is available from http://dags.stanford.edu/cancer.) The compendium consists of 26 different sets, each from a different publication. Altogether the data covers 1973 arrays and 14143 genes. Three different types of microarrays were included, and the studies were carried out in 6 different institutions.

The data sets were normalized using the same methods as in [7]. In the expression values measured with Affymetrix chips, logs (base 2) were taken (setting to 10 expression values that were < 10). For the data sets generated using cDNA chips the log-ratio (base 2) of the measured and control sample was taken. After this the expression values of data sets were normalized, for each gene and data set separately, by subtracting the mean of the gene's expression in the data set from each expression value. Finally the values were rounded to the accuracy of one decimal.

There are lots of values missing from the data set. Several strategies have been suggested on how they could be handled, and for best results the approach should be selected for each visualization method separately. As our goal was to study the visualization performance of the different methods and not to asses the quality of different approaches to handling missing values, we chose to simply remove samples with missing values for this study. First we removed genes that were missing from more than 300 arrays. Then we removed the arrays that still contained missing values. This resulted in a data set containing 1278 arrays and 1339 genes.

4.2 Comparison of visualization methods

Visualization of the compendium of gene expression data sets is a very demanding task for any method. The dimensionality needs to be reduced from 1339

to 2 while still preserving local structure of the data. To make the task even harder the data in the compendium has been produced with different methods and comes from different types of experiments. All this means that it is very unlikely that there is a nicely-formed low-dimensional manifold in the data space.

We compared the visualization methods using the same procedures as in Section 3. Methods having a nearest neighbor parameter were run with the parameter ranging from $k = 4$ to $k = 20$, CCA was run ten times from different random initializations, and the best ones in terms of the trustworthiness were selected.

The performance of the methods can be seen in Figure **7a**. None of the visualizations has a particularly high trustworthiness. This reflects the difficulty of the task. CCA was the best, followed by Laplacian eigenmap and PCA. All of the methods were somewhat better in preserving original neighborhoods (continuity). PCA was the best in this respect, followed by Laplacian eigenmap. LLE performed poorly on both measures. That PCA performs so well, in conjuction with the overall low trustworthiness values, suggests that there is very little low-dimensional manifold structure that could be utilized in the data.

To verify the results on a slightly easier data set, we additionally visualized a collection of gene expression profiles measured from different mouse tissues. The data contained 1600 genes and 45 tissues, and we sought to visualize the similarities of the genes. For details of the data set and of preprocessing see [3]. The results are shown in Figure **7b**. CCA is the best method in terms of trustworthiness, followed by Laplacian eigenmap and Isomap. LLE and PCA are clearly worse in this respect. Laplacian eigenmap and Isomap are the best methods in preserving the original neighborhoods. Qualitatively the visualizations produced by PCA, CCA, and Isomap are the best. Overall the results are similar to those found on the toy data sets, which verifies that the insights gained from the toy data sets can be extended to real-world data.

4.3 Searching for coherent groups in the data

In a cancer expression atlas we would expect the samples from a given cancer type to form a coherent region that is separable from the other cancer types, and we will use this separability to measure the quality of the atlases. Such separability measures both the quality of the data and the quality of the visualization, and that is why we will first measure the coherence of the groups in the original data space, to be used as a yardstick. Then we use this yardstick to measure coherence of the groups in the visualization.

In addition to cancer groups, we will measure coherence of data from the same measurement platform and coming from the same institution. It is expected that they could form relatively homogeneous sets as well. The relative salience of the different groupings, which is an indication of how much of the variation in the data they explain, will be evaluated in the next section.

We want to define coherence in a flexible way. A coherent group need neither be a cluster that is clearly separate from the other data, nor even a unimodal region. A flexible measure is *separability* from the other data. We measure coherence by the classification error of samples of the group, when the other class consists of all the other data (details below).

It is encouraging that many cancer types form coherent groups (first column in Table 1). A less encouraging finding is that cDNA arrays form a very coherent group as well, even though they cover arrays from several different cancer types. This is probably an indication of the fact that the preprocessing done on the arrays was not able to remove the differences between the platforms. This result is additionally related to the finding that arrays from the Stanford University form a very coherent group as well. Namely, most of the studies using cDNAs were done in Stanford.

How the classification was done. To measure how coherent a group X is, we constructed a simple non-parametric classifier to evaluate whether each sample of X is better classified to its own group or to the random group, sampled

without replacement from the rest of the data. If the great majority becomes classified to X, then the group is coherent, otherwise not.

Technically, we used a k-nearest-neighbor classifier ($k = 5$). A random data set of the same size as X is selected (same size to give both groups the same prior probability), and used as reference points of the other class. Each data point in X is then classified by finding the k nearest neighbors from the combined set of the randomly picked samples and X, with the sample to be classified left out. The majority class within the k closest samples wins.

The classification was repeated 1000 times with different random groups. At the end the *mean classification rate* of the group was calculated and reported. We additionally computed P-values for rejecting the hypothesis that the groups come from the same distribution, but the results were almost always highly significant and we decided to report the more informative mean classification rates.

Visualizations preserve coherent attribute groups. For the gene expression atlas to be the most useful, it should be able to preserve most of the coherent groups in the data. To test this we carried out the same tests as above, but this time with the projected samples within the two-dimensional display. We did this for two different methods: CCA which had the highest trustworthiness, and with PCA which was the best in preserving the original neighborhoods. The results are presented in the second and third columns of Table 1. It should be noted that the visualization methods did not utilize any information about the groupings; they were completely unsupervised.

Both methods preserve most of the groups having high coherence (boldface in the table; defined as having the average classification accuracy above the arbitrary threshold of 90%). CCA preserves six of the nine groups and PCA four. An interesting point is that CCA in fact increased the classification rate on the average. Especially several cancer types became more coherent than they were in the original data. Examples of this kind are groups of breast cancer,

various tumors, and NCI60. At the same time, the coherence of the cDNA platform was lowered. The changes caused by PCA tend to cancel each other on the average. Overall there is a tendency for groups that are very coherent in the original data to become slightly less coherent in the visualization and vice versa.

4.4 Are data from different platforms commensurable?

In the previous section it was noted that cDNA measurements formed a very coherent group. This suggests that a large portion of the variation in the data might actually stem from secondary attributes such as the expression platform and institute, rather than cancer.

This would naturally reduce the usefulness of the atlas. If a user is studying, say, B lymphoma and searches for relevant data, it would be nice to be able to find and use data from all platforms (and all institutions). This is of course sensible only if the variation due to the platform is relatively small.

We tested this by measuring whether data from a cancer type is more commensurable with data of the same cancer type, but measured on a different platform, than with any data measured from the same platform. For instance, to see whether the B lymphoma arrays in the gene expression compendium were really organized based on the cancer type and not the platform, we selected a set of arrays that were both cDNA and measured from B lymphoma samples. We then measured whether this set was better classified to either the class of B lymphoma arrays or the class of cDNA arrays. Details of the classification procedure are given in the previous section. The result was that cDNA measurements of B lymphoma samples were always closer to other cDNA measurements, and B lymphoma samples measured with other platforms were more different.

We performed the same experiment on several cancer type vs platform or institute pairs. The complete results are in Table 2.

The results are quite clear concerning the cDNA platform. The cDNA measurements are always closer to other cDNA measurements than to measurements

of the same cancer type but made with a different platform. A similar effect, although not as strong, can be found for the Hu95 platform. On HuGeneFL the results vary.

These results strongly suggest that the simple preprocessing used in [7] is not able to remove effects caused by the platform. Note that this does not necessarily imply anything about the validity of the results in [7]; they have different goals and methods. The visualizations seem to indicate that there might be a difference already in the scales between the data vectors from different platforms. Affymetrix data points seem to be spread out more widely than cDNA-produced points.

A similar effect can be found between the cancer types and the institute where the data was measured. This can, however, to a large extent be explained by the fact that most institutes prefer a specific platform.

Visualizations preserve the general organization of data. We wanted to check whether same findings apply to the visualizations as well, and carried out the same study on the visualized data (Tables 3 and 4). As in the previous section, the visualization was done both using CCA and PCA.

The conclusions on cDNA arrays were the same as with the original data above, although not quite as strong. On Affymetrix the results changed, however. In most cases the strength of the cancer type was increased, for example, in the measurements of Leukemia samples with Hu95. Hu95 is a better classifier than Leukemia in the original data, but the result is reversed in the visualizations. In three cases the classification strength of the cancer class was reduced.

The changes can be at least partly explained by changes in how coherent the groups are (see Table 1). For example, the classification rate of Hu95 was reduced in the visualization, and the Leukemia samples became more coherent.

Overall the visualization process seems to have a positive effect. On the average the cancer classes became stronger for both of the visualization methods.

4.5 Visualizing the gene expression compendium

A display of the gene expression compendium, the gene expression atlas, is presented in Figure 8. Based on the experiments reported above on both artificial data and on the gene expression data itself, we chose to use CCA. On the atlas, many of the cancer types seem to be fairly coherent in the sense of forming continuous areas on the display. This is especially clear for categories that were found to form coherent groups in the original data, such as "Stimulated immune" and "Hela cell cycle." Some classes are multimodal such as "Prostate cancer."

It is even more clear in the atlas that the measurement platforms explain the broad scale of proximity relationships. This matches the results in the previous section. A similar pattern can be found in the distribution of the institutes in the atlas. The institutes tend to use a single platform most of the time for their experiments, which is clearly visible in the displays, and can also be easily verified from the data. Only Harvard Medical School used two different platforms.

5 Discussion

There has been a lot of work done on visualizing gene expression data, but most of it focuses on visualizing the data from a single experiment. Visualizations of combined data from several experiments, measured on different platforms, are mainly targeted for very specific tasks, like in the ChromoViz package [19] that maps gene expression data to chromosomal order, or are limited to only two data sets at a time, like in co-inertia analysis of gene expression data sets [20]. Our goal differs from those in that we try to visualize the whole contents of a gene expression data bank, or possibly the contents of several data banks at once. Combined with the ability to make dynamic queries [21] and possibly with a focus and context interface, the visualization could then be used as the basis for a powerful tool for visual exploration of gene expression data banks,

and for searching of useful data sets for in silico experiments.

Based on our results the simple preprocessing used for the gene expression compendium was not good enough to make the data from different studies commensurable. The main source of variation in the data was the platform used in the experiment. Some recent studies [22, 23] indicate that at least some parts of the differences between platforms are caused by the different sequences used in the probes, and that matching the probes based on the probe sequence instead of the gene id can alleviate the problem. This does not solve the whole problem, however, as there is also evidence of a relatively strong lab effect [24], which can only be removed by standardization of procedures and training of personnel.

In addition to the visualization methods tested here we briefly tried two other methods: Hessian eigenmaps [25] and Alignment of local models [26]. Neither of these was able to produce meaningful results on the gene expression compendium data. It is still somewhat unclear whether the problems were inherent to the methods or were caused by the specific implementations we used.

6 Conclusions

We benchmarked a set of methods for the extremely difficult task of visualizing proximity relationships within the high-dimensional space of microarray measurements. It turned out that all the recently proposed manifold estimation or embedding methods had severe difficulties when the output dimensionality was fixed to two for visualization purposes. An earlier method called curvilinear components analysis (CCA) outperformed them all in terms of trustworthiness of the visualizations.

The methods were compared as a feasibility study for constructing a visualizable gene expression atlas, that is, an atlas of gene expression data sets. It turned out that, as expected, the simple preprocessing methods could not

make the different data sets particularly commensurable. The visualizations did show, however, relationships between the different measurement labs and chip platforms, which are the main sources of variation in the data. Hence, if standardization and more sophisticated preprocessing methods continue to develop to bring the biologically interesting variation to the fore, the information visualization methods are likely to be able to visualize it.

References

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

[2] Ingwer Borg and Patrick Groenen. *Modern Multidimensional Scaling*. Springer, New York, 1997.

[3] Samuel Kaski, Janne Nikkilä, Merja Oja, Jarkko Venna, Petri Törönen, and Eero Castrén. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics*, 4:48, 2003.

[4] Teuvo Kohonen. *Self-Organizing Maps*. Springer, Berlin, 3rd edition, 2001.

[5] Johan Himberg. *From insights to innovations: data mining, visualization, and user interfaces*. PhD thesis, Helsinki University of Technology, Espoo, Finland, 2004.

[6] Pierre Demartines and Jeanny Hérault. Curvilinear component analysis: A self-organizing neural network for nonlinear mapping of data sets. *IEEE Transactions on Neural Networks*, 8:148–154, 1997.

[7] Eran Segal, Nir Friedman, Daphne Koller, and Aviv Regev. A module map showing conditional activity of expression modules in cancer. *Nature Genetics*, 36:1090–1098, 2004.

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

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

[10] John W. Sammon, Jr. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, C-18:401–409, 1969.

[11] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrica*, 29:1–26, 1964.

[12] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[13] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2001. MIT Press.

[14] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. Technical Report TR-2002-01, Department of Computer Science, The University of Chicago, 2002.

[15] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[16] Mira Bernstein, Vin de Silva, John C. Langford, and Joshua B. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Department of Psychology, Stanford University, 2000.

[17] Jarkko Venna and Samuel Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. In G. Dorffner, H. Bischof, and K. Hornik, editors, *Proceedings of ICANN 2001, International Conference on Artificial Neural Networks*, pages 485–491, 2001. Springer.

[18] Jihun Ham, Daniel D. Lee, Sebastian Mika, and Bernhard Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *Proceedings*

of the 21st International Conference on Machine Learning (ICML 2004). ACM, 2004.

[19] Kim Jihoon, Chung Hee-Joon, Park Chan Hee, Park Woong-Yang, and Kim Ju Han. Chromoviz: multimodal visualization of gene expression data onto chromosomes using scalable vector graphics. *Bioinformatics*, 20:1191–1192, 2004.

[20] Aedín C Culhane, Guy Perriére, and Desmond G Higgins. Cross-platform comparison and visualization of gene expression data using co-inertia analysis. *BMC Bioinformatics*, 4:59, 2003.

[21] Christopher Ahlberg, Christopher Williamson, and Ben Shneiderman. Dynamic queries for information exploration: An implementation and evaluation. In *Proceedings of the ACM CHI '92 Conference*, pages 619–626, 1992.

[22] Brigham H. Mecham, Gregory T. Klus, Jeffrey Strovel, Meena Augustus, David Byrne, Peter Bozso, Daniel Z. Wetmore, Thomas J. Mariani, Isaac S. Kohane, and Zoltan Szallasi. Sequence-matched probes produce increased cross-platform consistency and more reproducible biological results in microarray-based gene expression measurements. *Nucleic Acids Research*, 32:e74, 2004.

[23] Kyu-Baek Hwang, Sek Won Kong, Steve A Greenberg, and Peter J Park. Combining gene expression data from different generations of oligonucleotide arrays. *BMC Bioinformatics*, 5:159, 2004.

[24] Rafael A Irizarry, Daniel Warren, Forrest Spencer, Irene F Kim, Shyam Biswal, Bryan C Frank, Edward Gabrielson, Joe G N Garcia, Joel Geoghegan, Gregory Germino, Constance Griffin, Sara C Hilmer, Eric Hoffman, Anne E Jedlicka, Ernest Kawasaki, Francisco Martínez-Murillo, Laura Morsberger, Hannah Lee, David Petersen, John Quackenbush, Alan Scott,

Michael Wilson, Yanqin Yang, Shui Qing Ye, and Wayne Yu. Multiple-laboratory comparison of microarray platforms. *Nature Methods*, 2:345–350, 2005.

[25] David L Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100:5591–5596, 2003.

[26] Jakob J Verbeek, Sam T Roweis, and Nikos Vlassis. Non-linear CCA and PCA by alignment of local models. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, 297–304, Cambridge, MA, 2004. MIT Press.

Figure and table captions

Figure 1: Illustration of the measures of trustworthiness and continuity. Visualization is trustworthy if proximate points in the visualization are proximate also in the original space. Points that are not among the nearest neighbors in the original space but are in the neighborhood in the visualization decrease the trustworthiness (Circles inside the dashed neighborhood on the right; $U_k(i)$). The projection is *continuous* if originally proximate points remain proximate. Points that are in the neighborhood in the original space but not in the visualization decrease continuity (Squares outside the dashed neighborhood on the right; $V_k(i)$). *Green Triangle*: The point defining the neighborhood. *Blue Square*: The five nearest neighbors of the data point in the original space. The saturation of the color indicates how close the point is in the original space. *Red Circle*: Data points that are not neighbors in the original space. The more saturated the color the further the point is in the original space. *Solid line*: Connects neighbors to the data point. *Dashed line*: connects points that have moved in/out of the neighborhood in the visualization process. *Dashed Circle*: The neighborhood.

Figure 2: Images of the artificial data sets. The coordinates of data points are coded with RGB values to help in inspecting the visualizations. Each data set was first scaled to fill the unit cube and then each coordinate axis was associated with one of the RGB components. **a**) Two-dimensional S-curve **b**) Thin S-curve **c**) Thick S-curve **d**) Sphere data **e**) Thick sphere data **f**) Cluster data.

Figure 3: Summary of the comparison of the methods based on the toy data sets

Figure 4: Numerical comparison of the methods on toy data sets: trustworthiness and continuity as a function of the number of neighbors k in the neighbor set. **a**) Two-dimensional S-curve manifold **b**) Thin S-curve manifold **c**) Thick S-curve manifold. Rproj is the average value of 100 linear random projections. The trustworthiness and continuity values of a random mapping are approximately 0.5.

Figure 5: Numerical comparison of the methods on toy data sets: trustworthiness and continuity as a function of the number of neighbors k in the neighbor set. **a**) Sphere data. **b**) Thick sphere data **c**) Cluster data. Rproj is the average value of 100 linear random projections. The trustworthiness and continuity values of a random mapping are approximately 0.5.

Figure 6: Sample projections of toy data, chosen to illustrate visually salient properties of the methods. The color coding of the points is the same as in Figure 2. **a**) LLE projection ($k = 7$) of the S-curve. **b**) Laplacian eigenmap projection ($k = 4$) of the S-curve. **c**) Laplacian eigenmap projection ($k = 72$) of the cluster data. **d**) Isomap projection ($k = 4$) of the sphere data. **e**) CCA projection of the thick S-curve. **f**) CCA projection of the thick sphere data.

Figure 7: Trustworthiness of visualizations and continuity of the projections in (**a**) the gene expression compendium and (**b**) the mouse data, as a function of the number of neighbors in the neighbor set.

Figure 8: A gene expression atlas. Coloring based on cancer type (top), microarray type (bottom left) and institute that performed the experiment (bottom right)

Table 1: Coherence of the groups measured by classification percentage of cancer (and other) groups vs. a random group of the same size. The size of the group is given in parentheses after the group name. Groups containing several studies are marked with an asterisk. Groups having a high classification rate ($> 90\%$) have been emphasized. *Data*: Classification in the original data space. *CCA/PCA*: Classification after dimensionality reduction by CCA/PCA.

Table 2: Are data from the same platform more similar than data from the same cancer type? Pairwise comparison of the classification strength of the cancer type vs the platform/institution. Data points measured with the platform of the column on samples of the cancer type of the row were classified to either of the two classes. The winner (on the average) is shown in the table, together with the classification rate in percentages. Classification was done in the original data space.

Table 3: Are data from the same platform more similar than data from the same cancer type, in the CCA visualization? For explanation see caption of Table 2.

Table 4: Are data from the same platform more similar than data from the same cancer type, in the PCA visualization? For explanation see caption of Table 2.

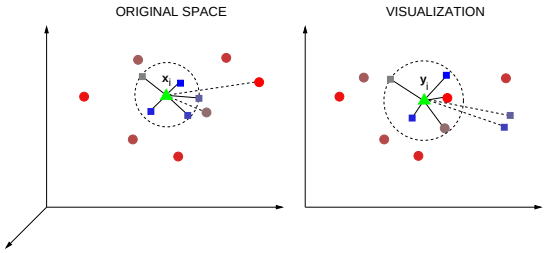

Figure 1:

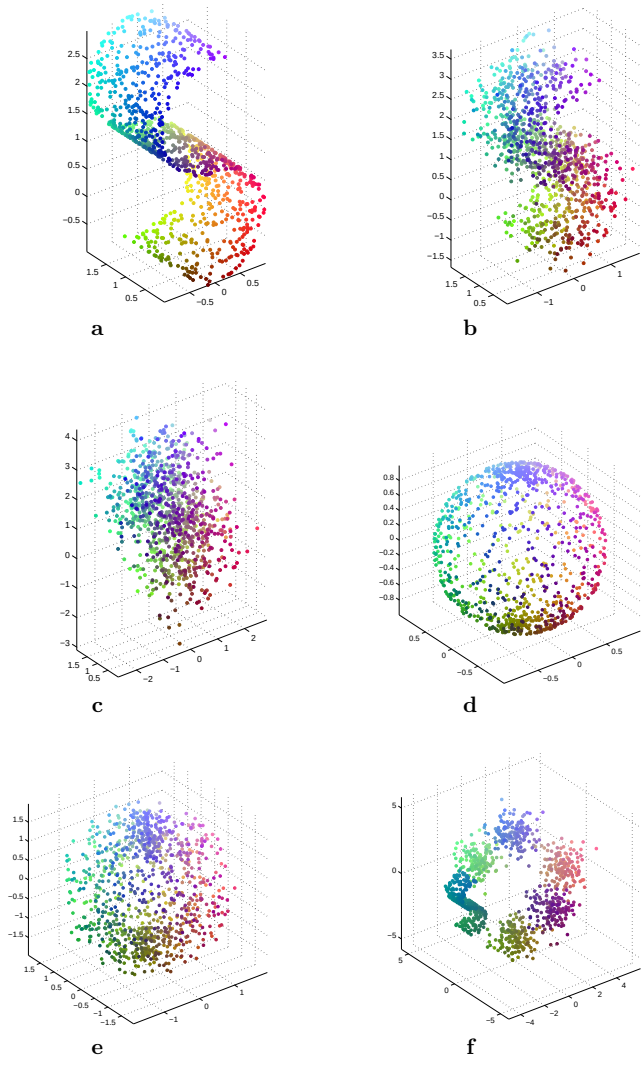

Figure 2:

Method	Advantages	Disadvantages
PCA	Good in terms of continuity of the mapping. Easy to interpret the results.	One of the worst methods in terms of trustworthiness. Can't unfold nonlinear structures.
LLE	Performs relatively well on simple manifolds.	Has severe problems if the data contains clusters. Is sensitive to the number of neighbors parameter k.
Laplacian eigenmap	The second best in terms of trustworthiness.	Has a strong tendency to magnify some distances, which can produce qualitatively very bad visualizations.
Isomap	Good in terms of continuity.	Trustworthiness is only average. Produces some distortions though less than the Laplacian eigenmap
CCA	The best one in terms of trustworthiness. The only method that was able to split the sphere open instead of squashing it flat.	Often the worst in terms of continuity. Sometimes data points can be "dropped off" by the algorithm so that they look like outliers. The cost function has local optima.

Figure 3:

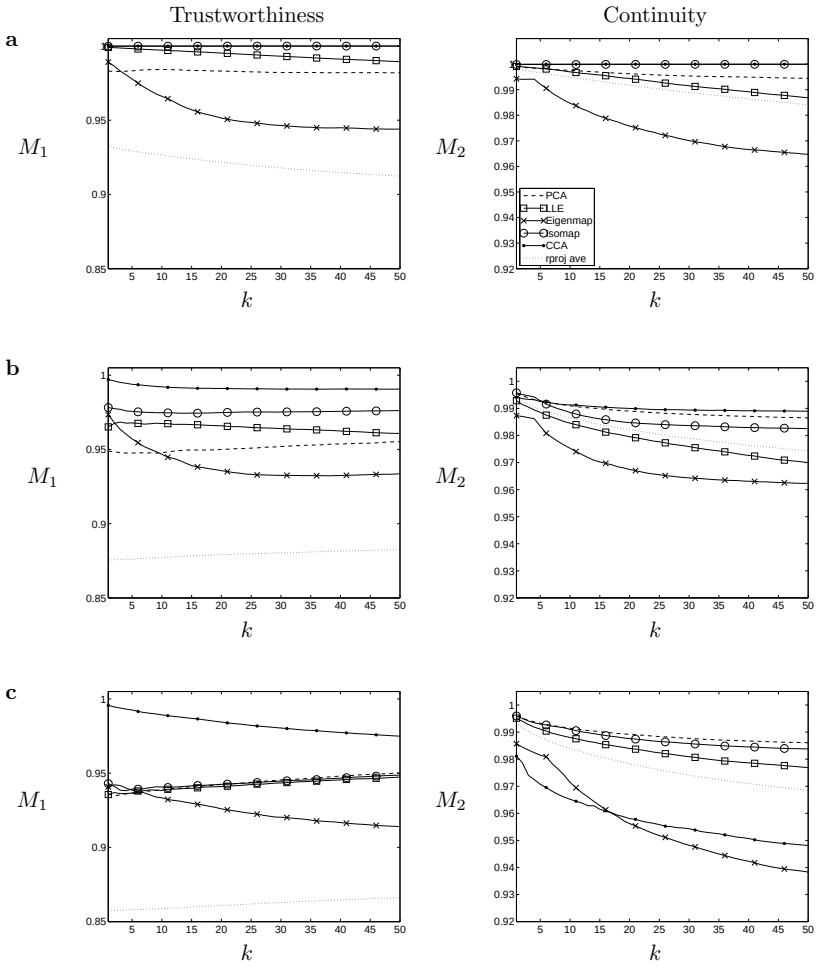
Trustworthiness
Continuity
a
b
c
M_1
M_2
k
PCA
LLE
Eigenmap
Isomap
CCA
rproj ave

Figure 4:

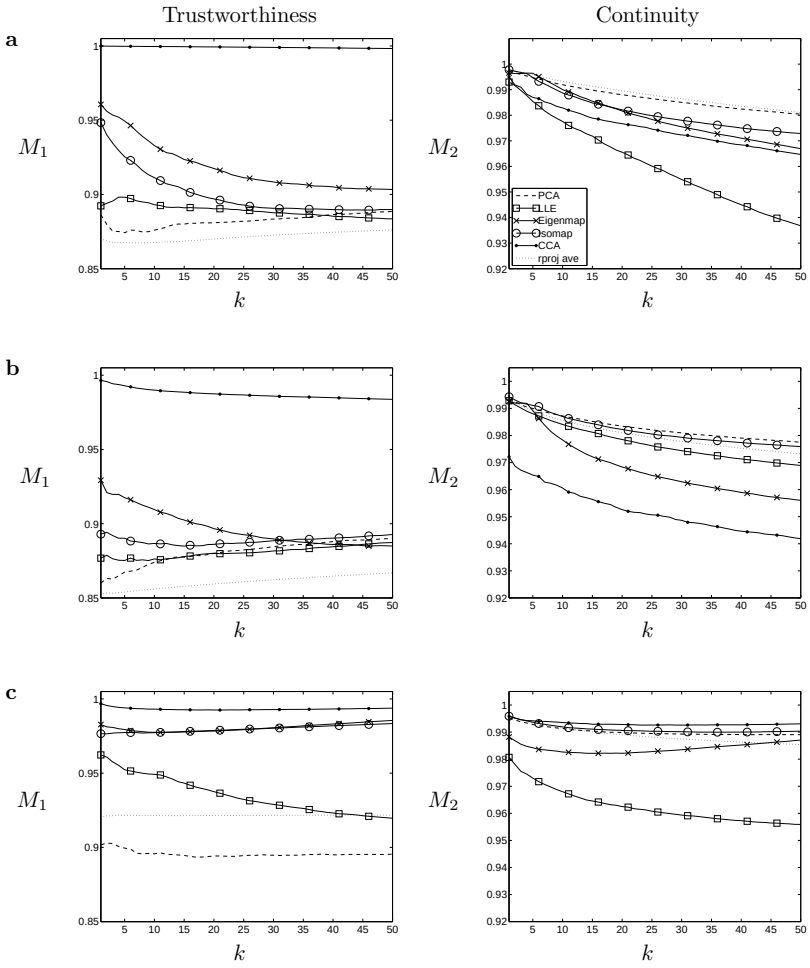
Trustworthiness
Continuity
a
b
c
M_1
M_2
k
PCA
LLE
Eigenmap
Isomap
CCA
rproj ave

Figure 5:

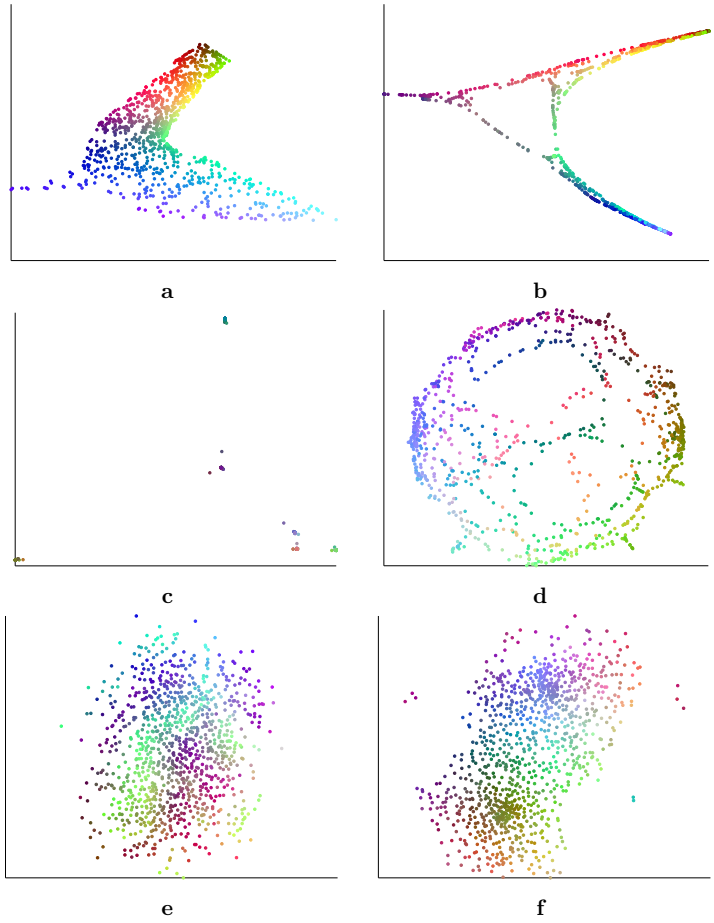

Figure 6:

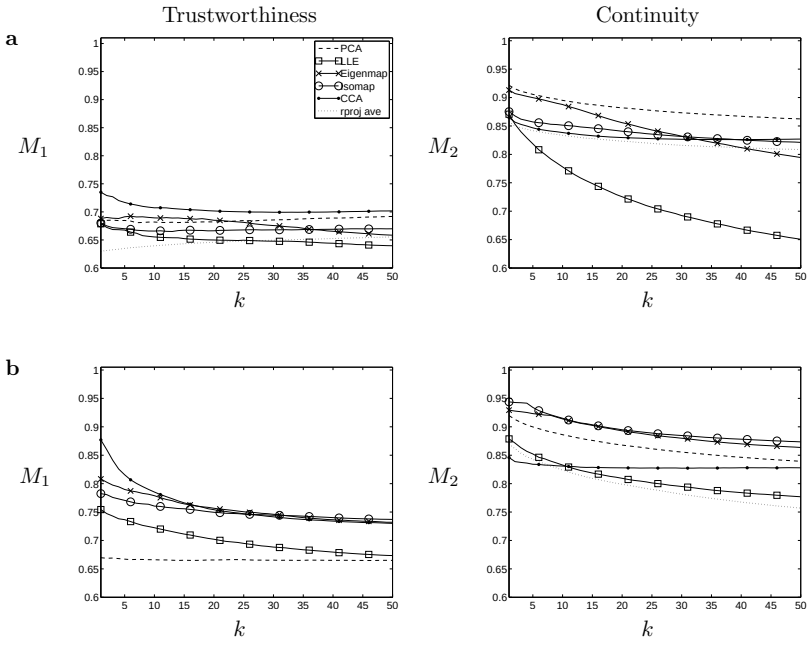
Trustworthiness
Continuity
a
b
M_1
M_2
k
PCA
LLE
Eigenmap
Isomap
CCA
rproj ave

Figure 7:

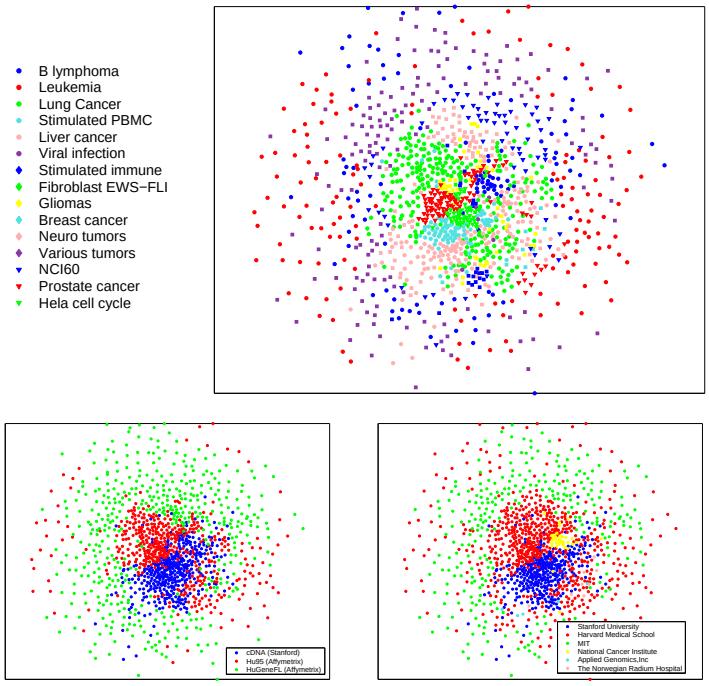
B lymphoma
Leukemia
Lung Cancer
Stimulated PBMC
Liver cancer
Viral infection
Stimulated immune
Fibroblast EWS−FLI
Gliomas
Breast cancer
Neuro tumors
Various tumors
NCI60
Prostate cancer
Hela cell cycle
cDNA (Stanford)
Hu95 (Affymetrix)
HuGeneFL (Affymetrix)
Stanford University
Harvard Medical School
MIT
National Cancer Institute
Applied Genomics,Inc
The Norwegian Radium Hospital

Figure 8:

Table 1:

Cancer type	**Data**	**CCA**	**PCA**
B lymphoma * (112)	66	83	60
Leukemia * (141)	45	90	79
Lung Cancer * (253)	78	84	67
Stimulated PBMC (82)	**99**	88	90
Liver cancer (137)	79	82	81
Viral infection (5)	50	50	51
Stimulated immune (24)	**100**	**92**	80
Fibroblast EWS-FLI (10)	**98**	**92**	**92**
Gliomas (47)	**91**	**91**	72
Breast cancer (8)	74	**93**	**97**
Neuro tumors (90)	78	84	72
Various tumors (154)	80	**91**	77
NCI60 * (65)	21	88	79
Prostate cancer (102)	**100**	**96**	**92**
Hela cell cycle (48)	**100**	**97**	**93**
Average	77	87	79
Platform			
cDNA (Stanford) * (394)	**96**	79	83
Hu95 (Affymetrix) * (434)	87	74	63
HuGeneFL (Affymetrix) * (450)	51	82	69
Average	78	79	72
Institution			
Stanford University * (346)	**97**	82	82
Harvard Medical School * (601)	76	65	56
MIT * (283)	47	88	74
National Cancer Institute (35)	**97**	**91**	**94**
Applied Genomics,Inc (5)	49	50	50
The Norwegian Radium Hospital (8)	75	**92**	**97**
Average	74	78	76

Table 2:

Cancer type vs Platform	cDNA (Stanford)	Hu95 (Affymetrix)	HuGeneFL (Affymetrix)
B lymphoma	cDNA 100		B lymphoma 70
Leukemia		Hu95 97	HuGeneFL 100
Lung Cancer	cDNA 100	Hu95 85	
NCI60	cDNA 100		NCI60 87
Cancer type vs Institution	Stanford University	Harvard Medical School	MIT
B lymphoma		Harvard 84	
Leukemia		Harvard 96	MIT 98
Lung Cancer	Stanford 100	Harvard 78	
NCI60			NCI60 98

Table 3:

Cancer type vs Platform	cDNA (Stanford)	Hu95 (Affymetrix)	HuGeneFL (Affymetrix)
B lymphoma	cDNA 95		HuGeneFL 96
Leukemia		Leukemia 94	HuGeneFL 55
Lung Cancer	cDNA 78	Hu95 69	
NCI60	cDNA 92		HuGeneFL 69
Cancer type vs Institution	Stanford University	Harvard Medical School	MIT
B lymphoma		Harvard 89	
Leukemia		Leukemia 94	MIT 57
Lung Cancer	Stanford 78	Harvard 70	
NCI60			MIT 61

Table 4:

Cancer type vs Platform	cDNA (Stanford)	Hu95 (Affymetrix)	HuGeneFL (Affymetrix)
B lymphoma	cDNA 92		HuGeneFL 82
Leukemia		Leukemia 87	Leukemia 58
Lung Cancer	cDNA 89	Hu95 83	
NCI60	cDNA 100		HuGeneFL 54
Cancert type vs Institution	Stanford University	Harvard Medical School	MIT
B lymphoma		Harvard 88	
Leukemia		Leukemia 84	Leukemia 57
Lung Cancer	Stanford 86	Harvard 80	
NCI60			NCI60 54