Helsinki University of Technology
Dissertations in Computer and Information Science
Espoo 2007 Report D20

DIMENSIONALITY REDUCTION FOR VISUAL EXPLORATION OF SIMILARITY STRUCTURES

Jarkko Venna

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Computer Science and Engineering for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 8th of June, 2007, at 12 o'clock noon.

Helsinki University of Technology
Department of Computer Science and Engineering
Laboratory of Computer and Information Science
P.O. Box 5400
FI-02015 TKK
Finland

Distribution:

Helsinki University of Technology
Laboratory of Computer and Information Science
P.O.Box 5400
FI-02015 TKK
Finland

Tel. +358 9 451 3272
Fax +358 9 451 3277
http://www.cis.hut.fi/

Available in pdf format at http://lib.hut.fi/Diss/2007/isbn9789512287529/

ISBN 978-951-22-8751-2 (printed version)
ISBN 978-951-22-8752-9 (electronic version)
ISSN 1459-7020

Multiprint Oy/Otamedia
Espoo 2007

Venna, J. (2007): **Dimensionality reduction for visual exploration of similarity structures.** Doctoral thesis, Helsinki University of Technology, Dissertations in Computer and Information Science, Report D20, Espoo, Finland.

Keywords: Dimensionality reduction, exploratory data analysis, information retrieval, information visualization, manifold learning, Markov Chain Monte Carlo.

ABSTRACT

Visualizations of similarity relationships between data points are commonly used in exploratory data analysis to gain insight on new data sets. Answers are searched for questions like: Does the data consist of separate groups of points? What is the relationship of the previously known interesting data points to other data points? Which points are similar to the points known to be of interest? Visualizations can be used both to amplify the cognition of the analyst and to help in communicating interesting similarity structures found in the data to other people.

One of the main problems faced in information visualization is that while the data is typically very high-dimensional, the display is limited to only two or at most three dimensions. Thus, for visualization, the dimensionality of the data has to be reduced. In general, it is not possible to preserve all pairwise relationships between data points in the dimensionality reduction process. This has lead to the development of a large number of dimensionality reduction methods that focus on preserving different aspects of the data. Most of these methods were not developed to be visualization methods, which makes it hard to assess their suitability for the task of visualizing similarity structures. This problem is made more severe by the lack of suitable quality measures in the information visualization field.

In this thesis a new visualization task, visual neighbor retrieval, is introduced. It formulates information visualization as an information retrieval task. To assess the performance of dimensionality reduction methods in this task two pairs of new quality measures are introduced and the performance of several dimensionality reduction methods are analyzed. Based on the insight gained on the existing methods, three new dimensionality reduction methods (NeRV, fNeRV and LocalMDS) aimed for the visual neighbor retrieval task, are introduced. All three new methods outperform other methods in numerical experiments; they vary in their speed and accuracy.

A new color coding scheme, similarity-based color coding, is introduced in this thesis for visualization of similarity structures, and the applicability of the new methods in the task of creating graph layouts is studied. Finally, new approaches to visually studying the results and convergence of Markov Chain Monte Carlo methods are introduced.

Venna, J. (2007): **Dimensionalisuuden pienentäminen samankaltaisuuksien visuaalista tarkastelua varten.** Väitöskirja, Teknillinen korkeakoulu, Dissertations in Computer and Information Science, Raportti D20, Espoo, Suomi.

Avainsanat: Dimensionaalisuuden pienentäminen, eksploratiivinen dataanalyysi, informaation visualisointi, Markov-ketju Monte Carlo, monistojen oppiminen, tiedonhaku.

TIIVISTELMÄ

Samankaltaisuussuhteiden visualisointia käytetään eksploratiivisessa data analyysissä usein ensimmäisenä askeleena uuden datajoukon tarkastelussa. Tavoitteena on muodostaa alustava käsitys datan rakenteesta ja tuottaa vastaus kysymyksiin kuten: Jakautuuko data erillisiin ryhmiin? Mikä on aiemmin havaittujen kiinnostavien datapisteiden suhde uusiin tuntemattomiin datapisteisiin? Mitkä pisteet ovat samankaltaisia kuin kiinnostaviksi tiedetyt pisteet? Visualisointi voi sekä helpottaa datan analyysiä että auttaa havaittujen rakenteiden kommunikoinnissa.

Informaation visualisoinnissa data on tyypillisesti korkeaulotteista. Tämä on ongelmallista, koska näytöllä ei pystytä esittämään kuin korkeintaan kolme dimensiota kerrallaan. Tästä syystä datan dimensionaalisuus on saatava pudotettua kahteen tai kolmeen visualisointia varten. Dimension pienentämisestä seuraa lähes aina jonkinlaisia virheitä; ei ole mahdollista säilyttää kaikkia datassa esiintyviä samankaltaisuusuhteita ennallaan vaan informaatiota katoaa ja vääristyy. Eri dimensionaalisuuden pienennysmenetelmät pyrkivätkin säilyttämään datan eri ominaisuuksia. Ongelmana informaation visualisoinnissa on, että suurinta osa dimensionaalisuuden pienennysmenetelmistä ei ole kehitetty visualisointia varten, minkä vuoksi niillä tuotettujen kuvien laatu on varmistettava. Vaikeaksi laadun varmistamisen tekee sopivien mittareiden puute.

Tässä väitöskirjassa esitetään uusi visualisointitehtävä, visuaalinen naapureiden haku. Siinä informaation visualisointi hahmotetaan informaation hakutehtäväksi. Tätä formulaatiota käytetään muodostamaan kaksi paria uusia visualisoinnin laatumittareita, ja useiden dimensionaalisuuden pienennysmenetelmien soveltuvuutta tähän uuteen visualisointitehtävään tutkitaan. Tulosten pohjalta saatuja ideoita käytetään kolmen uuden dimensionaalisuuden pienennysmenetelmän luomiseen (NeRV, fNeRV ja LocalMDS). Kaikki kolme menetelmää ovat päihittäneet muut menetelmät numeerisissa kokeissa. Toisistaan ne eroavat nopeudessa ja tarkkuudessa.

Lisäksi tässä työssä esitetään uusi menetelmä värien allokoimiseksi datapisteille, samankaltaisuuksiin perustuvä värikoodaus, ja uusien menetelmien soveltuvuutta graafien visualisointiin testataan. Lopuksi tarkastellaan Markov-ketju Monte Carlo menetelmien konvergenssin ja tulosten visualisointia.

Contents

PREFACE

This work has been carried out in the Neural Networks Research Centre / Adaptive Informatics Research Centre of the Laboratory of Computer and Information Science at the Helsinki University of Technology. It has been funded by the Academy of Finland through the Centre of Excellence programme and projects mentioned in the publications and by the European Union under the PASCAL Network of Excellence, IST-2002-506778. Additionally, I have received funding from the graduate school of the Department of Computer Science of Helsinki University of Technology.

Special thanks go to my instructor and supervisor Professor Samuel Kaski, who has taught me everything I know about scientific work. His contribution to my work is evident from the publications in this thesis.

I express my gratitude to Professor Erkki Oja, Academician Teuvo Kohonen and Professor Olli Simula for having had the possibility and privilege of working at the research centre.

I wish to thank Professor Samuel Kaski, Dr. Kai Puolamäki and the pre-examiners, Professor Pasi Fränti and Dr. Oleg Okun for their comments on the manuscript of my thesis. Their efforts have helped me improve it significantly.

I am indebted to all my co-authors Samuel Kaski, Teuvo Kohonen, Janne Nikkilä, Jaakko Peltonen, Merja Oja, Petri Törönen and Eero Castrén. I am honored to have had the opportunity to work with Academician Teuvo Kohonen. Additional thanks belong to the MI group and to the rest of the laboratory for providing an enjoyable and exciting atmosphere. Especially I wish to thank my friend and co-worker Jarkko for the long discussions that have helped me get over difficult times both at work and outside it.

Finally, thanks to my parents for their support and most of all I thank my wife Marjo and my daughters Riikka, Helmi and Vappu.

Otaniemi, May 4, 2007
Jarkko Venna

LIST OF PUBLICATIONS

The thesis consists of an introduction and the following publications:

1. Samuel Kaski, Jarkko Venna, and Teuvo Kohonen. Coloring that reveals cluster structures in multivariate data. *Australian Journal of Intelligent Information Processing Systems*, 6:82–88, 2000.

2. Jarkko Venna and Samuel Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. In Georg Dorffner, Horst Bischof, and Kurt Hornik, editors, *Artificial Neural Networks - ICANN 2001*, Vienna, Austria, August 21–25, pp. 485–491. Springer, Berlin, 2001.

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

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

5. Jarkko Venna, and Samuel Kaski. Local multidimensional scaling. *Neural Networks*, 19:889–899, 2006.

6. Jarkko Venna, and Samuel Kaski. Visualizing gene interaction graphs with local multidimensional scaling. In Michel Verleysen, editor, *Proceedings of the 14th European Symposium on Artificial Neural Networks (ESANN'2006)*, Bruges, Belgium, April 26–28, pp. 557–562, d-side, Evere, Belgium, 2006.

7. Jarkko Venna, and Samuel Kaski. Nonlinear Dimensionality Reduction as Information Retrieval. In Marina Meila and Xiaotong Shen editors, *proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, San Juan, Puerto Rico, March 21-24, pp. 568–575, 2007.

8. Jarkko Venna and Samuel Kaski. Visualizing high-dimensional posterior distributions in Bayesian modeling. In O. Kaynak, E. Alpaydin, E. Oja, L. Xu, editors, *Artificial Neural Networks and Neural Information Processing–Supplementary proceedings ICANN/ICONIP 2003*, Istanbul, Turkey, June 26–29, pp. 165-168. 2003

9. Jarkko Venna, Samuel Kaski and Jaakko Peltonen. Visualizations for Assessing Convergence and Mixing of MCMC. In N. Lavrac, D. Gamberger, H. Blockeel, L. Todorovski, Editors, *Proceedings of the 14th European Conference on Machine Learning (ECML 2003)*, Cavtat-Dubrovnik, Croatia, September 22–26, pp. 432-443. Springer, Berlin, 2003.

THE AUTHOR'S CONTRIBUTION

Most of the ideas presented in the publications have been developed as team work, mostly in collaboration with Samuel Kaski. Giving individual credit is therefore, to a large part, not meaningful.

Publication 1 describes a new way of developing color scales for data visualization. It was Teuvo Kohonen who had the original idea of coloring the Self-Organizing Map. The projection method used to perform the color coding was developed jointly with Samuel Kaski and the author was responsible for selecting and dealing with the color space. The author was also responsible for making all the experiments. The insights gained from the work on developing the color coding method were the initial motivation that lead to the trustworthiness and continuity measures and the new dimensionality reduction methods developed later on.

In Publication 2 the trustworthiness and continuity measures are introduced. The original idea of developing a new topology preservation measure which would be more expressive when comparing the Self-Organizing Map to other dimensionality reduction methods, came from Samuel Kaski. The author was mostly responsible for developing the details of the measures as well as running all the experiments.

In Publication 3 the trustworthiness and continuity measures are applied for the first time in a visualization context. Additionally, a method for improving the quality of a visualization based on the trustworthiness measure was introduced. The author was mostly responsible for developing ways to apply the two measures to new forms of visualizations, developing the method for improving the visualizations, as well as running the experiments relating to them. The biological content and the work on the new SOM approach was done by the other authors of the paper.

In Publication 4, the trustworthiness and continuity measures are used to analyze the performance of a large set of dimensionality reduction methods in a visualization context. Additionally, the feasibility of developing a graphical user interface to a gene expression data bank is tested. The author developed the toy data sets used in the experiments, ran the experiments, and analyzed the results. The idea of making the feasibility study was Samuel Kaski's. In this paper the first sketch of the connection between the trustworthiness and continuity measures on the one hand and the precision and recall measures of information retrieval on the other, is introduced.

In Publication 5 a new visualization method, Local Multidimensional Scaling, is introduced. Samuel Kaski brought out the notion that it would be beneficial to have a method where the tradeoff between trustworthiness and continuity could be selected with a parameter. The author was responsible for the original idea leading to the method as well as doing most of the development work. The author was also mainly responsible for designing and running the experiments and analyzing the results.

In Publication 6 Local MDS is applied to graph visualization. The idea of applying local MDS to graphs was the author's. The author also developed the graph versions of the trustworthiness and continuity measures and carried out all the experiments.

In Publication 7, the visual neighbor retrieval task is defined and the connection between dimensionality reduction and information retrieval is formulated. This paper also introduces the Neighbor Retrieval Visualizer (NeRV) algorithm, a new dimensionality reduction method, aimed at the visual neighbor retrieval task. The connection between information retrieval and information visualization as well as the visual neighbor retrieval task were developed jointly by the author and Samuel Kaski. The NeRV algorithm was developed by the author. The author also ran all the experiments.

In Publication 8 a new method for visualizing high-dimensional posterior distributions is introduced. The idea of using the Fisher metric to make the visualization independent of parameterization of the model was developed by Samuel Kaski. The author was mainly responsible for developing the method and experiments.

In Publication 9 a visual approach to analyzing convergence problems in MCMC simulations is described. The author was responsible for noting the connection between Linear Discriminant Analysis (LDA) and a commonly used convergence measure, and for developing the visualization approach based on LDA. The RCA method was developed by Samuel Kaski and Jaakko Peltonen. The experiments were selected and run by the author and Jaakko Peltonen.

The writing of the actual text in the publications has in all cases been a very collaborative work.

1 INTRODUCTION

This thesis is about studying, applying, and developing dimensionality reduction methods for exploratory visualization of similarity structures. Similarity is usually defined as a relation between two data points. In this work we are interested in larger structures, with more than two data points. These can be for example groups of similar data points or manifolds along which the data points change by small differences from more similar points to very different ones. An illustration of the visual analysis of similarities is given in Figure 1. A set of data points in a high-dimensional space is mapped to a low-dimensional output space for visualization. From the visualization it is easy to identify the set of five points that are the most similar ones to the yellow point. Additional information about the overall similarity structure of the data set is readily available from the visualization. It is easy to see that there are two or three separate groups of data points and there is one outlier (magenta), a data point that is different from the rest of the data set. This kind of an analysis gives an idea of the overall structure of the data and can help in raising new questions like: What has caused the one data point to become an outlier?

INPUT SPACE

Point	Var 1	Var 2	Var 3	Var 4
1	5.1	3.5	1.4	2.3
2	5.4	3.7	1.5	0.2
3	5.4	3.4	1.7	0.2
4	4.8	3.1	1.6	0.2
5	5.0	3.5	1.3	0.3
6	7.0	3.2	4.7	1.4
7	5.0	2.0	3.5	1.0
8	5.9	3.2	4.8	1.8
9	5.5	2.4	3.8	1.1
10	5.5	2.6	4.4	1.2
11	6.3	3.3	6.0	2.5
12	6.5	3.2	5.1	2.0
13	6.9	3.2	5.7	2.3
14	7.4	2.8	6.1	1.9
15	6.7	3.1	5.6	2.4

OUTPUT SPACE

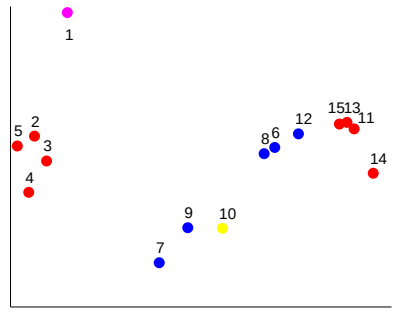


Figure 1: A four-dimensional data set is mapped from the input space to the two-dimensional output space for visualization. Blue points are the points most similar to the yellow point.

This kind of an analysis is called exploratory. The term *exploratory data analysis* refers to the task of creating new hypotheses and gaining insight by analyzing data. It differs from *confirmatory data analysis* where the goal is to test the validity of a given hypothesis. A task commonly used as a part of exploratory data analysis is *information visualization* which can be defined as follows:

The use of computer-supported, interactive, visual representations of abstract data to amplify cognition [28].

Information visualization is a subfield of visualization that concentrates on abstract data. The goal of visualization is to amplify cognition, that is, to allow the user to find aspects of the data that would be hard to notice otherwise. Information visualization differs form *scientific visualization*, another subfield of visualization, in that the data need not have a natural visual form or physical meaning. A typical example of scientific visualization is a flow visualization that displays how liquid or gas flows around an object.

One point in the definition that could be argued against is the inclusion of interaction. In fact, there are situations where interaction can make the visualization harder to use [147]. In this work the focus is on noninteractive visualizations that could also be used as part of an interactive visualization system but that do not rely on the ability of the user to manipulate the display.

Zhang [170] points out three tasks that are common in information visualization:

- **Information retrieval tasks:** either looking up values of parameters or finding relevant objects.
- **Comparison tasks:** comparison of values in one attribute or between attributes.
- **Integration tasks:** finding patterns in the data by combining aspects of the data through visualization.

Many basic information visualization methods can be used for all of these tasks. For example, a scatter plot (see Figure 2**a**) can be used for value lookup by reading the value from the axis. Correlations of two variables can be seen as line-like patterns in the plot and clusters can be seen as separate point clouds on the display. When the goal is to find more complex relationships that include more than two variables the methods become more complex and it is not always possible to perform all three tasks using just one visualization.

Several approaches have been developed for visualizing high-dimensional data. Many of the methods like parallel coordinates [75], Star glyphs [31], Chernoff faces [33], and stacked dimensions [97] try to show all dimensions of the data at the same time (for some examples see Figures 2**b**–**d**). This approach is only suitable for relatively few dimensions. When the dimensionality of the data increases, some other means have to be used.

One of the main strategies used to handle very high dimensional data in information visualization is *dimensionality reduction* where the task is to reduce the dimensionality of the data to two or three for visualization. A large number of different methods have been developed for this task. Most of these methods are not aimed at visualization, however. In many cases the goal of the dimensionality reduction method is to reduce the dimensionality to the point where it matches the intrinsic dimensionality of the data. The dimensionality of the low-dimensional manifold that the data is assumed to

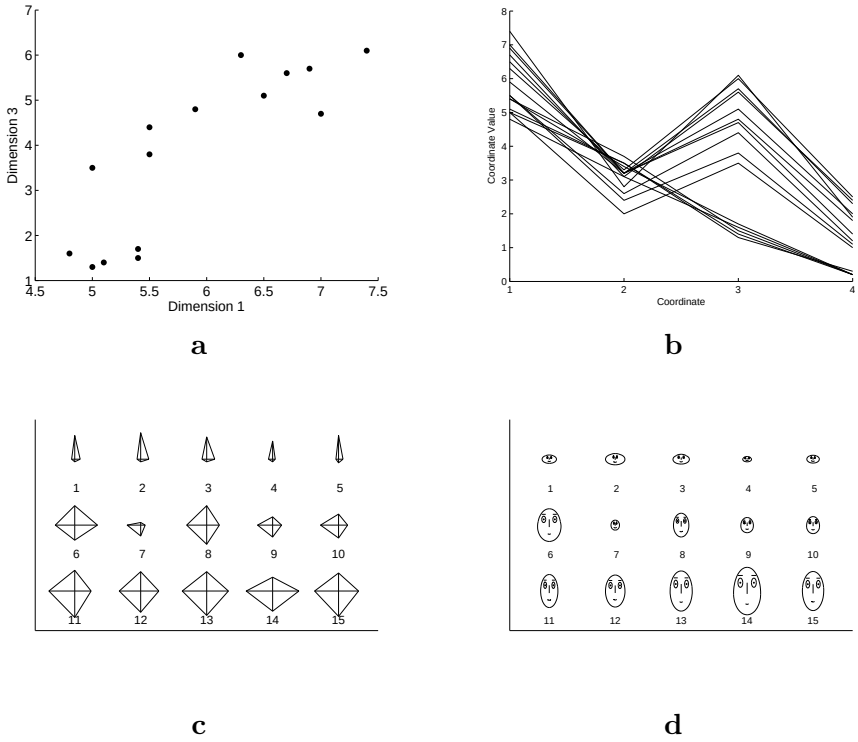


Figure 2: A four dimensional data set of 13 data points is visualized with four methods. **a**) Scatter plot of data on dimensions 1 and 3, **b**) Parallel coordinates plot, **c**) Star plot, and **d**) Chernoff faces.

lie on in the high-dimensional space. In a visualization context the output dimensionality is constrained by the display to only two or three, which is usually lower than the intrinsic dimensionality of the data. This difference in goals can cause problems in some cases and the performance of each method should be carefully tested to find out if it is suitable for visualizing the data at hand.

In a visualization context trying to assess the quality of a visualization typically leads to usability studies that are difficult to design and can be costly. Although this is the only way to find out how well a human can utilize the visualization it is possible to use quality measures to verify that the data represented by the visualization is accurate. Unfortunately there is a lack of suitable quality measures in the information visualization field for assessing the quality of the results of dimensionality reduction.

The focus of this work is on dimensionality reduction methods in the context of information visualization. Unlike the more specific multidimensional visualization methods, dimensionality reduction provides a more general framework which works even on very high-dimensional data sets. While many dimensionality reduction methods have been used for visualization there has been very little work on analyzing how well they perform in this context. In this work a new visualization task focusing on studying similarity structures is introduced, and new quality measures to assess the quality of visualizations are defined. The quality measures are used to analyze the behavior of several dimensionality reduction methods. Based on the insights gained from the analysis, new methods are proposed. Finally some applications illustrating the use of dimensionality reduction for visualization are discussed.

In the following sections the term method is used to refer to a way of transforming the data. Typically a method is defined by a cost function that is optimized, but it can also be defined by an iterative algorithm. A measure, on the other hand, is a function that gives a quantitative evaluation on how good the visualization is for the given task. Thus a measure is always connected to a specific visualization task. In some cases the same function can be either a quality measure or the cost function of a method. Whether it is referred to as a method or a measure depends on the context. Is it used to transform data or to evaluate the results?

2 METHODS FOR DIMENSIONALITY REDUCTION

One of the main problems in information visualization is the fact that while the data typically is high-dimensional, only two or three dimensions can be easily shown in an image. There have been many different approaches that try to solve this problem. Possibly the simplest method to visualize high dimensional data is to create a set of images where each image shows the scatter plot of the data on two dimensions at a time. The images can be collected to a scatter plot matrix [35] (see Figure 3) where the images are arranged into a grid formation. Scatter plots are fairly simple to interpret and by using color, shape, or some other method for distinguishing different points, it is possible to add more information, such as the class or type of the data point, to the plot. There is one major drawback in this approach, however. The number of images in the scatterplot matrix increases rapidly with the number of dimensions in the data. Additionally, there is usually no way of knowing which dimensions are the most relevant ones. Thus there is no prior information on which images the user should check first.

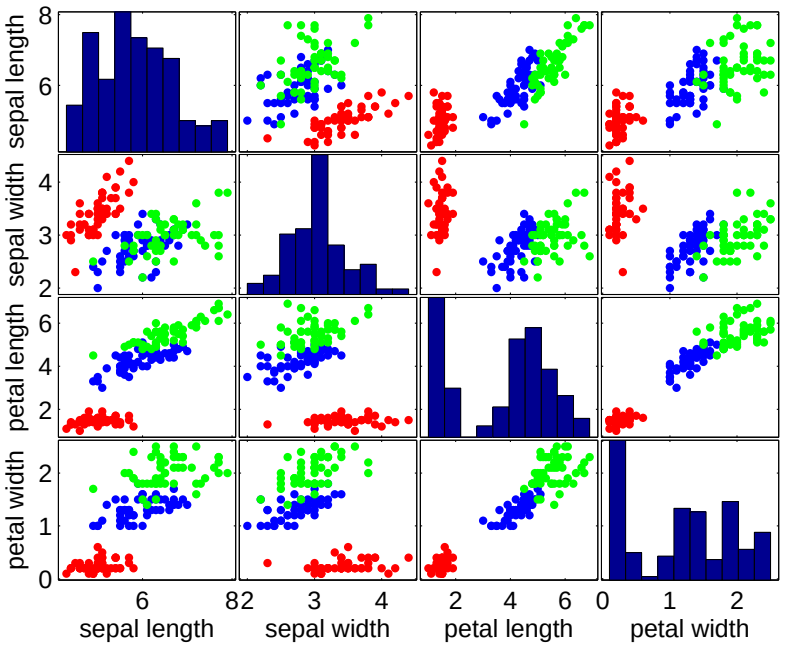
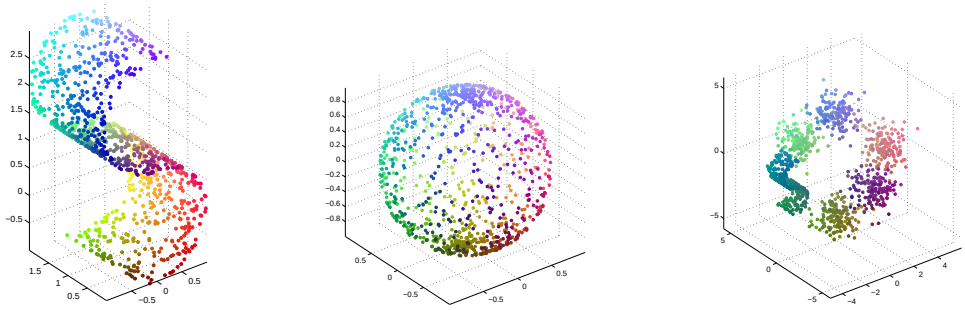


Figure 3: A scatter plot matrix of the iris data set [50]. The colors indicate the three classes in the data. A histogram of the variable is included on the diagonal where both axis would be the same.

To solve the problems connected with the simple straightforward approach a large number of methods have been introduced. Traditionally the methods

have mainly followed two principles. They have either relied on linear projections or they have tried to produce a nonlinear mapping that preserves pairwise distances between data points. A third approach has been gaining popularity recently. The manifold learning methods expect the data to lie on a low dimensional manifold in the high dimensional input space, and their goal is to find and unfold this manifold. In addition to these three main approaches there is a large number of methods that do not fit well in any of the three main categories. The groupings are not crisp, that is, a method could be classified into many of the groups at the same time. For example a distance preserving mapping can be designed in a way that it unfolds a nonlinear manifold, thus making it both a distance preserving mapping and a manifold learning method.

The main methods belonging to the three main groups, together with a few methods based on other approaches, are discussed in more detail below. In each case three toy data sets are visualized for illustration (Figure 4). Each data set contains 1000 data points in a three-dimensional space. The first data set consists of an S-shaped two-dimensional manifold. The second data set contains data that is distributed on the surface of a sphere and the third data set contains six clusters, of which five are spherical Gaussians and one is an S-shaped two-dimensional manifold. These data sets were selected to bring out differences in the behavior of the methods and to point out some of their shortcomings. In each case the method was run with several parameter values and the result producing the best trustworthiness (see section 3.3) was selected. A more profound analysis of the behavior of many of the methods introduced here can be found in Publications 4 and 5.



a **b** **c**

Figure 4: Three toy data sets used to illustrate the behavior of the various dimensionality reduction methods. Projection directions have been hand picked. Each data point is color coded according to its position in the original data space. The colors were defined by first scaling the data set to fit within the unit cube and then each coordinate axis was associated with one of the axes of the RGB-color space. **a**) S-shaped manifold, **b**) sphere, and **c**) clusters.

2.1 Traditional approaches

2.1.1 Linear methods

The approach used to develop many linear projection methods for visualization is well illustrated in the *projection pursuit* [52] algorithm. The goal is to find the linear projection that optimizes an index of usefulness. In the projection pursuit method this index is defined based on a combination of the local density and overall spread of the data. More generally the index of usefulness depends on what is desired from the projection images and it has been defined in several ways in different methods. For example, the goal can be to preserve distances [137], to maximize variance [74], or the cost function of a nonlinear method can be applied as the index of usefulness to create a new linear method [68, 69]. It is also possible to use some additional information to define what is interesting in the data like in the Linear Discriminant Analysis (LDA) [142] and Relevant Component Analysis (RCA) [119] which are discussed more closely in Section 5. By changing the projection direction smoothly an animation can be created. An animation of projection directions is called *the grand tour* [6]. The problem with the grand tour approach is that the length of the animation grows very fast as the dimensionality of the data increases. One of the main benefits of all linear methods, whether animated or static, is that they are easy to understand and interpret.

Principal Component Analysis (PCA). The goal of PCA [74] is to find linear projections that maximally preserve the variance in the data. More technically, the projection directions can be found by solving for the eigenvalues λ and eigenvectors $\mathbf{a}$ of the covariance matrix $\mathbf{C}_x$ of the data $\mathbf{X} \in \mathbb{R}^{d \times N}$, where d is the dimensionality of the data and N is the number of data points.

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

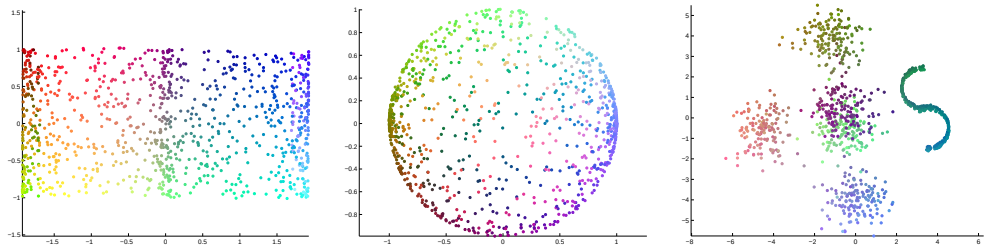
The data points $\mathbf{x}_i$ can then be visualized by projecting them with

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

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 one of the first methods that is usually tried when visualizing new data. It produces images that are easy to interpret and works quite well when the variance in the data is mainly concentrated in only a few directions. Like all linear methods it can have problems when the data lies on a nonlinear manifold. The results from projecting the three toy data sets (Figure 5) are typical of linear projections. The data is in effect squashed flat and this results in areas where points that are originally far from each other are close by in the visualization. This is most prominent on the visualizations of the S-shaped

manifold and the sphere data set (Figures 5**a** and **b**). On the other hand, the rectangular shape of the S-manifold and the circular shape of the sphere are easy to see from the images. The PCA projection of the clusters data set (Figure 5**c**) shows that while the PCA often works quite well, it can also produce quite bad results in some cases. In this case the maximal variance is in a direction that causes two of the clusters to be mixed together in the projection. Actually, a totally random linear projection works, on the average, better on this data set than PCA. A more profound analysis of the behavior of the method on different data sets can be found in Publications 4 and 5.



a **b** **c**

Figure 5: PCA projections of the three toy data sets. **a**) S-shaped manifold, **b**) sphere and **c**) clusters.

Linear Multidimensional Scaling (MDS) [63, 143], also called Classical Scaling, uses an approach that is more commonly used with nonlinear mapping methods. It starts by calculating the squared Euclidean distance matrix $\mathbf{D}$ of the data. The goal is to find a representation of points that preserves these distances.

First the squared distance matrix is centered to produce a new matrix

$$\mathbf{B} = -\frac{1}{2}\left(\mathbf{I} - \frac{\mathbf{1}_N^T\mathbf{1}_N}{N}\right)\mathbf{D}\left(\mathbf{I} - \frac{\mathbf{1}_N^T\mathbf{1}_N}{N}\right). \tag{3}$$

This fixes the location of the configuration so that it is centered around zero [63]. After centering the configuration of points is found by solving for the eigenvectors of B. The coordinates of the data points in the d-dimensional output space can be constructed with the following equation.

$$\mathbf{y}_j = \{\sqrt{\lambda_1}a_{1j}, \sqrt{\lambda_2}a_{2j}, \ldots, \sqrt{\lambda_d}a_{dj}\}. \tag{4}$$

Here the λ_i are the d largest eigenvalues and a_{ij} is the j:th component of the i:th eigenvector.

Linear MDS is very closely related to PCA. It can be shown [63] that when the dimensionality of the solution space 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 squared Euclidean distance matrix

of the data. Thus the cost function of 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.

2.1.2 Nonlinear methods

Traditionally nonlinear methods have been distance preserving mappings, or Multidimensional Scaling (MDS) methods as they are often called. Their goal is to find a configuration of points that reproduces a given pairwise distance matrix. The distance matrix can be calculated from data points or it can result from a direct evaluation of similarities between objects. The latter is the case, for example, in market studies when people are asked to assess how similar two products are. In such a case only the pairwise similarities are known and not the underlying variables. Traditionally, finding the underlying dimensionality of the data has been an additional goal for these methods, but for visualization the dimensionality is restricted to two or three.

Traditional Multidimensional Scaling. There are several different variants of Multidimensional Scaling (MDS) [19], but they all have a common goal: to find a configuration of points that preserves the pairwise distance matrix of the data.

The simplest nonlinear Multidimensional Scaling method is metric MDS. Its cost function [91], called the *raw stress*, is

$$E = \sum_{ij} (d(\mathbf{x}_i, \mathbf{x}_j) - d(\mathbf{y}_i, \mathbf{y}_j))^2, \tag{5}$$

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. Variables $\mathbf{y}_i$ and $\mathbf{y}_j$ are the representations (locations) of points $\mathbf{x}_i$ and $\mathbf{x}_j$ in the output space. The cost function is minimized in respect to the representations $\mathbf{y}_i$.

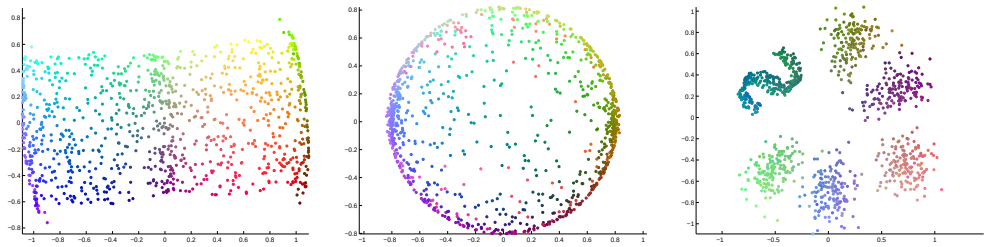
There are a number of different variants of MDS, but all have a cost function that is basically of the same form. Sammon's mapping [129] gives small distances a larger weight:

$$E = \sum_{ij} \frac{(d(\mathbf{x}_i, \mathbf{x}_j) - d(\mathbf{y}_i, \mathbf{y}_j))^2}{d(\mathbf{x}_i, \mathbf{x}_j)}, \tag{6}$$

and in non-metric MDS [91] the distances are modified by a monotonic function. There are also probabilistic variants that define a probabilistic model either for the distances [114] or for the latent positions defining the distances [174].

The behavior of metric MDS is illustrated in Figure 6. Metric MDS focuses on the preservation of long distances. A error of one percent in a long distance is considered to be more severe by the cost function than a similar error in a

short distance. This results usually in images where the global structure of the data is well presented. The clusters in the cluster data set are well separated and the sphere results in a nice circular disc. The disadvantage is that the local structure suffers which can be seen from the overlapping areas in the S-shaped manifold. The longer pairwise distances do not allow the manifold to be unfolded.



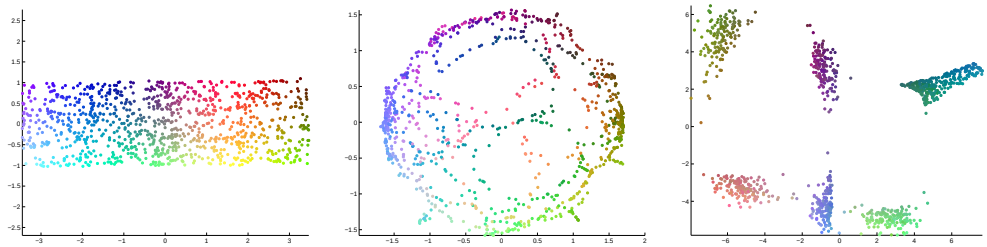
a **b** **c**

Figure 6: Metric MDS projections of the three toy data sets. **a**) S-shaped manifold, **b**) sphere and **c**) clusters.

Isomap. Although the Isomap [141] algorithm was originally developed as a manifold learning method it can be described as 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 assumes that the data lies on a low-dimensional manifold in the input space. Instead of the usual direct pairwise distances used in MDS it uses geodesic distances on the manifold as input. An unfolding of the manifold happens when these geodesic distances are represented with Euclidean distances in the output space. The unfolding effect can be illustrated with a string that has knots tied to it. Each knot represents a data point and the geodesic distance between knots is the distance measured along the string. To make the geodesic distances coincide with the Euclidean distances the string needs to be pulled straight and thus the manifold becomes unfolded.

The geodesic distances are approximated with the shortest path distances calculated along the k-nearest-neighbor graph. Each data point forms a node in the graph and its k-nearest neighbors are connected to it with edges. The weight of each edge is the Euclidean distance between the data points at the ends. The actual embedding of points is found by linear MDS, applied to the shortest-path distance matrix.

It has been shown [17] that this algorithm is asymptotically able to recover certain types of manifolds, like the S-shaped manifold in Figure 7**a**. It is not usually able to cut open closed manifolds like a sphere, however. The exception is the case where the k-nearest neighbor graph accidentally happens to create a break that allows the manifold to be unfolded. The projection of the sphere data set also shows a typical artefact that is produced by approximating



a **b** **c**

Figure 7: Isomap projections of the three toy data sets. **a**) S-shaped manifold (k=20), **b**) sphere (k=4) and **c**) clusters (k=67).

geodesic distances with graph distances. The projection contains "holes" that are caused by overestimation of the manifold distances.

The basic Isomap algorithm has been extended in several ways. De Silva and Tenenbaum [41] developed a method that adjusts the graph distances depending on the density of the data. This has the effect that dense areas get spread out and sparse areas get more concentrated. Yang [164] proposed that the linear MDS step should be replaced by Sammon's mapping. Isomap has also been formalized as Kernel PCA [131] where the centered distance matrix takes the role of the kernel matrix [15, 34, 66]. This formulation allows new samples to be added after the initial configuration has been calculated. A slightly different solution for the out of sample problem has been proposed by Law and Jain [93].

Curvilinear Component Analysis (CCA). The CCA algorithm [44] is a variant of MDS. The starting point is a random initialization of points in the low-dimensional output space, and a pairwise distance matrix between the original data points. It differs from the formulation of traditional MDS in that it does not try to preserve all distances. It concentrates on preserving only the distances of points that are close to each other in the *output space*. The cost function

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

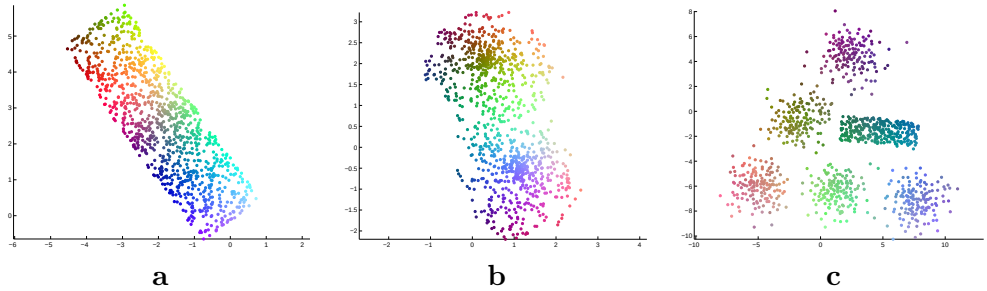
measures preservation of the original pairwise distances, weighted by a coefficient F that depends on the distance between the points in the output space.

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

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

The cost function is optimized using a stochastic gradient descent algorithm. In the beginning of optimization the radius of the area of influence, σ, is kept large enough to cover all or at least most of the data points. During the optimization it is slowly decreased to zero. This reduction of the radius causes the method to first find a global order for the whole set of data points, and then gradually focus on representing well those distances that are local in the output space.

The behavior of CCA is illustrated in Figure 8. The most noteworthy differences from the results of metric MDS are the unfolding of the S-curves in the S-shaped manifold and clusters datasets, and the splitting of the sphere into two halves. The splitting of the sphere allows the method to present the local structure better at the cost of missing some neighborhood relationships inherent in the data. This is similar to using a cartographic projection to study the map of the Earth instead of looking at a transparent globe.



a **b** **c**

Figure 8: CCA projections of the three toy data sets. **a**) S-shaped manifold, **b**) sphere and **c**) clusters.

An extension of CCA, Curvilinear Distance Analysis (CDA), was recently introduced by Lee et al. [99, 100]. The idea 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.

2.2 Manifold learning methods

Manifold learning methods are based on the assumption that the data lies on a low-dimensional nonlinear manifold in the high-dimensional input space. A two-dimensional manifold, for example, can be though of as a surface on which the data lies in the high-dimensional space. The goal of the manifold learning method is then to find and unfold this manifold. In the manifold learning task the output dimensionality would be selected to be the intrinsic dimensionality of the manifold but for visualization the output dimensionality has to be set to two or three *independent* of the dimensionality of the data manifold. This can cause problems to some of the manifold learning methods.

2.2.1 Locally Linear Embedding (LLE)

The LLE [128] algorithm is based on the assumption that the data manifold is sampled densely enough and is smooth enough so that we can make a locally linear approximation of it. If the assumption holds then each data point lies in or close to a locally linear subspace on the manifold. The geometry of this local subspace can be captured by calculating the linear coefficients that reconstruct each data point from its k nearest neighbors. In effect the set of neighbors defines a local coordinate system for the data point. Unless the dimensionality of the data manifold is smaller than k the point can not be reconstructed from its neighbors exactly and a reconstruction error has to be minimized. In LLE the reconstruction error is defined as

$$E(\mathbf{W}) = \sum_i \|\mathbf{x}_i - \sum_j W_{ij}\mathbf{x}_j\|^2, \tag{9}$$

where $\sum_j W_{ij}\mathbf{x}_j$ is the reconstruction of the point $\mathbf{x}_i$. 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 $\sum_j W_{ij} = 1$.

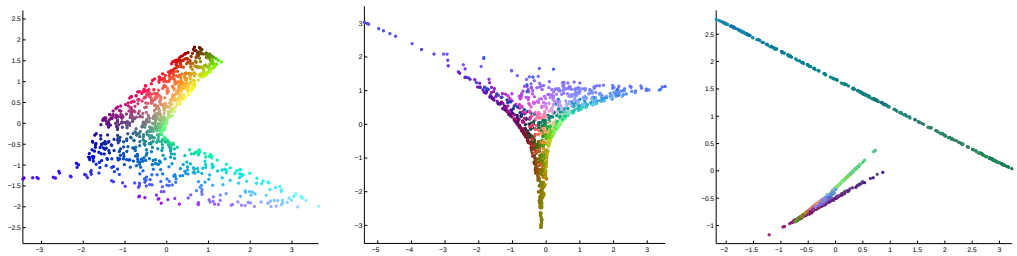
To unfold the manifold the dimensionality of the data has to be reduced to the dimensionality of the manifold. This is achieved by finding the low-dimensional coordinates that preserve the local coordinate systems as well as possible. More specifically, the configuration of points is found by fixing the weight matrix $\mathbf{W}$ and minimizing the cost function

$$E(\mathbf{Y}) = \sum_i \|\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j\|^2 , \tag{10}$$

where $\mathbf{y}_i$ is the low-dimensional representation of the data point x_i. The problem can be solved by finding the $p+1$ smallest eigenvalues of the matrix $(\mathbf{I}-\mathbf{W})^T(\mathbf{I}-\mathbf{W})$, where p is the dimensionality of the output [128]. The smallest eigenvalue corresponds to a constant eigenvector and the next p give the coordinates of the data points within the output space.

The result of LLE can be very sensitive to the selection of the parameter k, the number of neighbors used. If k is too small then the the graph formed by the neighbors gets split into unconnected parts and the result is a projection where all points are projected to only a few locations. The number of locations depends on the number of connected components in the graph. On the other hand a large k makes the method unable to unfold nonlinear manifolds. Results produced on the toy data sets are shown in Figure 9.

LLE can also be formulated in a way that takes the pairwise distance matrix of the data instead of the data matrix as the input [130]. The same formulation also leads to a kernelized version of LLE [43]. Other variants include supervised LLE [39] and Local Fisher Embedding [40] both of which are supervised versions of the LLE algorithm. It is also possible to formulize LLE as kernel PCA [66], and as with Isomap this formulation can be used



a **b** **c**

Figure 9: LLE projections of the three toy data sets. **a**) S-shaped manifold, (k=7) **b**) sphere (k=6) and **c**) clusters (k=66).

to develop an out of sample extension of LLE [15]. An incremental learning approach can also be used to add new data points to an existing LLE solution [90]. Additionally the cost function of LLE has been used to create a linear projection method [68]. Finally Wang and Zhang [154] have developed a variant that utilizes multiple weight vectors to stabilize the results.

2.2.2 Laplacian Eigenmap

The Laplacian Eigenmap algorithm [10] uses a graph embedding approach. 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. 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 [11].

The neighborhood graph extracts a similarity structure defined by the manifold. We would like the configuration of points in the output space to reflect this structure. In the Laplacian Eigenmap algorithm this is formalized as

$$\frac{1}{2}\sum_{i,j}\|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij} = \mathbf{y}^T\mathbf{L}\mathbf{y}, \tag{11}$$

where $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian and $\mathbf{D}$ is the diagonal matrix with elements $\mathrm{D}_{ii} = \sum_j W_{ij}$. The minimization of this cost function tries to put points that are connected in the graph as close by as possible and does not care what happens to the other points. There are two problems with the cost function, however. First the scaling of the configuration is not fixed. The cost function can be brought to zero by minimizing the scale of the solution. Setting all points to the same position will also bring the cost function to zero. These problems can be solved by adding suitable constraints to the optimization problem. In practice the configuration of points in the low-dimensional space

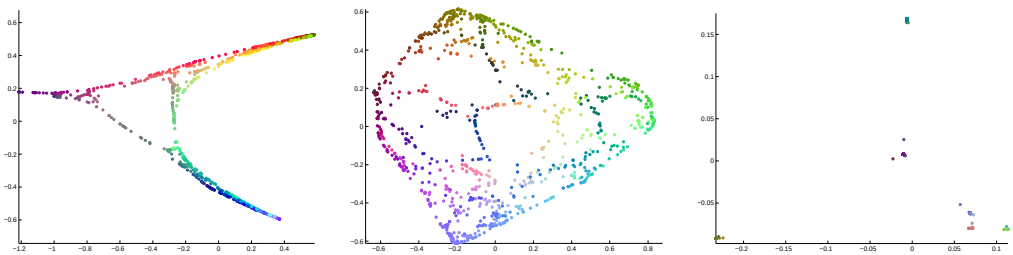
is found by solving the generalized eigenvalue problem [10]

$$\mathbf{L}\mathbf{y} = \lambda\mathbf{D}\mathbf{y}. \tag{12}$$

The smallest eigenvalue always vanishes and corresponds to the solution where all points are at the same location. The configuration of points in the output space is given by the eigenvectors having the next smallest eigenvalues.

The Laplacian Eigenmap algorithm can, in addition to the graph embedding framework described above, be derived based on differential geometry [14]. The eigenfunctions of the Laplace-Beltrami operator form a basis for the manifold. The graph Laplacian can be seen as the graph version of this operator. A third view to the algorithm is provided by interpreting the Laplacian Eigenmap algorithm as an MDS method that tries to preserve the expected commute times on the graph [66]. The commute time between points is defined by the expected time a Markov chain random walker takes when traveling from one point (node) of the neighborhood graph to the other and back. The expectation is taken over all possible paths.

The Laplacian Eigenmap algorithm has a tendency to magnify some distances. This is clearly illustrated in Figure 10. Especially on the clusters dataset the distances between clusters have been magnified in comparison to the distances within clusters. The result is that it is not possible at all to see the structure within clusters from the image. Another effect of the distance magnification is the appearance of "holes" like in the Isomap (Figure 7).



a **b** **c**

Figure 10: Laplacian Eigenmap projections of the three toy data sets. **a**) S-shaped manifold (k=4), **b**) sphere (k=5) and **c**) clusters (k=72).

Another manifold learning method, Hessian Eigenmap [47], is very similar to Laplacian Eigenmap. The algorithm differs from the Laplacian Eigenmap algorithm in that a quadratic form based on the Hessian is substituted in place of the graph Laplacian. This change makes it possible to formulate additional theoretical proofs on the correctness of the method. A linear version of the Laplacian Eigenmap algorithm, Locality Preserving Projections [69], has also been developed.

2.2.3 Other manifold learning methods

Charting methods. Charting or alignment methods are a group of manifold learning methods that work by first defining local linear models and then finding a transformation that aligns the local representations to form a global coordinate system. There are several variants of this theme.

The first method, Global Coordination for Local Linear Methods [127], optimizes both the linear representations and the global coordinates at the same time. This was followed by two methods, Charting [20] and Automatic Alignment of Local Representations [139]. Both first train a mixture model and then find a global coordinate system that combines the local representations. Dividing the problem in two tasks makes the optimization process more tractable.

Verbeek et al. [149] extended the charting method to handle several mixture models in parallel. The mixtures can in this case be either based on the same data set or on different data sets, in which case the method can be seen as a form of non-linear canonical correlation analysis, a classical statistical method for analyzing relationships between two data sets.

Two other, somewhat different, approaches to charting are the Local Tangent Space Alignment algorithm [171] and Locally Smooth Manifold Learning [46]. They do not use mixture models for the local representations.

Manifold learning through semidefinite programming. Manifold learning has also been approached as a semidefinite programming [148] problem. Maximum Variance Unfolding (MVU; also known as Semidefinite Embedding) [160, 161, 162] is very similar to the Isomap algorithm. Instead of forming the distance or kernel matrix using the lengths of the shortest paths on a graph, the distances are optimized. The local distances to neighbors are kept intact and other distances are maximized. This has the effect of unfolding the manifold in a similar way as with the Isomap algorithm. The configuration of points is then found by using linear MDS.

The behavior of the Maximum Variance Unfolding is illustrated in Figure 11. More specifically due to the large computational cost of the algorithm a faster approximate version Landmark Maximum Variance Unfolding (lMVU) [159] was used to generate the images. Even then the cluster data set produced a too large optimization problem for the semidefinite programming solver that was used. The results on the S-shaped manifold and sphere data sets are quite similar to those produced by the Isomap algorithm (see Figure 7) with the exception that lMVU did not produce any "holes" on the sphere data set.

Conformal Eigenmap [132] take a slightly different approach to dimensionality reduction. The manifold is first unrolled using the Laplacian Eigenmap algorithm or LLE to a solution with m-dimensions. The goal is then to transform these points in such a way that the resulting configuration is conformal, that is, the angles between neighbors are preserved as well as possible. This is

achieved by posing the problem as an semidefinite programming problem. Finally the output is produced by using PCA on the transformed data to reduce the dimensionality for visualization.

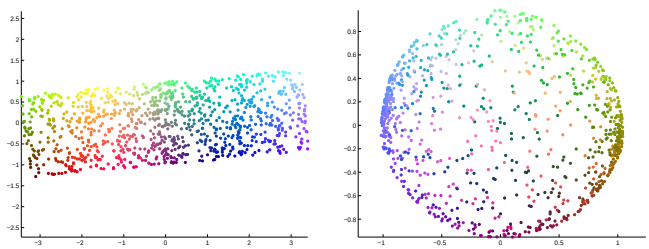
2.3 Other approaches

In addition to the methods described more thoroughly in the previous sections there is a large number of other approaches of dimensionality reduction that often combine aspects from the methods presented above. To give an idea of the breadth of the subject a few recent methods are first listed and then two better known ones are discussed more profoundly.

Tejada et al. [140] proposed a method that projects each data point based on its two closest neighbors among the set of points already projected. Yang's algorithm [163] uses a similar strategy but instead of only considering the two closest neighbors of a data point, his method aims to preserve exactly d, where d is the dimensionality of the output, of the distances from a point to other points already mapped.

Distributional Scaling [123] tries to combine metric MDS with preservation of the distribution of the distances. The Stochastic MDS Network algorithm [107] and Relational Perspective Map algorithm [104] can also be seen as variants of MDS that add additional constraints. In the Stochastic MDS Network algorithm a stochastic gradient update rule is formulated. It tries to present the original distances with similarities constructed in the output space with the constraint that the data points can only be placed on locations defined by a fixed grid. The Relational Perspective Map, on the other hand, uses a force-directed particle system on a surface of a torus to find a distance preserving mapping. For visualization the torus is cut open to create a two-dimensional image.

Instead of trying to preserve pairwise distances as MDS methods do, Globerson et al.[61] have provided a method for visualizing co-occurrence data. In this setting neither the direct pairwise similarities between data points or the



a **b**

Figure 11: lMVU projections of two toy data sets. **a**) S-shaped manifold (k=5) and **b**) sphere (k=4)

similarities between classes or settings are known. Only the probability of a data point being a member of a class or occurring in a setting, is known. The similarity between data points and between classes is only defined through these probabilities. The result from the method is a visualization where both the data points and classes are given locations.

Verbeek et. al. [150] combine the Topology Representing Network of Martinetz and Schulten [106] with Isomap and LLE to form a new projection method. First the Topology Representing Network is used to find a graph representation of the data which is then visualized using Isomap or LLE. In the Isotop algorithm [98, 101] the approach is somewhat reversed. Prototypes are first selected by vector quantization. Then neighborhoods are introduced based on the k-nearest neighbor graph. Finally the projection is generated by adjusting the locations of the prototypes in the output space using a winner take all learning rule.

Finally Lawrence [95] has proposed a Gaussian process model for dimensionality reduction and Meinicke et al. [108] propose that unsupervised kernel regression could be used for finding principal surfaces and for nonlinear dimensionality reduction.

2.3.1 Self-Organizing Map (SOM)

The SOM [87, 88] consists of a regular grid of *units*. The units are typically organized in either a rectangular or hexagonal formation. Each unit contains a model vector $\mathbf{m}_i \in \mathbb{R}^d$, where d is the dimensionality of the data.

The SOM algorithm iterates two steps. First, for data point $\mathbf{x}(t)$ chosen randomly at iteration step $t = 0, 1, 2, \ldots$, the best matching model vector $\mathbf{m}_{c(t)}$ is sought using the equation

$$c(t) = \underset{j}{\operatorname{argmin}}\{d(\mathbf{x}(t), \mathbf{m}_j)\}, \tag{13}$$

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

$$\mathbf{m}_j(t+1) = \mathbf{m}_j(t) + h_{c(t),j}(t)[\mathbf{x}(t) - \mathbf{m}_j(t)]. \tag{14}$$

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, is the neighborhood function. Two forms of the neighborhood function are commonly used. The *bubble* neighborhood is defined as a step function. All the units within the radius of the neighborhood are adapted equally and no adaptation is done on units that lie outside the neighborhood. The other common type of neighborhood function is a Gaussian. Each unit in the map is updated on each step, but the amount of update depends on their distance from the winning unit on the SOM grid. The neighborhood function makes the map ordered.

The visualizations produced by the SOM differ from the other methods presented in this section, in that instead of each data point having its own location, each data point is placed at the location of the best matching unit on the fixed SOM grid. If labeled data exist the units can be labeled according to the data for which the unit is the best matching unit.

Typically SOMs are visualized using the U-matrix [146] or its variants. A gray-shade code is used to display distances between neighboring units. On such a display, the light areas contain units that are mutually more similar than on the dark areas (Figure 12). Technically, space is added in between each pair of SOM units and shaded according to the distance between their model vectors. The shade at the locations of the map units is usually set proportional to the median or average of distances to its neighboring units.

Assessing the similarity (dissimilarity) of data points on the SOM display is somewhat more complicated than on a scatter plot display. There are two approaches. First the similarity can be defined simply as the distance on the display plane like with scatter plots. This measure does not, however, take into account the density of the model vectors that is visualized by the U-matrix. The second way to define similarity is slightly more complex. The SOM can be thought to form a graph where each node is a vertex and the neighboring units are connected with edges. Each edge has a weight that is equal to the distance between the nodes in the input space. The similarity is then defined to be the shortest path distance between two nodes. In the U-matrix, on light areas, such distances are shorter and on dark areas they are longer. This way of assessing similarity is more in tune with the way an experienced analyst utilizes the SOM.

There are two main learning algorithms for the SOM: a batch algorithm and the sequential algorithm, which was described above. In the batch algorithm the updates are done for the whole data set at the same time instead of one data point at a time. In addition to the two main algorithms there is a huge number of different variants of the SOM with different cost functions, update rules and topologies [115].

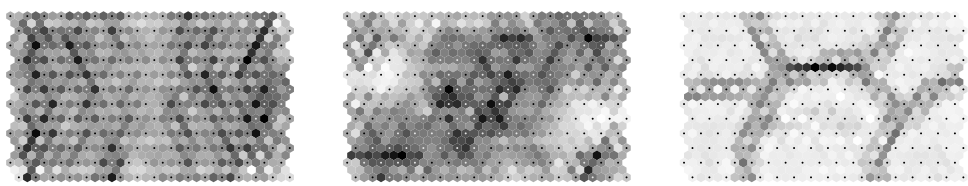


Figure 12: U-matrix visualizations of the three toy data sets. **a**) S-shaped manifold, **b**) sphere and **c**) clusters.

2.3.2 Stochastic Neighbor Embedding (SNE)

Although SNE is quite similar to several MDS methods there is one profound difference. The SNE algorithm [73] 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 such that those probabilities are the same 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

$$p_{ij} = \frac{\exp\left(-d(\mathbf{x}_i, \mathbf{x}_j)/\sigma_i^{(i)}\right)}{\sum_{k \neq i} \exp\left(-d(\mathbf{x}_i, \mathbf{x}_k)/\sigma_i^{(i)}\right)}, \tag{15}$$

where $d(\mathbf{x}_i, \mathbf{x}_j)$ is the squared pairwise Euclidean distance between the data points in the input space. The parameter $\sigma_i^{(i)}$ that controls the width of the Gaussian is either set manually or by fixing the entropy of the distribution. Setting the entropy equal to $\log k$ sets the "effective number or neighbors" to k.

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

$$q_{ij} = \frac{\exp\left(-\|\mathbf{y}_i - \mathbf{y}_j\|^2/\sigma_i^{(o)}\right)}{\sum_{k \neq i} \exp\left(-\|\mathbf{y}_i - \mathbf{y}_k\|^2/\sigma_i^{(o)}\right)}. \tag{16}$$

In the original SNE formulation the parameter $\sigma_i^{(o)}$ is always set equal to one for each data point. The configuration of points $\mathbf{y}_i$ that minimizes the Kullback-Leibler divergence [92] between the probability distributions in the input and output spaces is the solution for the problem. The cost function is thus

$$E = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{17}$$

Selecting a constant width ($\sigma_i^{(o)} = 1$), as was originally proposed, for the Gaussian in the output space causes SNE to perform a probability preserving mapping, but not a distance preserving mapping. SNE can also be made distance preserving if we set $\sigma_i^{(o)} = \sigma_i^{(i)}$. Whether this is desirable depends on the application. If the input probabilities are thought to form a good model of the structure of the data and the output distribution is a good model for the observer then using a constant $\sigma_i^{(o)} = 1$ in the output space is a good choice. On the other hand if SNE is just used as a dimensionality reduction method for visualization then it is probably better to use the same σ_i in both spaces. This removes one aspect that can make the interpretation of the results

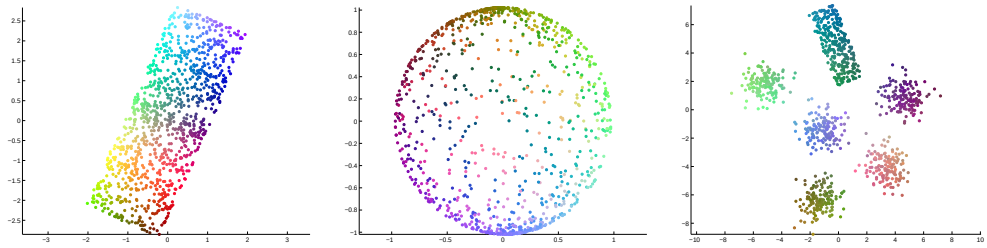


Figure 13: SNE projections of the three toy data sets. **a**) S-shaped manifold, **b**) sphere and **c**) clusters.

harder. The performance of SNE is illustrated in Figure 13. The figures were computed using a constant $\sigma_i = 1$ in both spaces.

SNE has been extended in several ways. A mixture version was introduced in the original paper. It replaces the Gaussian probability distribution in the output space with a mixture of Gaussians that allows the data point to have multiple locations. Multiple Relational Embedding (MRE) [109] extends SNE by allowing multiple similarity matrices to be used in the input space. Iwata et al. [77] extend SNE for visualizing the data together with class labels. The method can also very easily be modified for visualizing co-occurrence data. SNE can also be seen as a special case of Bernoulli Relational Embedding algorithm [153] or Linear Relational Embedding [117]. Also a semisupervised variant of SNE, Large Margin Non-linear Embedding [173] was recently introduced.

2.4 Discussion

Many of the dimensionality reduction methods rely on a graph representation of the data manifold to describe what is local or interesting in the data. This leads to the question of what is the best way to define the graph. Typically the methods use k-nearest neighbor graphs in which the data point is connected with an edge to its k nearest neighbors. The problem in this case is how to select a suitable k. A too small k will create a graph that is not connected, which causes problems. On the other hand, a large k will lessen the locality of the features and might prevent the unfolding of the data manifold. For example an edge might be added that "short-circuits" two different parts of a nonlinear manifold. There is still very little research on how to define a good graph for these methods. Lee and Verleysen [102] have proposed a method that can be used to break loops in data manifolds such as spheres or toruses. Yang [165, 166] and Carreira-Perpiñán and Zemel [29] have introduced relatively similar graph construction methods that utilize minimum spanning trees to create connected neighborhood graphs. It is still unclear, however, how well these methods work in practice.

Using graphs to define the data manifold also suggests that the large number of different graph layout or drawing methods could be utilized for dimensionality reduction. In fact, the Laplacian Eigenmap algorithm is directly derived from graph theory. Many of the force directed methods, such as the algorithm from Fruchterman and Reingold [53] and the LGL algorithm [1], that are aimed for finding layouts for general undirected graphs, could also be suitable for dimensionality reduction. On the other hand, many dimensionality reduction methods can be used for finding layouts for graphs. In fact, MDS methods have been used for graph drawing for a long time [36, 55, 82].

Another related field is intrinsic dimensionality estimation. Knowing the intrinsic dimensionality of the data does not help in a visualization context, because the visualization constrains the output dimensionality to two or at most three, but it can be necessary in other tasks that dimensionality reduction is used for. There have been several different approaches to intrinsic dimensionality estimation. For example methods have been based on local PCA [54], neural networks [26, 45], inversion error [105], vector quantization [124], geodesic entropic graphs [37], kernels [70], maximum likelihood [103], fractals [27], and packing numbers [85].

3 VISUALIZATION: TO TRUST OR NOT TO TRUST

One of the longstanding problems in the field of information visualization is how to assess the quality of produced visualizations or the tools. The main difficulty is that humans are a part of the visualization loop. This leads to usability studies as the only way of assessing the final quality. Although work has been done in trying to define simple, more basic visualization tasks and criteria (see for example [4, 51]) it is still difficult to define the exact test setting to be used in assessing a new method. In addition, usability studies are relatively expensive and time consuming. Typically they are not available to developers of new methods, at least at the initial stages of development. Because of the difficulties with usability studies the need for quality metrics has been brought up [32, 110].

The goal of quality metrics is not to assess how well humans can extract information from the visualization but to quantify how well the visualization represents the underlying data, without taking the human component into account. Accurate data representation and good readability of the visualization are necessary conditions for a usable visualization. If they are not met there is no need to assess the full visualization process in more detail.

3.1 Measuring the quality of visualizations

One of the first persons to propose metrics to evaluate the quality of visualizations was Edward Tufte [145]. He proposed measures like the *data-ink ratio*, which is the ratio of ink used to draw the actual data to the ink used in the image overall, the *lie factor*, the ratio of the perceived value in the image to the real value in the data and *data density* which relates the amount of data to the size of the image. Although the measures proposed by Tufte are relatively general, and not that exactly defined, two things can be noted. First, there are two categories of quality: technical quality and representation quality. Data-ink ratio and data density measure the technical quality of the image, in the sense that they are diminished by the axis labels, tick marks and other such items that are not directly related to the data. The lie factor, on the other hand, tries to quantify how truly the data is represented in the visualization. Besides, it is clear that the quality of the visualization has several different aspects and no single measure can quantify them all.

After Tufte there has been some work done to create new quality measures. Brath [21] has extended Tufte's concepts to 3D visualizations by proposing measures like the mapping score and occlusion percentage. They measure the cognitive difficulty of the mapping of dimensions to graphical features and the ratio of data points completely covered by other points, respectively. Bertini and Santucci [18] proposed a way to quantify data density in a way that the number of data points in the visualization can be reduced without affecting the overall structure seen in the data. Yang-Peláez and Flowers [167] propose to

measure the effectiveness of a display by comparing the information content of the data visualization and the information capacity of the display. If they are equal then the visualization is effective. These measures are mostly concerned with the technical quality of the visualization.

Evaluating the representation quality of the visualization is typically very task-specific. Different aspects of the data need to be preserved for different tasks. For example when the task is to assess the distribution of values in certain dimensions of a high-dimensional data set the values in that variable should not be distorted but there is no restriction on the values in the other dimensions. On the other hand if the goal is to represent the pairwise similarities between data points then the visualization will typically not even display single dimensions and it would be meaningless to try to assess how well the values are preserved. In the following we will only concentrate on quality measures that are aimed at studying how well the pairwise similarities in the data are preserved in the visualization process.

3.1.1 Use of visualization quality measures in the dimensionality reduction literature

Table 1 gives a review of the methods that have been used to assess the quality of dimensionality reduction methods in the literature. It contains 69 papers from the years 2000–2006 on dimensionality reduction methods that can be used for visualization. The papers were collected by the author in the course of the research work. Papers on the Self-Organizing Map were not included in the review. There is a very large amount of papers on Self-Organizing Maps [115] and they typically differ from the other dimensionality reduction methods in how the quality of the mapping is assessed.

The most notable finding is that 28 ($\approx 40\%$) of the papers only presented visualizations of toy or real data sets as a proof of quality. Most of the more quantitative quality measures or approaches were based on two principles. The first group of methods compares all pairwise distances or the order of all pairwise distances. This group contains the MDS type cost functions like Sammon's cost and stress, graphical methods that relate the distances in the input space to the output space, and various correlation measures, including rank based correlation measures, that asses preservation of all pairwise distances. The other common quality assurance strategy is to classify the data in the low dimensional space and report the classification performance. This approach was typically used with supervised methods, but also with a few unsupervised methods.

The disadvantage of the measures that are based on all pairwise distances is that they are not directly related to any specific visualization task, unless one is doing distance estimation. Classification, on the other hand, is directly related to the task where the user is trying to find specific groupings in the data. The disadvantage of this approach is that it can be quite insensitive

Table 1: Methods used in publications to assess the quality of dimensionality reduction methods. **Vis. real data**: Example visualization of a real world data set or a data set that simulated a real world data set. **Vis. toy data**: Example visualization using a simple toy data set. **Classification**: Some classification accuracy measure. **MDS cost**: Stress, Sammon's cost function or other measures of pairwise distance preservation. **Graphical distance comp.**: Any graphical method that compares original and projected distances. **Cost function**: The cost function of the method itself. **Correlation of dist.**: Some correlation measure used between input and output distances. **Trustworthiness**: The trustworthiness measure (Publ. 2). **Other**: Computational speed, approximation accuracy, distribution of eigenvalues, etc.

Quality assessment method	Number of Publications	References
Vis. real data	50	[3, 10, 12, 20, 39, 40, 42, 43, 59, 61, 62, 69, 72, 73, 76, 77, 80, 89, 93, 94, 95, 98, 100, 101, 102, 104, 108, 109, 112, 113, 114, 123, 126, 127, 130, 132, 138, 139, 140, 149, 152, 153, 154, 159, 160, 161, 162, 163, 164, 171]
Vis. toy data	43	[3, 10, 12, 20, 34, 42, 46, 47, 48, 59, 62, 66, 69, 89, 93, 94, 98, 99, 100, 101, 102, 104, 107, 108, 109, 123, 130, 132, 139, 140, 149, 150, 152, 153, 154, 159, 160, 161, 162, 164, 165, 166, 169, 171]
Classification	16	[13, 14, 39, 40, 61, 62, 68, 69, 77, 93, 108, 130, 144, 152, 159, 162]
MDS cost	9	[3, 29, 76, 113, 114, 121, 123, 163, 164]
Graphical distance comp.	7	[38, 66, 107, 114, 149, 155, 171]
Cost function	5	[107, 112, 140, 160, 164]
Correlation of dist.	3	[29, 165, 166]
Trustworthiness	3	[48, 72, 138]
Other	14	[38, 46, 72, 93, 94, 98, 107, 108, 123, 132, 159, 160, 161, 162]

if the classes are large. Classification accuracy is not affected by anything that happens inside the classes. The other disadvantage of the classification approach is that it requires labeled samples that are not always available.

A group of quality measures that are used in the Self-Organizing Map literature but not typically with other dimensionality reduction approaches are the topology preservation measures. Examples of this group are the Topological Product [8], Topological Function [151], and Topographic Error [86].

Another aspect of assessing the quality of the visualizations is that typically the measures are not meaningful unless there is some kind of a reference that they can be compared to. For example a 100% correct classification might not mean that the method is good if it is known that the data set is very easy to classify. The easiest way to provide a reference is to visualize the data set with several methods. This will give an idea of what kind of performance one can expect on the data set. If the method is aimed for more general use then it should also be tested on several data sets to see that the results are stable. Table 2 provides a summary of the number of methods compared and the number of data sets used in the reviewed papers. Typically only one other method was used to provide a reference to the performance of the new method. Most commonly the reference method was PCA. The papers that had a very large number of data sets or methods utilized the classification strategy. The performance of the visualization method (and a classifier on top of it) was compared with a large set of other classifiers.

3.1.2 Need for new quality measures based on simple visualization tasks

It is quite surprising that a large number of the papers studied in the survey did not use any quantitative measures of quality to verify the results. One reason for this can be the lack of suitable quality measures. Of the existing quality measures classification accuracy is the only one that is directly related to a visualization task, the task of finding specific groupings in the data. Unfortunately classification accuracy can only be used on the small subset of data sets that contain labeled samples which limits its usability. It is clear that new quality measures need to be devised and that the measures need to be related to a specific visualization task.

In the next section a new visualization task is formulated and then quality measures are designed for it.

3.2 Visualizing similarities: the information retrieval point of view

Although visualization algorithms and applications have often been motivated using an information retrieval task the connection has been vague. A more specific connection can be developed by studying how an analyst explores the

Table 2: Number of methods compared and data sets used in publications. If the publication introduced a new method only methods it was compared to were counted.

Methods Compared	Number of Publications	References
0	8	[3, 43, 46, 99, 127, 139, 149, 160]
1	28	[10, 13, 12, 14, 34, 59, 73, 76, 93, 94, 98, 100, 101, 109, 112, 113, 123, 126, 130, 140, 144, 150, 153, 155, 165, 166, 169, 171]
2	13	[20, 38, 40, 47, 48, 68, 78, 80, 95, 138, 161, 163, 164]
3	13	[39, 42, 62, 66, 69, 72, 89, 104, 108, 114, 132, 154, 159]
4	6	[61, 77, 102, 107, 121, 162]
≥ 10	1	[152]
Total	69	

Data Sets Used	Number of Publications	References
1	13	[34, 43, 47, 66, 68, 72, 78, 107, 113, 126, 150, 165, 169]
2	14	[13, 38, 48, 59, 73, 93, 94, 109, 112, 121, 123, 127, 138, 160, 166]
3	16	[10, 14, 61, 76, 77, 80, 95, 99, 101, 102, 108, 114, 132, 149, 162, 163]
4	7	[12, 20, 69, 104, 139, 153, 154]
5	6	[3, 42, 98, 155, 159, 164]
6	4	[46, 62, 100, 161]
7	3	[89, 140, 144]
8	3	[130, 152]
≥ 10	3	[39, 40, 171]
Total	69	

pairwise similarities in a data set. For example the data could consist of a large set of indicators of welfare for all the countries in the world. The analyst then selects a country, say Finland, and tries to find out which other countries have a similar welfare profile. He then picks a few other similar countries for more profound study to see what makes the countries similar and what makes them different. Another example is the case of reading an article and trying to learn more of the subject by studying other similar papers. The third example comes from bioinformatics. The functional class of several genes may be known but many genes are still totally unknown. By studying the behavior of genes in various situations a hypothesis can be formed on the functionality of an unknown gene based on the known functions of other genes that behave similarly. In all cases the visualization system can provide a visual interface, that allows neighbors, similar objects, to be selected from the display. There are two common themes in all three examples. First, the user selects one object and then wants to find or retrieve other similar objects from the data set. Secondly, the user is typically only interested in the small set of the most similar items.

In a visualization context the above behavior can be used to define a new specific task, *visual neighbor retrieval*, where the goal of the user is to find neighbors of a few interesting data points. The k neighbors of a data point are the set of k data points that lie closest to it in the input space. The task of a visualization algorithm is then to provide a single display that allows the user to find the neighbors of a data point as faithfully as possible, without prior knowledge of which data points the user is going to select.

Although an information retrieval system could easily be used to present the neighbors as a list, a visual interface to the data provides several advantages. The interface remains constant. Instead of getting a different list on every query the display remains the same. It becomes possible to remember where the objects are located on the display. A visualization gives additional information of the data that is not easily available from the query engine. By looking at the visualization the user can possibly answer such questions as: Are there unknown data points that do not seem to be related to the known ones? Which groups are most closely related to each others? Are there denser areas and more sparse ones? Does the data form clusters? Questions like these are very hard to answer by doing specific queries on the neighbors of single data points. A good visualization can help in answering them by giving an overview of the data at the same time as providing the means to find the neighbors of specific data points. The disadvantage is that the visualization cannot specialize each result to a specific query in a similar way as an information retrieval system can. An information retrieval system can produce the correct list of neighbors every time in this case, but in general it is not possible to represent all neighbor relationships correctly in the visualization. It is, however, possible to augment an interactive visualization system by providing the results of an information retrieval query in addition to the visual display.

In information retrieval the performance of the system is typically assessed by two measures, *precision* and *recall*. Precision is defined as

$$\text{precision} = \frac{N_{TP}}{k} = 1 - \frac{N_{FP}}{k}, \tag{18}$$

where N_{TP} is the number of the true positives, that is the number of correctly retrieved relevant items, N_{FP} is the number of the false positives and k is the number of the retrieved items. Recall is defined as

$$\text{recall} = \frac{N_{TP}}{r} = 1 - \frac{N_{MISS}}{r}, \tag{19}$$

where N_{MISS} is the number of the misses, the relevant objects not retrieved, and r is the total number of the relevant objects in the data base.

These measures can easily be adapted to visualization. The number of neighbors r in the input space is the set of relevant items. The set of the retrieved items is the number of neighbors k that are looked at in the visualization. Finally, for the full visualization the measures calculated for the neighborhood of each data point need to be combined, in effect defining average precision and recall of the visualization.

The problem with the average precision and recall measures in a visualization context is that they do not utilize very well the information available in the data. The measures rely only on a binary relation but typically a lot more information is available of the similarities of the data points. This allows more informative measures to be developed.

3.3 Measures of trustworthiness and continuity

When the dimensionality of the data is reduced it is not in general possible to preserve all similarities in the data. The reduction of dimensionality causes two kinds of errors. First, data points that are not neighbors in the input space can be mapped close by in the output space causing data points to be falsely identified as neighbors. These are the errors that decrease the precision of the visualization. Secondly, data points that are originally close by are mapped far away in the visualization process. These kinds of errors are caused by discontinuities in the mapping and result in some of the neighbor relations not being present in the visualization. These errors reduce recall. In the precision and recall measures these errors are quantified based on counts of how many data points cause errors. This means that each error is equally bad. An equal error is not intuitive in a visualization context where we usually know the distance between data points in the input space. Intuitively, a data point that comes into the neighborhood of another one from far away causes a larger error than the one that comes from closer by. By ranking the data points based on their similarity we can devise two new quality measures, *trustworthiness* and *continuity*, that quantify the errors by the ranks of the erroneous data points.

More formally the trustworthiness of a visualization is defined as follows. 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. The measure of trustworthiness $M_{trust}(k)$ of the visualization is

$$M_{trust}(k) = 1 - A(k) \sum_{i=1}^{N} \sum_{j \in U_k(i)} (r(i,j) - k), \tag{20}$$

where the term $A(k)$ scales the measure to be between zero and one. The error gets its maximum value when the ranks in the input and output space are reversed. The scaling term can then be found by considering that the maximum error in each data points neighborhood is the sum of the k last ranks (minus the neighborhood size k). Thus the scaling term becomes

$$A(k) = \begin{cases} \frac{2}{Nk(2N-3k-1)}, \text{ if } k < \frac{N}{2} \\ \frac{2}{N(N-k)(N-k-1)}, \text{ if } k \geq \frac{N}{2} \end{cases}, \tag{21}$$

The trustworthiness measure is very closely related to the precision measure (18). The neighborhood size k in the trustworthiness measure defines the number of items retrieved. In the precision measure the error made in the retrieval is quantified by the number of erroneously retrieved items and then scaled (with the number of retrieved items in this case) to lie between zero and one. In the trustworthiness the error is quantified by the shifted ranks of the erroneously retrieved data points. In effect we can say that the trustworthiness measure is a kind of precision measure for the case where the objects are ranked based on their relevance.

The trustworthiness measure could also be based on the distances instead of ranks. Such a variant of the measure was tested (unpublished results) and the behavior was very similar to the rank-based one. Differences could arise in cases where the scales of distances vary a lot in the data space. The choice to base the measure on ranks was made to allow the use of any ordinal dissimilarities (or similarities) as the base for the measure instead of only metric ones and to make it possible to scale the measure between one and zero. This makes interpreting the resulting values easier. Additionally, using ranks instead of distances makes the measure more robust to outliers.

The errors caused by discontinuities can 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

$$M_{cont}(k) = 1 - A(k) \sum_{i=1}^{N} \sum_{j \in V_k(i)} (\hat{r}(i,j) - k) \; . \tag{22}$$

The scaling factor $A(k)$ is the same as in (20). The continuity measure is similarly related to the recall measure as trustworthiness is to precision. Now both the number of relevant items and the number of retrieved items is defined by the number of neighbors k, and instead of quantifying errors purely based on the number of misses, rankings are used. Thus a point that is projected far away from the neighborhood in the visualization process will cause a larger error than a point that was projected just out of the neighborhood. In recall both errors are considered equally severe.

A small technical detail should be noted. In both measures if there are ties in rank ordering, caused by equal distances, all compatible rank orders are assumed equally likely. In practice the best and worst case values of the error measures are calculated and the average of them is reported. Equal distances do not typically occur in scatter plot visualizations, but are common in cases where visualizations are combined with clustering. For example Self-Organizing Maps and hierarchical clustering methods [79] produce visualizations where equal distances are common. An additional point that should be noted is that although the measures are normalized to lie between zero and one, it is unlikely to have a trustworthiness or continuity value of zero in any practical situation. In practice, an estimate of the lower bound of usable quality can be found by studying the trustworthiness and continuity values of purely random mappings. Typically a random mapping to two dimensions will have both a trustworthiness and continuity value of approximately 0.5. The actual value depends somewhat on the neighborhood size k (see Figure 14).

The trustworthiness and continuity measures can also be motivated from the point of view of topology preservation, as was done in Publication 2. In this framework the continuity measure is related to the other topology preservation measures like topological product [8], topological function [151] and topographic error [86]. They all measure the discontinuity of the mapping in some way. The trustworthiness measure, on the other hand, does not have any counterparts in this field.

There are two main applications for the trustworthiness and continuity measures. First, they can be used to compare the performance of different methods on a data set or assess the quality of a single method. Secondly, the measures can be used to improve visualizations.

Comparing different dimensionality reduction methods. The main application of the trustworthiness and continuity measures is to assess the quality of visualizations. Based on the quality assessment the best run, method

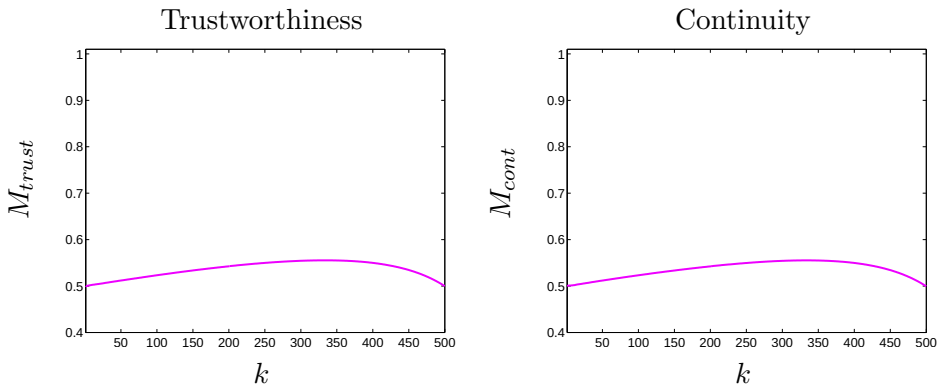


Figure 14: Average trustworthiness and continuity of 100 random mappings of the S-shaped manifold data set as a function of the neighborhood size k.

or parameters for the data set can be selected. A good way to compare methods is to study how they behave on a large range of neighborhood sizes. Although small neighborhoods, relating to the data points most likely to be thought of as relevant, are the most important ones, this gives an overview of performance for a wide range of situations. The comparison is made by plotting the trustworthiness and continuity measures of each method as a function of the neighborhood size k. This is illustrated in Figure 15 on two real world gene expression data sets (For a description of the data sets see Publications 3 and 4). A variation of this theme is to combine the two plots so that the axes are trustworthiness and continuity. It should be noted that this is somewhat different from the way precision–recall curves are used in information retrieval. The size of the neighborhood k does not directly affect the tradeoff between trustworthiness and continuity like in the precision–recall curves. This visualization method is illustrated in Figure 16. In this display when a single value of k is selected each method is represented by a single point instead of a curve. Especially, if the number of the methods is large this makes the visualization clearer.

Interpreting the curves in Figure 15 is easy. On both data sets SOM is the best overall method being the best or second best in terms of trustworthiness on all neighborhoods sizes while still managing to retain a good continuity. CCA has good trustworthiness on small neighborhoods but the disadvantage is the comparatively low continuity. SNE is the best method in terms of continuity on both data sets and it is also among the three best methods in terms of trustworthiness. LLE seems to perform poorly on these data sets. It is among the worst methods on both data sets in terms of both measures. From Figure 16 we can see that PCA seems to have an almost constant trustworthiness on all neighborhood sizes but the continuity decreases as the neighborhood size grows. CCA, on the other hand, decreases only slightly in continuity while the trustworthiness decreases fast as k increases.

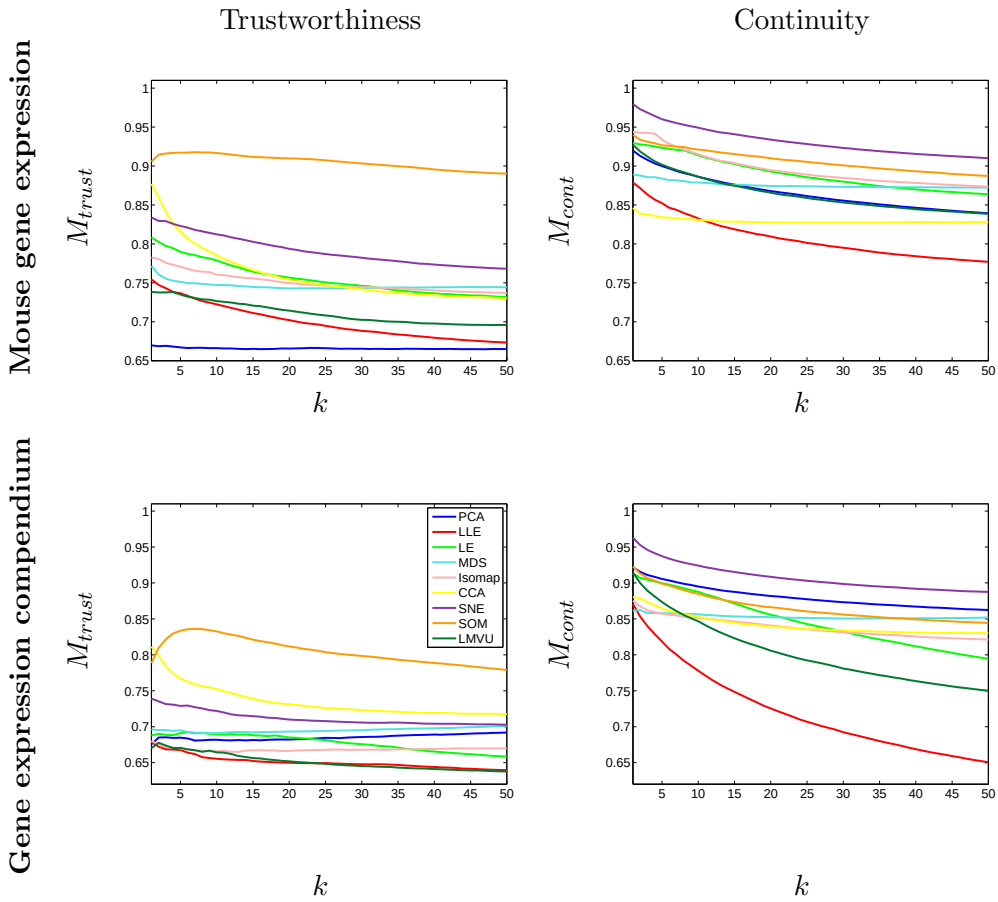


Figure 15: Trustworthiness and continuity of the mapping on the Mouse gene expression (Top) and Gene expression compendium (Bottom) data sets as a function of k, the size of the neighborhood used in measuring them. The trustworthiness and continuity values of a random mapping are approximately 0.5. PCA: Principal Component Analysis, LLE: Locally Linear Embedding, LE: Laplacian Eigenmap, MDS: Metric Multidimensional Scaling, CCA: Curvilinear Component Analysis, SOM: Self-Organizing Map, LMVU: Landmark Maximum Variance Unfolding and SNE: Stochastic Neighbor Embedding.

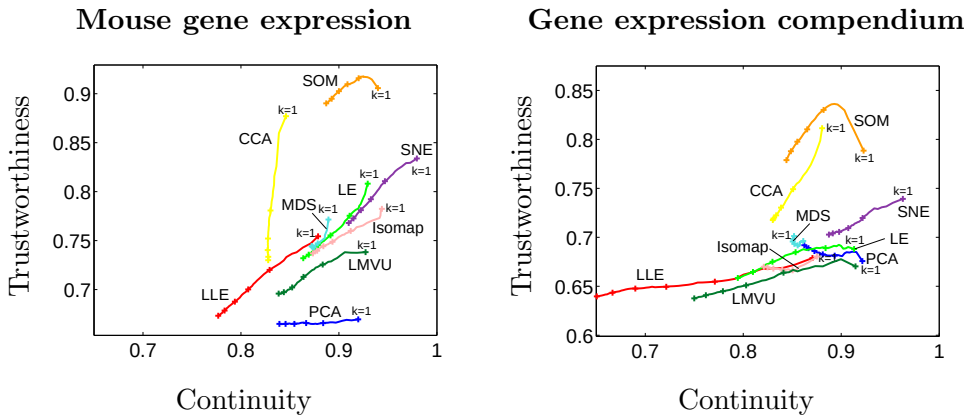


Figure 16: Trustworthiness - Continuity curves of several methods on the Mouse gene expression (Left) and Gene expression compendium (Right) data sets. The number of neighbors k used to calculate the measures is varied from 1 to 50. The markers on the curves are set at 10 point intervals. The best performance is at the upper right corner of the plot. For key see Figure 15.

MDS seems to have an almost constant performance on different neighborhood sizes. More in-depth examples on how the measures can be applied to compare or study different methods can be found in Publications 2, 3, 4, 5 and 7.

Improving visualizations using trustworthiness. There are two ways how the trustworthiness measure can be used to improve a visualization. The first method, used in Publication 3, is to recursively remove data points from the visualization based on how much they reduce the overall trustworthiness. By removing those points that are badly visualized the overall visualization can be made more trustworthy. There are several approaches to selecting how many data points should be removed. A desired trustworthiness value can be set and data points removed until it is met. A threshold can be set to the maximum total error that the data point is allowed to induce and data points are removed until the remaining ones do not cause a large enough error. Or simply a fixed number of points is removed. Alternatively, badly visualized points can be marked according to how much they contribute to the overall trustworthiness error.

The second method is to mark single data points or areas in the visualization according to the trustworthiness of the neighborhood of the data point. In [116] units on the SOM were color coded according to the trustworthiness of the unit. On a scatterplot-like display the individual data points or the Voronoi cells around them, as Aupetit [7] suggests, can be color coded according to the trustworthiness of the neighborhood of the data point.

The continuity measure has not thus far been used in enhancing visualizations, but it could be possible to develop a method where the pairs of points that contribute the most to the continuity error are colored in a similar way or

connected with lines. This would show where the mapping is causing discontinuities. If an interactive display can be used it would be easy to mark the data points that are either too near or too far away from the selected point. The intensity of the marker could depend on the magnitude of the error.

3.4 Precision and recall of stochastic neighbors

In the precision and recall measures the relevancy of the object is defined as a binary variable. Each object either is or is not relevant. In the trustworthiness and continuity measures the relevancy is defined on an ordinal scale. Objects can also be assigned a probability of being relevant. This is in fact done in the Stochastic Neighbor Embedding method (see Section 2.3.2), where each data point is assigned a probability of being a neighbor of another both in the input and output space. The Kullback-Leibler divergence is then used to find a mapping that tries to preserve the probabilities of being a neighbor as well as being possible.

A new interesting connection between information retrieval and SNE can be found if we replace the Gaussian neighbor distribution used in SNE with one where the l closest points in the input space are neighbors with a high probability and the rest with a very low probability. Technically, this is done by defining the probability of point j being a neighbor of point i in the input space, p_{ij}, as a step function

$$p_{ij} = \begin{cases} a \equiv \frac{1-\delta}{r}, & \text{if point } j \text{ is among the } r \text{ nearest} \\ & \text{neighbors of } i \text{ in the input space} \\ b \equiv \frac{\delta}{N-r-1}, & \text{otherwise} \end{cases}, \tag{23}$$

where N is the total number of data points, r is the size of the desired neighborhood and $0 < \delta < 0.5$ gives the non-neighbors a very small probability.

Similarly, we define the probability of j being a neighbor of i in the output space by

$$q_{ij} = \begin{cases} c \equiv \frac{1-\delta}{k}, & \text{if point } j \text{ is among the } k \text{ nearest} \\ & \text{neighbors of } i \text{ in the visualization} \\ d \equiv \frac{\delta}{N-k-1}, & \text{otherwise} \end{cases} \tag{24}$$

where k is the neighborhood size in the output space.

It is straightforward to check that if δ is very small, then terms concerning the misses dominate the SNE cost function (for details see Publication 7) and, moreover,

$$D(p_i, q_i) \approx N_{MISS} \frac{1-\delta}{r} \log \frac{(1-\delta)}{\delta} = \frac{N_{MISS}}{r} C_1, \tag{25}$$

where C_1 is a constant and N_{MISS} is the number of misses. Thus SNE (17) would try to maximize (average) recall (19).

By reversing the direction of the Kullback-Leibler divergence we get the following result:

$$D(q_i, p_i) \approx \frac{N_{FP}}{k} C_2, \tag{26}$$

where N_{FP} is the number of false positives and k is the number of retrieved points. Minimizing this would correspond to maximizing precision (18).

By using the Gaussian neighbor distributions in the SNE cost function, it can be used as a quality measure that focuses on a kind of expected smoothed recall. Similarly by reversing the direction of the Kullback-Leibler divergence in the cost function we get a measure that assesses a kind of expected smoothed precision. It should also be noted that the neighbor distribution in the input space needs not be a Gaussian distribution based on the pairwise distances. Instead, the probabilities could arise from any probabilistic similarity model.

3.5 Discussion

One of the most important questions in the field of information visualization is how to assess the quality of the produced visualizations. The problem is that the final arbiter of value is the person who uses the visualization to analyze data. To find out whether the visualization method is a good one for a specific task, carefully controlled usability studies are required. Although this is the case, a lot can be achieved by defining measures that can verify that the visualization represents the data accurately. Accurate representation of the data is a necessary, but not sufficient, condition for a good quality visualization.

Often the quality measures used in the literature are not tuned to any specific visualization task. The most notable exceptions are the classification accuracy, which is connected to finding groups of similar items from the data and the new measures introduced in this work which are connected to a new task, visual neighbor retrieval.

For this new task three pairs of new measures were proposed. The traditional measures of information retrieval, namely precision and recall, have been extensively used to assess the quality of information retrieval systems, but are new in the context of information visualization. Although new, the trustworthiness and continuity measures have been successfully used in conjuction with several different dimensionality reduction methods. In addition to dimensionality reduction they have also been applied to other types of visualization methods like hierarchical clustering (Publication 3) and graph layouts (Publication 6). In the third pair of measures the definition of similarity (relevance) is based on a probabilistic model instead of a binary or ordinal relationship. This leads to a pair of new quality measures that focus on a form of smoothed recall and precision. In all three cases there is a tradeoff between the pair of measures. Aiming for a high trustworthiness/precision will typically lead to a lower continuity/recall and vice versa.

Although the trustworthiness and continuity measures were defined by using the k-nearest neighbors it is not the only possible way of defining the neighborhood structure. In Publication 6 the trustworthiness and continuity measures are extended to layouts of undirected unweighted graphs by considering all directly connected nodes as neighbors, regardless of their number.

It should be noted that while the trustworthiness and continuity measures have shown great promise in practice, they are not sufficient to guarantee that the visualization is good. For example, consider a data set that consists of clusters of data points. If the visualization process magnifies the distances between clusters in comparison to distances within clusters it can happen that the resulting visualization only shows the clusters as small clumps and individual points cannot be seen at all. Still the trustworthiness of the visualization can be high if the order of distances is well preserved. This illustrates that no single measure is good enough to guarantee that the visualization is of good quality. At least a qualitative visual check should be made to complement the numerical measures. On the other hand, it is usually very hard to verify the quality of a visualization by just looking at it, so this is not a magic bullet either.

4 METHODS FOR VISUAL NEIGHBOR RETRIEVAL

In Section 2 a large number of methods was described. Each has its own notion of what should be preserved in the dimensionality reduction process but none of them is aimed for the visual neighbor retrieval task introduced in the previous section.

4.1 Neighbor Retrieval Visualizer (NeRV)

In section 3.4 a pair of measures, for smoothed precision and recall, were introduced. It was also noted that the cost function of a dimensionality reduction method SNE (see Section 2.3.2) optimizes smoothed recall. As it is well known in the information retrieval field, optimizing recall will typically lead to a low precision and vice versa. Thus SNE focuses on only one aspect of the visual neighbor retrieval task.

In Publication 7 a new nonlinear dimensionality reduction method, called *Neighbor Retrieval Visualizer (NeRV)*, is introduced. The idea is to add a term to the SNE cost function that matches the smoothed precision measure (26) thus bringing in the other aspect of quality. The cost function of the NeRV algorithm is

$$\begin{aligned} E_{\mathrm{NeRV}} &= \lambda E_i[D_{KL}(p_i, q_i)] + (1-\lambda) E_i[D_{KL}(q_i, p_i)] \\ &= \lambda \sum_i \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}} + (1-\lambda) \sum_i \sum_{j \neq i} q_{ij} \log \frac{q_{ij}}{p_{ij}}, \end{aligned} \quad (27)$$

where $\lambda \in [0 \ldots 1]$, is a parameter that selects the tradeoff between (smoothed) precision and recall, p_{ij} is the probability of point j being a neighbor of point i in the input space and

$$q_{ij} = \frac{\exp\left(-\frac{\|\mathbf{y}_i - \mathbf{y}_j\|^2}{\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|\mathbf{y}_i - \mathbf{y}_k\|^2}{\sigma_i^2}\right)}, \quad (28)$$

is the probability of point j being a neighbor of i in the output space. If λ is set to one then the NeRV cost function will be reduced to the SNE cost function and it tries to maximize recall. With $\lambda = 0$ the cost function tries to maximize precision. The width of the Gaussian neighborhood σ_i is set in a similar way as in the SNE algorithm. Either a suitable value is selected manually or it is optimized to set the effective number of neighbors for each data point. In our experiments a suitable value for the number of effective neighbors has been around 20.

For optimizing the NeRV cost function we have used the following strategy. The optimization is started with a relatively large σ that is the same for each data point. A few gradient optimization steps are preformed and then σ is reduced. This is repeated a few times. When σ becomes smaller than σ_i the

reduction of the width of the Gaussian for the data point i is stopped and the width is set to σ_i. Finally a longer optimization is performed keeping all σ_i:s constant at their desired final values. In our experiments we have used conjugate gradient optimization [9] to perform the optimization steps, but any gradient based optimization method could be used.

The effect of λ is illustrated in Figure 17 on the Sphere data set. When $\lambda = 0$ the method focuses on optimizing precision and the sphere is split open. Increasing λ pulls the sides of the split sphere closer until they start to get mixed within the central area. At this point small artefacts, empty spaces, appear in the inner part of the image. These are caused by the first part of the cost function that tries to increase precision by pushing the offending points further away from each other. Finally, the sides of the sphere are pulled together and the sphere appears to have become squashed flat. When compared to other methods NeRV has outperformed them in terms of both trustworthiness and continuity (see Publication 7).

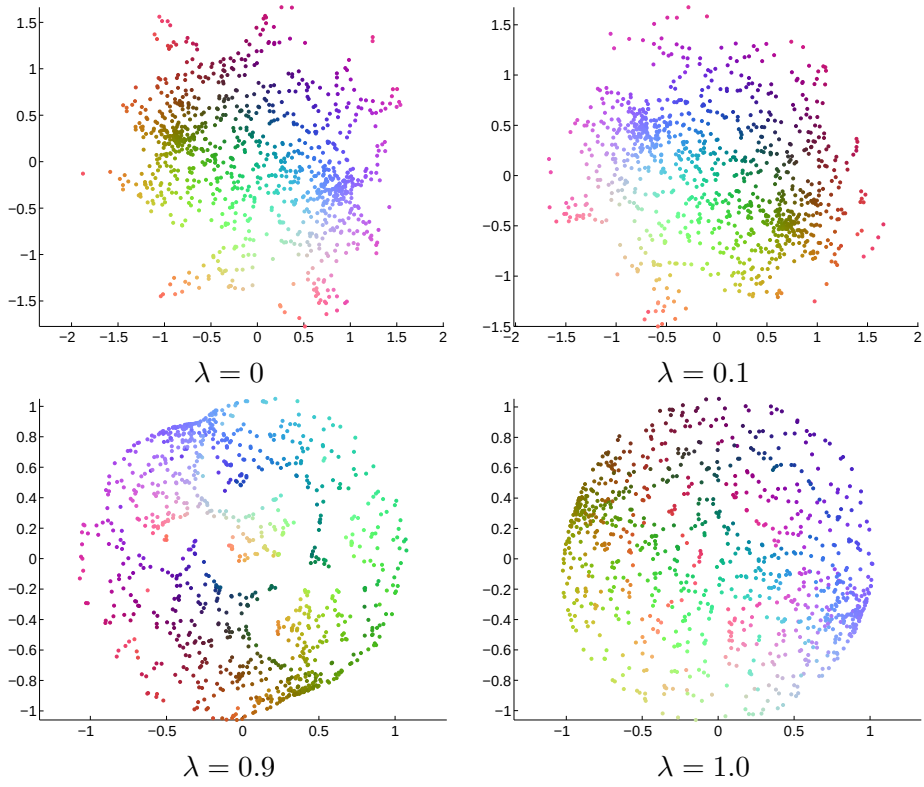


Figure 17: The effect of λ on the NeRV projections of the Sphere data set.When $\lambda = 0$ the sphere is split open. Increasing λ causes the sides to be pulled closer together and finally the sphere appears to have become squashed flat.

The main drawback of the NeRV algorithm is its computational complexity. Each gradient step is of complexity $\mathcal{O}(n^3)$, where n is the number of

data points. This makes NeRV only practical for relatively small data sets. One possibility for reducing the computational complexity of NeRV is to use landmarks [42] to reduce the number of distances that are included in the computation. The idea is to select a sample of data points, landmarks, and only compute pairwise distances between the landmarks and from landmarks to other points. Another strategy is to change the cost function to an approximative one.

Zhu and Rohwer [172] have developed an information-geometric extension of the Kullback-Leibler divergence that is valid for all positive measures instead of just normalized ones. The extended divergence is

$$D_{KLe}(p_i, q_i) = \sum_{j \neq i} q_{ij} - p_{ij} + p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{29}$$

By replacing the Kullback-Leibler divergences in the NeRV cost function with the extended divergence we can use the exponential density values in p_{ij} and q_{ij} (28) without the normalization. This reduces the complexity of the gradient step to $\mathcal{O}(n^2)$ which is comparable with other distance based methods. The method optimizing this approximate cost function is called *fast Neighbor Retrieval Visualizer (fNeRV)*. As can bee seen by comparing Figures 17 and 18 fNeRV behaves similarly to NeRV when λ is varied (for a more in-depth comparison see Publication 7).

4.2 Local Multidimensional Scaling (LocalMDS)

In the previous section a dimensionality reduction method, NeRV, was introduced. It optimizes a quality measure derived from the visual neighbor retrieval task. The disadvantage of the method is its high computational cost. In Publication 5 a new method, called *Local Multidimensional Scaling (LocalMDS)*, is introduced. Although it was originally introduced before NeRV, it can be thought of as a more heuristic but faster method aimed at achieving the same goals. It is a derivative of CCA (Section 2.1.2) with the ability to control the tradeoff between trustworthiness and continuity of the mapping.

The CCA cost function,

$$E = \frac{1}{2} \sum_i \sum_{j \neq i} (d(\mathbf{x}_i, \mathbf{x}_j) - d(\mathbf{y}_i, \mathbf{y}_j))^2 F(d(\mathbf{y}_i, \mathbf{y}_j), \sigma_y) \ , \tag{30}$$

penalizes errors in preserving distances of points that are neighbors in the *output space*. By disregarding errors in distances between points that are far from each other in the visualization but close by in the input space, CCA allows discontinuities to be created in the mapping. These discontinuities allow better preservation of local distances on both sides of the "cut" which usually leads to good trustworthiness in the resulting visualization. On the other hand, the discontinuities reduce the continuity of the mapping.

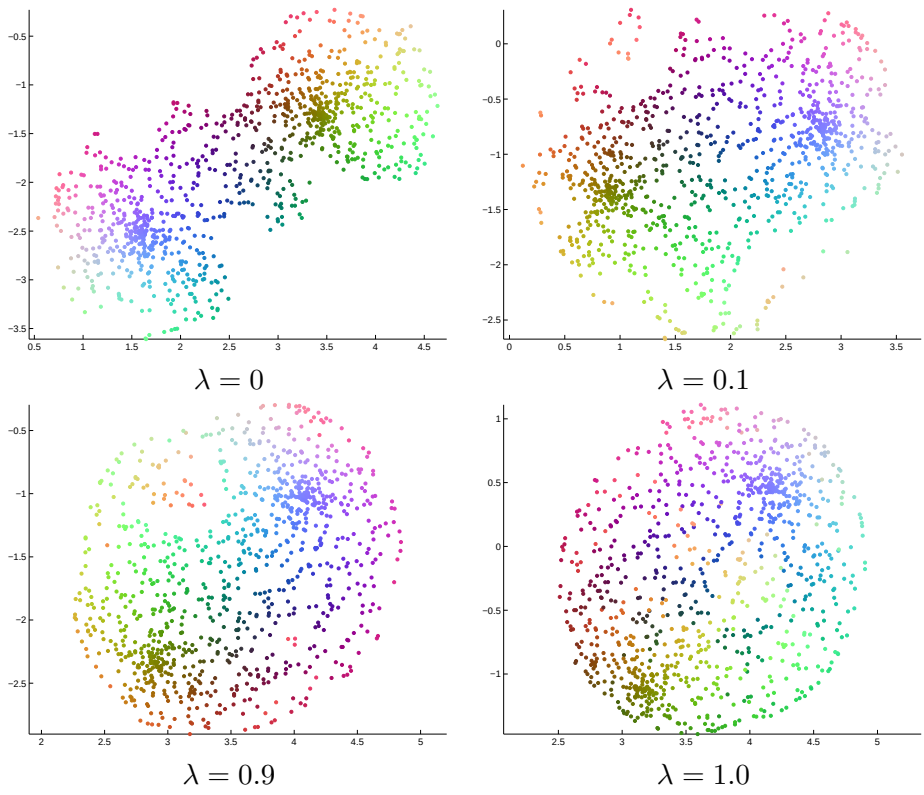
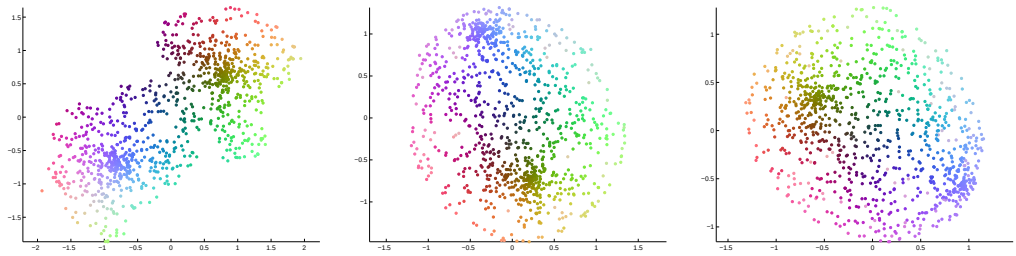


Figure 18: The effect of λ on the fNeRV projections of the Sphere data set.When $\lambda = 0$ the sphere is split open. Increasing λ causes the sides to be pulled closer together and finally the sphere appears to have become squashed flat.



$\lambda = 0$ $\lambda = 0.1$ $\lambda = 0.5$

Figure 19: The effect of λ on the LocalMDS projections of the Sphere data set.When $\lambda = 0$ the sphere is split open and spread out. When λ is increased the sides get pulled closer together.

The idea is to add a term to the cost function that discourages formation of discontinuities. This is achieved by additionally penalizing errors in distances between points that are close by in the *input space*. The tradeoff between these two terms governs the tradeoff between trustworthiness and continuity. The cost function of LocalMDS is

$$\begin{aligned} E &= \frac{1}{2}\sum_i \sum_{j\neq i} [(1-\lambda)(d(\mathbf{x}_i,\mathbf{x}_j) - d(\mathbf{y}_i,\mathbf{y}_j))^2 F(d(\mathbf{y}_i,\mathbf{y}_j),\sigma_i) + \\ &\qquad + \lambda(d(\mathbf{x}_i,\mathbf{x}_j) - d(\mathbf{y}_i,\mathbf{y}_j))^2 F(d(\mathbf{x}_i,\mathbf{x}_j),\sigma_i)] \\ &= \frac{1}{2}\sum_i \sum_{j\neq i} (d(\mathbf{x}_i,\mathbf{x}_j) - d(\mathbf{y}_i,\mathbf{y}_j))^2 \times \\ &\qquad \times [(1-\lambda)F(d(\mathbf{y}_i,\mathbf{y}_j),\sigma_i) + \lambda F(d(\mathbf{x}_i,\mathbf{x}_j),\sigma_i)] \ , \end{aligned} \quad (31)$$

where the parameter $\lambda \in [0 \ldots 1]$ controls the tradeoff. When λ is set to zero the cost function is reduced to that of the basic CCA algorithm, with the exception that the neighborhood final radii are set differently for each data point. A good setting for λ is usually in the range $[0 \ldots 0.5]$. The cost function is optimized with the stochastic gradient descent introduced for CCA in [44]. As in the CCA algorithm, during the optimization the radius of the area of influence around data point i, σ_i, is slowly reduced. The final radius is set equal to the distance of the k:th nearest neighbor of the data point i in the original space. In practice a value around $k = 20$ has proven to be a good choice on the data sets tested.

The effect of λ on visualizations is illustrated in Figure 19 on the Sphere data set. When $\lambda = 0$ the sphere is split open and the parts are spread out. Increasing λ has the effect of pulling the sides of the split closer together. In experiments (see Publications 5 and 7) LocalMDS has achieved results comparable or better than other methods tested except NeRV, in terms of both trustworthiness and continuity. While the performance of LocalMDS in not quite as good as NeRV's it makes up for this by being faster.

4.3 Similarity-based color coding and graph layouts

There are two additional applications that NeRV, fNeRV and LocalMDS could be used for in information visualization. The first application is similarity-based color coding and the second is creating graph layouts.

Similarity-based color coding. Color has traditionally been used in information visualization to code values by mapping the values of a variable to colors on a color scale. In some cases two variables have been mapped to a color at the same time by using a specially developed two-dimensional color scale. In a few cases three variables have been linked with the three components of the color space to produce a three dimensional color space. This approach was used in the Figures of Section 2, where the original coordinates of each data point were coded using the red, green and blue component of the RGB color space. A more profound discussion on traditional color scales can be found in [125] and [156].

In this work a novel approach to color coding is introduced. Instead of mapping the variables directly to colors or color components the data points are assigned colors based on their similarity. The goal is to define a color mapping where similar data points have similar colors and dissimilar data points have colors that are perceived to be different.

In Publication 1 a similarity-based color coding method was introduced for the Self-Organizing Map. The U-matrix visualization commonly used with the SOM (see Section 2.3.1) can be somewhat unintuitive to a user who is not familiar with the methodology. The goal was to create a visualization method that is both intuitive and conveys the same information that the U-matrix visualization does. The added benefit of using color coding is that the similarity structure found by the Self-Organizing Map can be easily connected to some other display of the data. This is illustrated in Figure 20, where the colors associated with the SOM units have been used to color a geographical map of the world. Similar colors indicate that the countries are similar in terms of welfare. In the literature there are two other approaches for creating similarity-based color codes for the SOM. The first uses a contraction model [71] and the second is a very simple approach aimed for SOM-based geovisualization [65].

The problem with using the SOM to create the color code for the data is that several data points are assigned the same color even though they are not exactly similar. We could easily apply NeRV, fNeRV and LocalMDS to project the data points into a color space. This could be done using two strategies. The first one is to project the data to two or three dimensions and then scale and rotate the projection so that it fits the color space. The other strategy is to define a penalty function for points that lie outside the gamut of the output device, that is the set of colors reproducible on the device, and optimize the fit to the gamut together with the dimensionality reduction. Although this

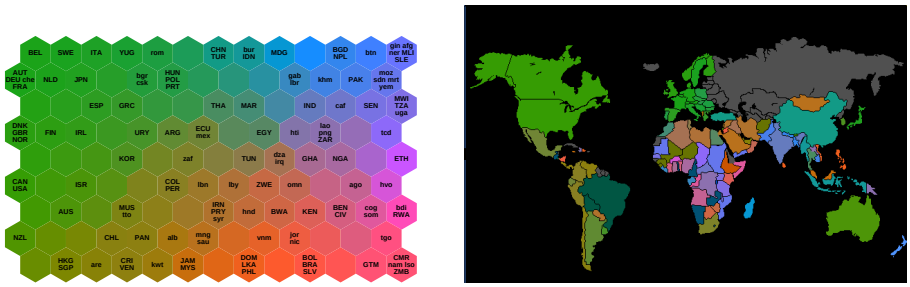


Figure 20: The Self-Organizing Map of world poverty (Publication 1 Figure 2). On the left: the SOM grid has been color-coded according to the similarity of the units. On the right: a geographical display of the same data. The colors illustrate the similarities found by the SOM.

was the strategy used for coloring the SOM in Publication 1, the projection method used in the publication depends on the order defined by the SOM grid and would not be applicable in a more general setting.

Creating graph layouts Graph visualization, or graph drawing, is a specialized area of information visualization. The goal is to find positions for the nodes in a graph in such a way that the resulting layout fulfills some goodness criteria. Although many goodness criteria have been used, most of them are based on purely esthetic values. The only criterion that has been shown to be clearly related to the readability of graphs in controlled usability studies is the number of edge crossings [122, 157], that is, the number of times the edges of a graph cross each other in the visualization. Unfortunately optimizing this criterion is a computationally very hard problem and typically more heuristic methods have to be applied.

There are some preliminary results (see Publication 6) that the LocalMDS algorithm outperforms some of the graph layout methods used for general undirected graphs. It is conceivable that NeRV might be even better for this task. An example of a graph layout produced by NeRV is given in Figure 21. To produce graph layouts with these methods the pairwise shortest path distance matrix is first calculated and then used as input for the dimensionality reduction method. The output is the locations of the nodes in the visualization.

4.4 Discussion

The three nonlinear dimensionality reduction methods, NeRV, fNeRV and LocalMDS, introduced above, are aimed at visualizing similarity structures in the data. On several data sets they have been found to outperform other dimensionality reduction methods in terms of trustworthiness and continuity. One drawback that these and the most of the other nonlinear dimensionality reduction methods have is that the basic formulation does not allow adding

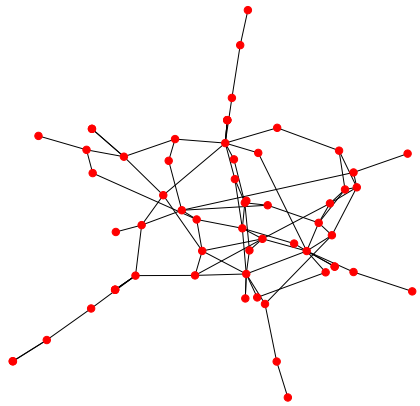


Figure 21: A Layout of the Lee data graph (see Publication 6). The layout was produced with NeRV, $\lambda = 0.4, k = 15$. There are 32 edge crossings in this layout and the graph trustworthiness and continuity are 0.99 and 0.96 respectively (for more details on the measures see Publication 6).

new points to the visualization without recomputing the whole solution. There are two strategies on how this can be solved. First, it is possible to run the algorithm keeping all other points constant and only optimize the locations of the new points. The problem with this approach is that the new points easily get stuck in local optima. The second strategy is to specify that each point in the original visualization is a model vector that represents an area around it in the original space. When a new point arrives we assign it to the location of the best matching point in the visualization. The disadvantage of this approach is that the positions of new points are restricted to lie only at the locations of the points in the original data set. On the other hand, this method allows visualization strategies that are commonly used in conjuction with the Self-Organizing Map. For example we might show the trajectory of process states on the visualization in real time for process monitoring purposes.

In the definition of the visual neighbor retrieval task (Section 3.2) it was assumed that there is no prior information on which data points are selected for neighbor retrieval and thus the errors in retrieving neighbors of each data point are treated equally. In many cases, however, there is some prior information available. For example, when analyzing gene expression data we would usually be more interested in the unknown genes than in the known ones. In such a case the visualization method should focus on preserving the neighborhoods of the points more likely to be analyzed. This additional knowledge can be incorporated into the NeRV cost function by assigning each point a probability P_i of the point being selected for analysis. The NeRV cost function would then

become

$$E_{\mathrm{NeRV}} = \lambda \sum_i P_i \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}} + (1-\lambda) \sum_i P_i \sum_{j \neq i} q_{ij} \log \frac{q_{ij}}{p_{ij}}. \qquad (32)$$

In the original cost function (27) the expectation is taken over the implicitly assumed uniform distribution. In effect, here the uniform distribution is replaced with the distribution of points being selected for analysis. A similar change could also be made to LocalMDS and fNeRV.

Other possible extensions to NeRV include all extension of SNE. For example the Parametric Embedding [77] approach for visualizing co-occurrence data and the multiple relational embedding extension of Mevisevic and Hinton [109] could easily be applied to both NeRV and fNeRV. The LocalMDS algorithm could possibly benefit from changing the neighborhood from a step function to a Gaussian. This could allow for an even better control of the tradeoff between trustworthiness and continuity.

5 USING DIMENSIONALITY REDUCTION TO STUDY RESULTS AND CONVERGENCE OF MCMC SAMPLING

Bayesian modeling (see [16] and [57] for a more in-depth description) is a principled way of incorporating prior knowledge into data analysis. The uncertainty in the data, $\mathbf{D}$, is transformed into uncertainty in the model parameters, $\boldsymbol{\theta}$, with the use of the *Bayes Formula*

$$p(\boldsymbol{\theta}|\mathbf{D},\mathcal{M}) = \frac{p(\mathbf{D}|\boldsymbol{\theta},\mathcal{M})p(\boldsymbol{\theta}|\mathcal{M})}{\int p(\mathbf{D}|\boldsymbol{\theta},\mathcal{M})p(\boldsymbol{\theta}|\mathcal{M})d\boldsymbol{\theta}}, \tag{33}$$

where $\mathcal{M}$ is the model, $p(\boldsymbol{\theta}|\mathbf{D},\mathcal{M})$ is the *posterior* distribution of the model parameters, $p(\mathbf{D}|\boldsymbol{\theta},\mathcal{M})$ is the likelihood term, $p(\boldsymbol{\theta}|\mathcal{M})$ is the prior distribution of the parameters that encodes our knowledge before seeing the data, and $\int p(\mathbf{D}|\boldsymbol{\theta},\mathcal{M})p(\boldsymbol{\theta}|\mathcal{M})d\boldsymbol{\theta}$ is a normalizing term. The posterior distribution encodes our knowledge of the situation after seeing the data. Unfortunately the posterior distribution can only be solved in closed form for very simple models and in practice some approximation method has to be used. Common approaches for approximating the posterior are variational methods [81] and different sampling methods such as importance sampling [57] and Markov Chain Monte Carlo (MCMC) [60] sampling. Of these we will focus on MCMC sampling which is the most commonly used method for handling more complex problems.

MCMC sampling is a very versatile yet computationally intensive procedure, which produces samples of parameter values from the posterior distribution. Probably the two most commonly used MCMC algorithms are the Metropolis-Hastings [67] algorithm and the Gibbs sampling [56] algorithm. In the Metropolis-Hastings algorithm samples are drawn from a jump or proposal distribution $J_t(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1})$. The sample is accepted with the probability

$$P_{accept} = \frac{p(\boldsymbol{\theta}^*|\mathbf{D})J_t(\boldsymbol{\theta}^{t-1}|\boldsymbol{\theta}^*)}{p(\boldsymbol{\theta}^{t-1}|\mathbf{D})J_t(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1})}, \tag{34}$$

otherwise the previous sample is repeated. This produces samples that are eventually distributed according to the posterior distribution $p(\boldsymbol{\theta}|\mathbf{D})$. The Gibbs sampler differs from the Metropolis-Hastings algorithm in two ways. First, it cycles through the parameters and draws only one parameter at a time. Secondly, the sample is drawn from the conditional distribution of the variable being sampled given all other variables and data. The sample is accepted each time. The Gibbs sampler can also be seen as a special case of Metropolis-Hastings algorithm where the jump distribution is replaced by the conditional distribution. This makes the acceptance probability P_{accept} equal to one. Another variant of the Metropolis-Hastings algorithm is the Reversible Jump MCMC [64] algorithm which allows sampling from models

with different complexities during the same sampling run. The result is a sample of the posterior distribution over different models in addition to the normal posterior distribution of the parameters of the models.

All of these sampling methods have a common problem in practice. The samples come from the true posterior distribution only after the sampler has converged. Convergence often takes only a few samples on simple models but it can also be very slow and take thousands of samples on more complicated models.

5.1 Assessing convergence of a MCMC simulation

There are several strategies for monitoring convergence (for reviews and new developments see [2, 24, 25, 49]). A common approach is to start the simulation from several different initial conditions and to measure when the different simulation chains become sufficiently mixed together. A slight variation of this idea is to monitor different parts of the same simulation chain. In both approaches the idea is to monitor how some statistic varies between the different chains or different parts of the chain. If the simulation has converged, the statistic is the same in all chains or parts. Two of the most commonly used convergence measures are the Potential Scale Reduction factor (PSRF) [58] and its multivariate extension [23].

5.1.1 Univariate PSRF

The Potential Scale Reduction Factor (PSRF) was proposed by Gelman and Rubin [58]. Multiple MCMC sequences are started from different (overdispersed) initial points. At convergence the chains should come from the same distribution, which is assessed by comparing the variance and mean of each chain to those of the combined chain.

The PSRF is defined as follows. A number (m) of parallel chains are started, with $2n$ samples each. Only the last n samples from each chain are used.

The between-chain variance B/n and pooled within-chain variance W are defined by

$$\frac{B}{n} = \frac{1}{m-1}\sum_{j=1}^{m}\left(\bar{\theta}_{j\cdot} - \bar{\theta}_{\cdot\cdot}\right)^2 \quad \text{and} \tag{35}$$

$$W = \frac{1}{m(n-1)}\sum_{j=1}^{m}\sum_{t=1}^{n}\left(\theta_{jt} - \bar{\theta}_{j\cdot}\right)^2 \ , \tag{36}$$

where θ_{jt} is the parameter value of the t:th sample in the j:th chain, $\bar{\theta}_{j\cdot}$ is the mean of the samples in chain j and $\bar{\theta}_{\cdot\cdot}$ is the mean of the combined chains.

By taking the sampling variability of the combined mean into account we get a pooled estimate for the posterior variance

$$\hat{V} = \frac{n-1}{n}W + \left(1 + \frac{1}{m}\right)\frac{B}{n}. \tag{37}$$

Finally an estimate $\hat{R}$ of PSRF is obtained by dividing the pooled posterior variance estimate with the pooled within chain variance,

$$\hat{R} = \frac{\hat{V}}{W} = \frac{n-1}{n} + \left(1 + \frac{1}{m}\right)\frac{B}{nW}\,. \tag{38}$$

If the chains have converged, the PSRF is close to one. A rule of thumb says that the simulation has converged if the value is below 1.2 [57]. The measure does not guarantee convergence, however. The chains might not have traveled the whole state space yet and might still be able to discover possible new areas of high probability if the sampling was continued.

5.1.2 Multivariate PSRF

One weakness of the PSRF measure is that it is only applicable to one variable at a time. Brooks and Gelman [23] have extended it to a multivariate version, MPSRF. It is defined, similarly to the univariate PSRF, in terms of the estimate of the posterior covariance matrix

$$\hat{\mathbf{V}} = \frac{n-1}{n}\mathbf{W} + \left(1 + \frac{1}{m}\right)\frac{\mathbf{B}}{n}\,, \tag{39}$$

which we get from (37) by replacing the scalar variances B/n and W with the corresponding covariance matrices

$$\frac{\mathbf{B}}{n} = \frac{1}{m-1}\sum_{j=1}^{m}\left(\bar{\boldsymbol{\theta}}_{j\cdot} - \bar{\boldsymbol{\theta}}_{\cdot\cdot}\right)\left(\bar{\boldsymbol{\theta}}_{j\cdot} - \bar{\boldsymbol{\theta}}_{\cdot\cdot}\right)^T \text{ and} \tag{40}$$

$$\mathbf{W} = \frac{1}{m(n-1)}\sum_{j=1}^{m}\sum_{t=1}^{n}\left(\boldsymbol{\theta}_{jt} - \bar{\boldsymbol{\theta}}_{j\cdot}\right)\left(\boldsymbol{\theta}_{jt} - \bar{\boldsymbol{\theta}}_{j\cdot}\right)^T. \tag{41}$$

In the multivariate case the comparison of within-chain variance to the pooled variance requires comparing the matrices. There are several ways how this could be done. Brooks and Gelman chose to summarize the comparison by a maximum root statistic which gives the maximum scale reduction factor of any linear projection of $\boldsymbol{\theta}$. The estimate $\hat{R}^p$ of MPSRF is defined by

$$\hat{R}^p = \max_{\mathbf{a}}\frac{\mathbf{a}^T\hat{\mathbf{V}}\mathbf{a}}{\mathbf{a}^T\mathbf{W}\mathbf{a}} \tag{42}$$

$$= \frac{n-1}{n} + \left(1 + \frac{1}{m}\right)\max_{\mathbf{a}}\frac{\mathbf{a}^T\mathbf{B}\mathbf{a}/n}{\mathbf{a}^T\mathbf{W}\mathbf{a}} \tag{43}$$

$$= \frac{n-1}{n} + \left(1 + \frac{1}{m}\right)\lambda_1, \tag{44}$$

where the λ_1 is the largest eigenvalue of the matrix $\mathbf{W}^{-1}\mathbf{B}/n$.

5.2 Visualizing convergence of a MCMC simulation

The convergence measures discussed in Section 5.1 cannot tell *why* simulations did not converge and in some cases they might even be fooled to falsely indicate convergence. It is therefore a common practice to complement the measures with *visualizations* of the MCMC chains. The visualizations help to assess convergence in detail and help to analyze reasons for convergence problems. On the other hand, visualizations are not very good for monitoring convergence and should only be used to check the results and to identify possible causes for convergence problems.

MCMC chains have traditionally been visualized in three ways (Figure 22) Each parameter of the posterior samples can be plotted as a separate time series, or the marginal distributions can be visualized as histograms. The third option is a scatter or contour plot of two parameters at a time, possibly showing the trajectory of the chain on the projection. The obvious problem with these visualizations is that they do not scale up to large models having lots of parameters. The number of displays would be large, and it would be hard to grasp the underlying high-dimensional relationships of the chains based on the component-wise displays.

Some new methods have been suggested. Advanced computer graphics methods can be used to visualize the shape of a three-dimensional distribution [158]. Alternatively, if the outputs of the models can be visualized in an intuitive way, an animation of the MCMC chain can be created. Each frame is a visualization of an individual sample [96]. However, these visualizations are applicable only to special models.

The worst problem with the straightforward visualization methods is that they lack the means to focus on visualizing variables or dimensions that are relevant for convergence. This worsens the problems caused by the required large number of plots. There has been some work on developing methods that are aimed for visual assessment of convergence [111, 168]. These methods plot a statistic that is related to the convergence of the model and the convergence is then assessed from the shape of the curve. The problem with these approaches is that they rely on an aggregate statistic that is plotted instead of plotting the produced samples, and are thus of little help in finding out causes of convergence problems.

5.2.1 Principled visualization of MCMC convergence with LDA

The goal of Linear Discriminant Analysis (LDA) [142] is to find the linear transformation $y = \mathbf{a}^T\boldsymbol{\theta}$ that maximizes the variance between classes, relative to the variance within classes. Here $\boldsymbol{\theta}$ is the multivariate data vector and $\mathbf{a}$ contains the parameters of the transformation. More formally, LDA solves the

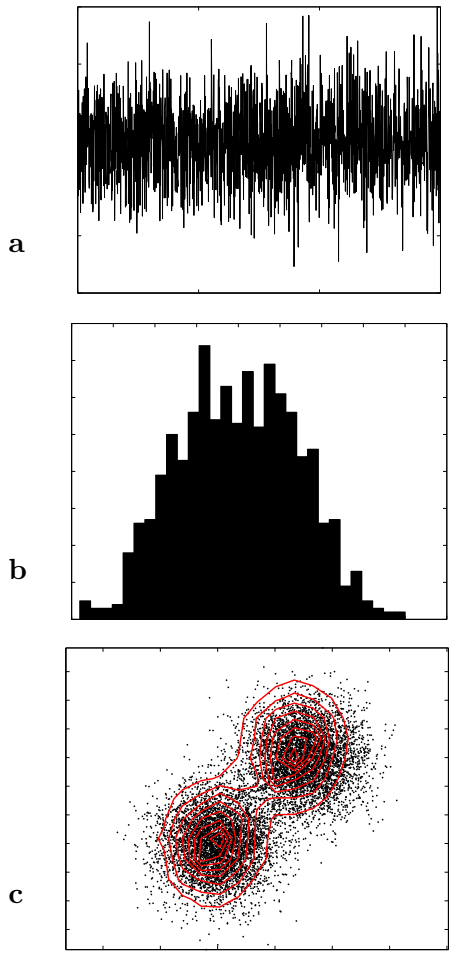


Figure 22: Common visualization methods used for visualizing convergence and results of MCMC simulations. **a**) Time series plots of variables. **b**) Histogram plots of marginal distributions. **c**) Scatter or contour plots of the joint marginal distribution of two variables.

problem

$$\max_{\mathbf{a}} \frac{\mathbf{a}^T \mathbf{B}_{ss} \mathbf{a}}{\mathbf{a}^T \mathbf{W}_{ss} \mathbf{a}} , \tag{45}$$

where $\mathbf{B}_{ss}$ and $\mathbf{W}_{ss}$ are the between class and within class sum of squares and cross products (SSCP) matrices which differ only by a constant scale from the corresponding covariance matrices (40) and (41). This is a generalized eigenvalue problem, and its solution $\mathbf{a}$ is the eigenvector corresponding to the largest eigenvalue of $\mathbf{W}_{ss}^{-1}\mathbf{B}_{ss}$.

Hence, disregarding the constants, MPSRF (43) equals the cost function of LDA where the MCMC chains are the classes. In other words, optimizing LDA is equivalent to choosing the component that best detects convergence, in the sense of the MPSRF. Projection of the samples using the first LDA component produces a projection that best separates the samples of different chains. There is no reason why only one component should be used for the projection, however. By using two components a scatter plot image can be produced. The only difference from the traditional plot of the joint marginal distribution of two parameters is that now the image axes are linear combinations of variables and the separation of different chains is maximized. Figure 23 illustrates how LDA can be used to visualize MCMC convergence problems. In this case some of the chains (2, 8, 9, 10) have got stuck in one mode in a multimodal posterior distribution and are not mixing with the others. They are clearly separated from the others in the visualization. A more profound description of the simulation and analysis of the results can be found in Publication 9.

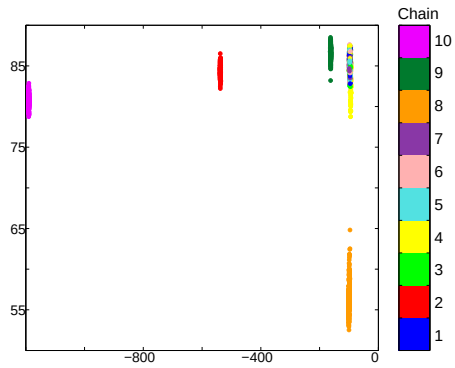


Figure 23: Two-dimensional LDA projection of samples from a MCMC simulation. Four of the chains are clearly separated from the others. Each chain has been given a unique color.

Problems with LDA. Like the PSRF and MPSRF measures, LDA compares only means and covariances. In addition, LDA assumes that each class is normally distributed and has the same covariance matrix. If these assumptions are correct, LDA discriminates optimally between two classes. However, this does not hold in general. Another problem surfaces when generalizing

LDA to several classes. The objective considers only pairwise divergences between classes, and does not result in a direction that discriminates optimally between all classes (for more information see the appendix of Publication 9). This is most evident in a situation where one of the classes is far away from the other classes. In such a case the first component of LDA will focus on separating the far away class from the others, mostly disregarding the variation between the nearby classes.

In the specific case of visualizing MCMC convergence, the posterior distributions being simulated are often not Gaussian, especially when there is only a small amount of data. Furthermore, the MCMC chains do not have the same covariance matrix before convergence. Therefore, although LDA often works well in practice, it is suboptimal.

5.2.2 Relevant Component Analysis

These shortcomings can be overcome by using a recently introduced generalization of LDA, Relevant Component Analysis (RCA; the method is also called Informative Discriminant Analysis or Discriminative Component Analysis)[119]. It finds *discriminative* components by directly maximizing their class-prediction power. Formally, the conditional (log) likelihood

$$L = \sum_{(\boldsymbol{\theta},c)} \log p(c|\mathbf{W}^T\boldsymbol{\theta}) \tag{46}$$

of classes is maximized within the subspace formed by the components. Here $\boldsymbol{\theta}$ is the sample, c is the chain, and $\mathbf{W}$ is the (orthogonal) projection matrix whose columns are the component directions. More details on the method and a sketch of the connection between LDA and RCA is presented in Publication 9.

The disadvantage of using RCA instead of LDA to visualize convergence problems is its higher computational cost. The optimization method used is iterative and there are several parameters in the current method, that have to be set by using cross-validation [133, 136] techniques. It is still somewhat unclear in which practical situations it is better to use LDA and in which RCA. The new sped-up variant of RCA [118] has partly solved the problem, however. Recently other linear discriminant methods such as the Large Margin Component Analysis algorithm [144] have been introduced and they could also be used to visualize convergence problems. In addition some nonlinear discriminant methods could also be utilized, but care should be taken to keep the mappings from overfitting the data. A too flexible mapping will discriminate between points, even if the chains come from the same distribution, given a chain of only a finite length.

5.3 Visualizing the result of a MCMC simulation using the Fisher metric

Besides using visualizations to analyze problems in convergence of MCMC simulations, they can also be used to study the posterior distribution once the simulation has converged. If the posterior distribution is high-dimensional the traditional methods of visualization of posterior distributions, illustrated in Figure 22, suffer from the large number of images needed to cover all parameters. It is also easy to find examples where the histograms of marginal distributions or even low-dimensional linear projections of a high-dimensional posterior distribution can give a misleading picture of the shape of the distribution (for an illustration see Publication 8). In such cases it can be beneficial to use nonlinear projection methods such as the ones described in Section 2 to visualize the distribution.

In many cases the goal of Bayesian analysis is not only to find the posterior distribution of a certain parameter, but to utilize the whole posterior for decision making or prediction of future values. In fact, Bayesian analysis can be derived from the need to make rational decisions [16]. In such cases the parameters of the model are not the focus of the study and the model can be reparametrized, for example to have better convergence properties, without any effect on the decisions or predictions.

Traditional visualizations of the posterior distribution depend on the parameterization of the model. If the parameterization is changed the visualization changes even though the resulting model does not change. A related problem is that a straightforward visualization of the posterior does not reflect the effect the parameters have on the model. A small change of parameter values might cause a large change in the model output if the model is sensitive in that part of the parameter space and vice versa.

Most visualization methods take the parameterization of the model and the metric it induces as given without any regard on how they effect the model. A parameter that has no effect on the model is given as much weight in forming the visualization as the one that is very relevant. In Publication 9 a new approach is introduced.

The difference of two distributions can be measured with the Kullback-Leibler divergence D_{KL}. If the distributions are close-by the divergence is

$$D_{KL}(p(\tilde{\mathbf{D}}|\boldsymbol{\theta}), p(\tilde{\mathbf{D}}|\boldsymbol{\theta} + d\boldsymbol{\theta})) = d\boldsymbol{\theta}^T \mathbf{J}(\boldsymbol{\theta}) d\boldsymbol{\theta}, \tag{47}$$

where $\mathbf{J}$ is the Fisher information matrix

$$\mathbf{J}(\boldsymbol{\theta}) = -E_{p(\tilde{\mathbf{D}}|\boldsymbol{\theta})} \left[\left(\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\tilde{\mathbf{D}}|\boldsymbol{\theta}) \right) \left(\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\tilde{\mathbf{D}}|\boldsymbol{\theta}) \right)^T \right]. \tag{48}$$

This defines the so called *Fisher* or *information metric* [5, 84] of the parameter space. Distances between two far-away points are defined as path

integrals along the shortest path between the points. The path integrals cannot usually be calculated in closed form and have to be approximated. A very simple local approximation can be formed by calculating the metric based on the Fisher matrix either at one or both of the endpoints of the path only. This approach has been used in Publication 8 and in [83]. This simple approximation is suitable for visualization methods that rely mostly on local features. On methods that require all pairwise distances more complex approximations are required. Examples of approximations using graph distances or multiple evaluations of the Fisher Matrix along the direct path between points can be found in [120].

Publication 8 introduces a Self-Organizing Map algorithm that uses Fisher distances of the model family for visualizing high-dimensional posterior distributions in a way that reflects the effect the parameters have on the model. For example, if some parameter has only a small effect on the model we might remove it from the model altogether. Or we might not consider it at all when checking the convergence of the simulation. On the other hand, we might identify parameters or parameter ranges where the model is very sensitive and then pay more attention to them. Any other distance-based projection method could also be utilized in a similar way. An example in a different context of using Sammon's Mapping in the Fisher metrics is given in [120].

5.4 Discussion

The visualization methods presented here can easily be utilized on many kinds of models, but there are two classes of models that cause problems. The first class is formed of the models that have an inherent unidentifiability built into the model structure, as in mixture models. The order of the components in the mixture can be changed without affecting the model in any way. The result is a multimodal posterior distribution which causes problems for all methods that try to diagnose convergence from several chains or from different parts of the same chain. Typically each chain mostly stays in only one mode (defined by one ordering of the components). Only very seldomly a chain moves between different modes. Comparing the chains based on the parameter values will thus lead to the conclusion that no convergence can be seen. In a strict sense this is true; all chains have not visited all possible modes and thus the distribution is not the true posterior. In practice, however this is a too constrained view. A better way to define convergence would be to say that the simulation has converged if everything except the ordering has converged. One way to solve the problem with unidentifiability is to use post processing to fix the ordering. Stephens has proposed a method to do this [135]. Another possibility might be to study convergence based on the posterior predictive distribution that is not affected by the unidentifiability in the model structure.

The second set of models that poses problems for convergence monitoring and especially visualization are the models with varying dimensionality. Sim-

ulation methods such as reversible jump MCMC [64] and Death and Birth processes [134] allow sampling to traverse through models with different complexities instead of focusing on only one model complexity at a time. The change in the number of parameters between different complexities makes it hard to analyze convergence and especially to visualize the sampling results. It might be possible to create visualizations based on the approaches [22, 30] that extend the MPSRF measure to cases where the complexity of the model is changed. Other possibilities are to use pairwise divergence measures between samples to from a distance matrix and then use MDS methods to form a vector representation that is of a constant dimensionality. A third possibility would be to base the visualization on the posterior predictive distribution.

6 CONCLUSIONS

One of the main problems in information visualization is the lack of suitable measures for assessing the quality of visualizations. A new visualization task, visual neighbor retrieval, that connects information visualization and information retrieval is introduced in this thesis. The connection between these two fields allows concepts from the information retrieval field to be utilized in analyzing the performance of dimensionality reduction methods. Based on these concepts two pairs of new quality measures, trustworthiness and continuity, and smoothed precision and recall, were introduced in this thesis.

Analyzing the behavior of existing dimensionality reduction methods in a visualization context has led to a better understanding of which methods are suitable for which tasks. These insights have also allowed the development of three new methods, NeRV, fNeRV and LocalMDS, which are aimed for the visual neighbor retrieval task. Of these NeRV directly optimizes a compromise between smoothed precision and recall. In experiments all three new algorithms have outperformed other methods in the visual neighbor retrieval task.

In addition to visual data analysis these methods seem to be well suited for two additional tasks: creating similarity-based color scales and producing graph layouts. The similarity-based color scale is a new approach to assigning colors to data points in visualizations. The similarity between data points is coded as a color difference. This allows combining similarity information to some other structure in the data. For example a data set consisting of economic indicators of countries defines the economic similarity between countries. By assigning colors based on this data and then coloring the countries on the world map we can find out how economically similar countries are located geographically. Graph visualization is a field that is gaining momentum. Especially in bioinformatics the number of graph-structured data sets is increasing rapidly. Based on the preliminary results in this theses the new methods, NeRV, fNeRV, and LocalMDS, could be applied to produce good graph layouts.

Markov Chain Monte Carlo simulation is one of the leading methodologies used in Bayesian modeling. Thus far there has been very little work on using visualization methods in analyzing the results. In this theses a new visual approach to analyzing convergence problems and a new method for studying the effect the model parameters have on the model output is introduced.

There are several ways the work presented in this theses can be continued. The new quality measures only measure two aspects of the visualization quality and it would be beneficial to define new visualization tasks and quality measures based on them. Another direction would be to extend the new methods to new fields. The preliminary study where LocalMDS was found to outperform other graph layout methods is a step in this direction. It is still unclear why LocalMDS performs so well in creating graph layouts and further

work is needed. Finally, the visualization approach to studying convergence problems in MCMC simulations could be extended to more complex models.

References

[1] Alex T. Adai, Shailesh V. Date, Shannon Wieland, and Edward M. Marcotte. LGL: Creating a map of protein function with an algorithm for visualizing very large biological networks. *Journal of Molecular Biology*, 340:179–190, 2004.

[2] Salaheddine El Adlouni, Anne-Catherine Favre, and Bernard Bobée. Comparison of methodologies to asses the convergence of Markov chain Monte Carlo methods. *Computational Statistics & Data Analysis*, 50:2685–2701, 2006.

[3] Dimitris K. Agrafiotis. Stochastic Proximity Embedding. *Journal of Computational Chemistry*, pages 1215–1221, 2003.

[4] Robert Amar, James Eagan, and John Stasko. Low-level components of analytic activity in information visualization. In *Proceedings of the 2005 IEEE Symposium on Information Visualizatin (INFOVIS'05)*, pages 111–117, 2005.

[5] Shun-ichi Amari. *Differential-Geometrical Methods in Statistics*. Springer, New York, 1990.

[6] Daniel Asimov. The grand tour: a tool for viewing multidimensional data. *SIAM Journal on Scientific and Statical Computing*, 6:128–143, 1985.

[7] Michaël Aupetit. Visualizing the trustworthiness of a projection. In Michel Verleysen, editor, *Proceedings of the 14th European Symposium on Artificial Neural Networks (ESANN'2006)*, pages 271–276, Evere, Belgium, 2006. d-side.

[8] H.-U. Bauer and K.R. Pawelzik. Quantifying the neighborhood preservation of self-organizing feature maps. *IEEE Transactions on Neural Networks*, 3:570–579, 1992.

[9] Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear programming: Theory and Algorithms*. John Wiley & Sons, 2nd edition, 1993.

[10] 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*, pages 585–591, Cambridge, MA, 2002. MIT Press.

[11] 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.

[12] Mikhail Belkin and Partha Niyogi. Laplacian Eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2003.

[13] Mikhail Belkin and Partha Niyogi. Using manifold stucture for partially labeled classification. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 929–936, Cambridge, MA, 2003. MIT Press.

[14] Mikhail Belkin and Partha Niyogi. Semi-supervised learning on Riemannian manifolds. *Machine Learning*, 56:209–239, 2004.

[15] Yoshua Bengio, Jean-François Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Le Roux, and Marie Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

[16] José M. Bernardo and Adrian F. M. Smith. *Bayesian theory.* John Wiley & Sons, West Sussex, England, 1994.

[17] 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, December 2000.

[18] Enrico Bertini and Giuseppe Santucci. Quality metrics for 2D scatterplot graphics: automatically reducing visual clutter. In Andreas Butz, Antonio Krüger, and Patrick Olivier, editors, *Smart Graphics: Fourth International Symposium on Smart Graphics, SG2004*, Lecture Notes in Computer Science, pages 77–89, Berlin / Heidelberg, 2004. Springer.

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

[20] Matthew Brand. Charting a manifold. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[21] R. Brath. Metrics for effective information visualization. In *Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis '97)*, pages 108–111, 126, Washington, DC, USA, 1997. IEEE Computer Society.

[22] S. P. Brooks and P. Giudici. MCMC convergence assesment via Two-Way ANOVA. *Journal of Computational and Graphical Statistics*, 9:266–285, 2000.

[23] Stephen Brooks and Andrew Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7:434–456, Dec 1998.

[24] Stephen Brooks and Andrew Gelman. Some issues in monitoring convergence of iterative simulations. In *Proceedings of the Section on Statistical Computing.* ASA, 1998.

[25] Stephen P. Brooks and Graeth O. Roberts. Convergence assesment techniques for Markov chain Monte Carlo. *Statistics and Computing*, 8:319–335, 1998.

[26] J. Bruske and G. Sommer. Intrinsic dimensionality estimation with optimally topology preserving maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:572–575, 1998.

[27] Francesco Camastra and Alessandro Vinciarelli. Estimating the intrinsic dimension of data with a fractal-based method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:1404–1407, 2002.

[28] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers, 1999.

[29] Miguel Á. Carreira-Perpiñán and Richard S. Zemel. Proximity graphs for clustering and manifold learning. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 225–232, Cambridge, MA, 2005. MIT Press.

[30] John M. Castelloe and Dale L. Zimmerman. Convergence assesment for Reversible Jump MCMC samplers. Technical report, Department of Statistics and Actuarial Science, University of Iowa, Feb 2002.

[31] J. M. Chambers, W. S. Cleveland, B. Kleiner, and P.A. Tukey. *Graphical methods for Data Analysis*. Wadsworth, Belmont, CA, 1983.

[32] Chaomei Chen. Top 10 unsolved information visualization problems. *IEEE Computer Graphics and Applications*, 25:12–16, 2005.

[33] Herman Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68:361–368, Jun 1973.

[34] G. Choi and S. Choi. Kernel Isomap. *Electronics letters*, 40:1612–1613, 2004.

[35] W.S. Cleveland and R. McGill. The many faces of a scatterplot. *Journal of the American Statistical Association*, 79:807–822, 1984.

[36] Jonathan D. Cohen. Drawing graphs to convey proximity. *ACM Transactions on Computer-Human Interaction*, 4:197–229, 1997.

[37] Jose A. Costa and Alfred O. Hero. Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Transactions on Signal Processing*, 52:2210–2221, 2004.

[38] Jose A. Costa, Neal Patwari, and Alfred O. Hero III. Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks. *ACM Journal on Sensor Networking*, 2:39–64, 2006.

[39] Dick de Ridder, Olga Kouropteva, Oleg Okun, Matti Pietikäinen, and Robert P.W. Duin. Supervised locally linear embedding. In *Proceedings of the Joint International Conference ICANN/ICONIP 2003*, Lecture Notes in Computer Science, pages 333–341, Berlin / Heidelberg, 2003. Springer.

[40] Dick de Ridder, Marco Loog, and Marcel J. T. Reinders. Local Fisher Embedding. In *Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04) Volume 2*, pages 295–298, Washington, DC, USA, 2004. IEEE Computer Society.

[41] Vin de Silva and Joshua B. Tenenbaum. Unsupervised learning of curved manifolds. In D.D. Denison, M. H. Hansen, C. C. Holmes, B. Mallick, and B. Yu, editors, *Proceedings of the MSRI workshop on nonlinear estimation and classification*, pages 453–466, New York, 2002. Springer-Verlag.

[42] Vin de Silva and Joshua B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 705–712, Cambridge, MA, 2003. MIT Press.

[43] Dennis DeCoste. Visualizing Mercer kernel feature spaces via kernelized Locally-Linear Embeddings. In *Proceedings of the 8th International Conference on Neural Information Processing*, 2001.

[44] 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, January 1997.

[45] David DeMers and Garrison W. Cottrell. Non-linear dimensionality reduction. In *Advances in Neural Information Processing Systems 5*, pages 580–587, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[46] Piotr Dollár, Vincent Rabaud, and Serge Belongie. Learning to traverse image manifolds. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

[47] 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.

[48] Pablo A. Estévez, Andrés M. Chong, Claudio M. Held, and Claudio A. Perez. Nonlinear projection using geodesic distances and the neural gas network. In *Proceedings of the 16th International Conference on Artificial Neural Networks (ICANN 2006), Part I*, pages 464–473, 2006.

[49] Y. Fan, S.P. Brooks, and A. Gelman. Output assesment for Monte Carlo simulations via the score statistic. *Journal of Computational and Graphical Statistics*, 35:683–713, 2006.

[50] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annual Eugenics*, 7(Part II):179–188, 179-188.

[51] Carla M. Dal Sasso Freitas, Paulo R. G. Luzzardi, Ricardo A. Cava, Marco A. Winckler, Marcelo S. Pimenta, and Luciana P. Nedel. On evaluating information visualization techniques. In *Advanced Visual Interfaces 2002, AVI'02*, pages 373–374, Trento, 2002.

[52] Jerome H. Friedman and John W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, C-23:881–890, 1974.

[53] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software–Practice and Experience*, 21:1129–1164, 1991.

[54] Keinosuke Fukuganga and David R. Olsen. An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, C-20:176–183, 1971.

[55] Emden R. Gansner and Stephen C. North. An open graph visualization system and its applications to software engineering. *Software–Practice and Experience*, 30:1203–1233, 2000.

[56] Alan E. Gelfand and Adriean F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398–409, 1990.

[57] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Texts in Statistical Science. Chapman & Hall/CRC, Boca Raton, Florida, 1995.

[58] Andrew Gelman and Donald B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, 7:457–472, Nov 1992.

[59] Ali Ghodsi, Jiayuan Huang, Finnegan Southey, and Dale Schuurmans. Tangent-Corrected Embedding. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 518–525, Washington, DC, USA, 2005. IEEE Computer Society.

[60] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Interdisciplinary Statistics. Chapman & Hall/CRC, Boca Raton, Florida, 1995.

[61] Amir Globerson, Gal Chechik, Fernando Pereira, and Naftali Tishby. Euclidean embedding of co-occurrence data. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 497–504, Cambridge, MA, 2005. MIT Press.

[62] Jacob Goldberger, Sam Roweis, Geoffrey Hinton, and Ruslan Salakhutdinov. Neighbourhood Components Analysis. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 513–520, Cambridge, MA, 2005. MIT Press.

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

[64] Peter J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[65] Diansheng Guo, Mark Gahegan, Alan M. MacEachren, and Biliang Zhou. Multivariate analysis and geovisualization with an integrated georaphic knowledge discovery approach. *Cartography and Geographic Information Science*, 32:113–132, 2005.

[66] Jihun Ham, Daniel D. Lee, Sebastian Mika, and Bernhard Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *ICML '04: Proceedings of the Twenty-First International Conference on Machine learning*, page 47, New York, NY, USA, 2004. ACM Press.

[67] W. K. Hastings. Monte Carlo sampling mehtods using Markov chains and their applications. *Biometrica*, 57:97–109, 1970.

[68] Xiaofei He, Deng Cai, Shuicheng Yan, and Hong-Jiang Zhang. Neighborhood Preserving Embedding. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1208–1213, Washington, DC, USA, 2005. IEEE Computer Society.

[69] Xiaofei He and Partha Niyogi. Locality preserving projections. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

[70] Matthias Hein and Jean-Yves Audibert. Intrinsic dimensionality estimation of submanifolds in Rd. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 289–296, New York, NY, USA, 2005. ACM Press.

[71] Johan Himberg. A SOM based cluster visualization and its application for false-coloring. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000)*, volume 3, pages 587–592, 2000.

[72] Johan Himberg and Aapo Hyvärinen. Icasso: software for investigating the reliability of ICA estimates by clustering and visualization. In *13th IEEE Workshop on Neural Networks for Signal Processing (NNSP'03)*, pages 259–268, Toulouse, France, 2003.

[73] Geoffrey Hinton and Sam Roweis. Stochastic neighbor embedding. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 833–840, Cambridge, MA, 2002. MIT Press.

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

[75] A. Inselberg. Multidimensional detective. In *INFOVIS '97: Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis '97)*, page 100, Washington, DC, USA, 1997. IEEE Computer Society.

[76] Kuncup Iswandy and Andreas Konig. Improvement of non-linear mapping computation for dimensionality reduction in data visualization and classification. In *Proceedings of the Fourth International Conference on Hybrid Intelligent Systems (HIS'04)*, pages 260–265, Washington, DC, USA, 2004. IEEE Computer Society.

[77] Tomoharu Iwata, Kazumi Saito, Naonori Ueda, Sean Stromsten, Thomas L. Griffiths, and Joshua B. Tenenbaum. Parametric embedding for class visualization. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 617–624, Cambridge, MA, 2005. MIT Press.

[78] Tomoharu Iwata, Kazumi Saito, and Naorori Ueda. Visual nonlinear discriminant analysis for classifier design. In Michel Verleysen, editor, *Proceedings of the 14th European Symposium on Artificial Neural Networks (ESANN'2006)*, pages 283–288, Evere, Belgium, 2006. d-side.

[79] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, NJ., 1988.

[80] Odest Chadwicke Jenkins and Maja J. Matarić. A spatio-temporal extension to Isomap nonlinear dimension reduction. In *ICML '04: Proceedings of the 21th International Conference on Machine Learning*, pages 441–448, New York, NY, USA, 2004. ACM Press.

[81] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

[82] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31:7–15, 1989.

[83] Samuel Kaski, Janne Sinkkonen, and Jaakko Peltonen. Bankruptcy analysis with self-organizing maps in learning metrics. *IEEE Transactions on Neural Networks*, 12:936–947, 2001.

[84] Robert E. Kass and Paul W. Vos. *Geometrical Foundations of Asymptotic Inference*. Wiley, New York, 1997.

[85] Balázs Kégl. Intrinsic dimension estimation using packing numbers. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 681–688, Cambridge, MA, 2003. MIT Press.

[86] Kimmo Kiviluoto. Topology preservation in self-organizing maps. In *Proceedings of IEEE International Conference on Neural Networks*, volume 1, pages 294–299. IEEE, June 1996.

[87] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

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

[89] Yehuda Koren and Liran Carmel. Robust linear dimensionality reduction. *IEEE Transactions on Visualization and Computer Graphics*, 10:459–470, 2004.

[90] Olga Kouropteva, Oleg Okun, and Matti Pietikäinen. Incremental locally linear embedding. *Pattern Recognition*, 38:1764–1767, 2005.

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

[92] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22:79–86, 1951.

[93] Martin H. C. Law and Anil K. Jain. Incremental nonlinear dimensionality reduction by manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:377–391, 2006.

[94] Martin H. C. Law, Nan Zhang, and Anil K. Jain. Nonlinear manifold learning for data stream. In *Proceedings of SIAM Data Mining*, pages 33–44, 2004.

[95] Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

[96] Nicole A. Lazar and Joseph B. Kadane. Movies for the visualization of MCMC output. *Journal of Computational and Graphical Statistics*, 11:836–874, Dec 2002.

[97] Jeffrey LeBlanc, Matthew O. Ward, and Norman Wittels. Exploring n-dimensional databases. In *Proceedings of the First IEEE Conference on Visualization, (Visualization '90)*, pages 230–237, Oct 1990.

[98] John Aldo Lee, Cédric Archambeau, and Michel Verleysen. Locally Linear Embedding versus Isotop. In *ESANN'2003 Proceedings - European conference on Artificial Neural Networks*, pages 527–534, 2003.

[99] John Aldo Lee, Amaury Lendasse, Nicolas Donckers, and Michel Verleysen. A robust nonlinear projection method. In M. Verleysen, editor, *ESANN'2000, Eighth European Symposium on Artificial Neural Networks*, pages 13–20, Bruges, Belgium, 2000. D-Facto Publications.

[100] John Aldo Lee, Amaury Lendasse, and Michel Verleysen. Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis. *Neurocomputing*, 57:49–76, 2004.

[101] John Aldo Lee and Michel Verleysen. Nonlinear projection with the Isotop method. In *Proceedings of International Conference on Artificial Neural Networks (ICANN'2002)*, Lecture Notes in Computer Science, pages 933–938. Springer, 2002.

[102] John Aldo Lee and Michel Verleysen. Nonlinear dimensionality reduction of data manifolds with essential loops. *Neurocomputing*, 67:29–53, 2005.

[103] Elizaveta Levina and Peter J. Bickel. Maximum likelihood estimation of intrinsic dimension. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 777–784, Cambridge, MA, 2005. MIT Press.

[104] James Xinzhi Li. Visualization of high-dimensional data with relational perspective map. *Information Visualization*, 3:49–59, 2004.

[105] Shawn Martin and Alex Bäcker. Estimating manifold dimension by inversion error. In *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 22–26, New York, NY, USA, 2005. ACM Press.

[106] Thomas Martinetz and Klaus Schulten. Topology representing networks. *Neural Networks*, 7:507–522, 1994.

[107] Yoshitatsu Matsuda and Kazunori Yamaguchi. An efficient MDS-based topographic mapping algorithm. *Neurocomputing*, 64:285–299, 2005.

[108] Peter Meinicke, Stefan Klanke, Roland Memisevic, and Helge Ritter. Principal surfaces from unsupervised kernel regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1379–1391, 2005.

[109] Roland Memisevic and Geoffrey Hinton. Multiple Relational Embedding. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 913–920, Cambridge, MA, 2005. MIT Press.

[110] Nancy Miller, Beth Hetzler, Grant Nakamura, and Paul Whitney. The need for metrics in visual information analysis. In *Proceedings of the 1997 workshop on New paradigms in information visualization and manipulation, NPIV '97*, pages 24–28, New York, NY, USA, 1997. ACM Press.

[111] Per Mykland, Luke Terney, and Bin Yu. Regeneration in Markov Chain samplers. *Journal of the American Statistical Association*, 90:233–241, 1995.

[112] Kijoeng Nam, Hongmo Je, and Seungjin Choi. Fast Stochastic Neighbor Embedding: A trust-region algorithm. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN 2004)*, pages 123–128, 2004.

[113] Antoine Naud. Visualization of high-dimensional data using an association of multidimensional scaling to clustering. In *Proceedings of the 2004 IEEE Conference on Cybernetics and Intelligent Systems*, pages 252–255, 2004.

[114] Man-Suk Oh and Adrian E. Raftery. Bayesian multidimensional scaling and choise of dimension. *Journal of the American Statistical Assosiation*, 96:1031–1044, 2001.

[115] Merja Oja, Samuel Kaski, and Teuvo Kohonen. Bibliography of self-organizing map (som) papers: 1998-2001 addendum. *Neural Computing Surveys*, 3:1–156, 2003.

[116] Merja Oja, Göran O. Sperber, Jonas Blomberg, and Samuel Kaski. Self-organizing map based discovery and visualization of human endogenous retroviral sequence groups. *International Journal of Neural Systems*, 15:163–179, 2005.

[117] Alberto Paccanaro and Geoffrey E. Hinton. Learning distributed representations of concepts using linear relational embedding. *IEEE Transactions on Knowledge and Data Engineering*, 13:232–244, 2001.

[118] Jaakko Peltonen, Jacob Goldberger, and Samuel Kaski. Fast discriminative component analysis for comparing examples. In *NIPS 2006 workshop on Learning to Compare Examples*, Whistler, Canada, December 8 2006.

[119] Jaakko Peltonen and Samuel Kaski. Discriminative components of data. *IEEE Transactions on Neural Networks*, 16:68–83, 2005.

[120] Jaakko Peltonen, Arto Klami, and Samuel Kaski. Improved learning of Riemannian metrics for exploratory analysis. *Neural Networks*, 17:1087–1100, 2004.

[121] John C. Platt. FastMap, MetricMap, and Landmark MDS are all Nyström algorithms. In *10th International Workshop on Artificial Intelligence and Statistics*, pages 261–268, 2005.

[122] H. C. Purchase, R. F. Cohen, and M. I. James. An experimental study of the basis for graph drawing algorithms. *ACM Journal of Experimental Algorithmics*, 2, 1997.

[123] Michael Quist and Golan Yona. Distributional Scaling: An algorithm for structure-preserving embedding for metric and nonmetric spaces. *Journal of Machine Learning Research*, 5:399–420, 2004.

[124] Maxim Raginsky and Svetlana Lazebnik. Estimation of intrinsic dimensionality using high-rate vector quantization. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1105–1112, Cambridge, MA, 2006. MIT Press.

[125] Penny L. Rheingans. Task-based color scale design. In William R. Oliver, editor, *Proceedings of the 28th AIPR Workshop: 3D Visualization for Data Exploration and Decision Making*, volume 3905 of *Proceedings of SPIE*, pages 35–43, 2000.

[126] Volker Roth, Julian Laub, Motoaki Kawanabe, and Joachim M. Buhmann. Optimal cluster preserving embedding of nonmetric proximity data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25:1540–1551, 2003.

[127] Sam Roweis, Lawrence K. Saul, and Geoffrey E. Hinton. Global coordination of local linear methods. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[128] Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by Locally Linear Embedding. *Science*, 290:2323–2326, December 2000.

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

[130] Lawrence K. Saul and Sam T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[131] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[132] Fei Sha and Lawrence K. Saul. Analysis and extension of spectral methods for nonlinear dimensionality reduction. In *Proceedings of the 22nd international conference on Machine learning (ICML '05)*, pages 784–791, New York, NY, USA, 2005. ACM Press.

[133] Jun Shao. Linear model selection by cross-validation. *Journal of the American Statistical Association*, 88:486–494, 1993.

[134] Matthew Stephens. Bayesian analysis of mixtures with an unknown number of components — an alternative to reversible jump methods. *Annals of Statistics*, 28:40–74, 2000.

[135] Matthew Stephens. Dealing with label-switching in mixture models. *Journal of the Royal Statistical Society, Series B*, 62:795–809, 2000.

[136] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36:111–147, 1974.

[137] M. Strickert, S. Teichmann, N. Sreenivasulu, and U. Seiffert. 'DiPPP' online self-improving linear map for distance-preserving data analysis. In *Proceedings of the 5th International Workshop on Self-Organizing Maps, WSOM 2005*, pages 661–668, Paris, France, 2005.

[138] Marc Strickert, Nese Sreenivasulu, and Udo Seiffert. Sanger-driven MDSLocalize - a comparative stufy for genomic data. In *Proceedings of the 14th European Symposium on Artificial Neural Networks*, pages 265–270, 2006.

[139] Yee Whye Teh and Sam Roweis. Automatic aligment of local representations. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

[140] Eduardo Tejada, Rosane Minghim, and Luis Gustavo Nonato. On improved projection techniques to support visual exploration of multi-dimensional data. *Information Visualization*, 2:218–231, 2003.

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

[142] Neil H. Timm. *Applied Multivariate Analysis*. Springer Texts in Statistics. Springer-Verlag, New York, 2002.

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

[144] Lorenzo Torresani and Kuang chih Lee. Large Margin Component Analysis. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

[145] Edward Rolf Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Connecticut, 1983.

[146] A. Ultsch. Self-organization neural networks for visualisation and classification. In O. Opitz, B. Lausen, and R. Klar, editors, *Information and Classification*, pages 307–313. Springer-Verlag, Berlin, 1993.

[147] Jarke J. van Wijk. Views on visualization. *IEEE Transactions on visualization and computer graphics*, 12:421–432, 2006.

[148] Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

[149] 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*, pages 297–304, Cambridge, MA, 2004. MIT Press.

[150] J.J. Verbeek, N. Vlassis, and B. Kröse. Fast nonlinear dimensionality reduction with topology preserving networks. In *Proceedings of 10th European Symposium on Artificial Neural Netorks (ESANN 2002)*, pages 193–198, 2002.

[151] T. Vilmann, R. Der, M. Herrmann, and T.M. Martinetz. Topology preservation in self-organizing feature maps: exact definition and measurement. *IEEE Transactions on Neural Networks*, 8:256–266, 1997.

[152] Michail Vlachos, Carlotta Domeniconi, Dimitrios Gunopulos, George Kollios, and Nick Koudas. Non-linear dimensionality reduction techniques for classification and visualization. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 645–651, New York, NY, USA, 2002. ACM Press.

[153] Gang Wang, Hui Zhang, Zhihua Zhang, and Frederick H. Lochovsky. A bernoulli relational model for nonlinear embedding. In *ICDM '05: Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 458–465, Washington, DC, USA, 2005. IEEE Computer Society.

[154] Jing Wang and Zhenyue Zhang. MLLE: Modified Locally Linear Embedding using multiple weights. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

[155] Jing Wang, Zhenyue Zhang, and Hongyuan Zha. Adaptive manifold learning. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1473–1480, Cambridge, MA, 2005. MIT Press.

[156] Colin Ware. *Information Visualization: Perception for design.* Elsevier, 2 edition, 2004.

[157] Colin Ware, Helen Purchase, Linda Colpoys, and Matthew McGill. Cognitive measurements of graph aesthetics. *Information Visualization*, 1:103–110, Jun 2002.

[158] Edward J. Wegman and Qiang Luo. On methods of computer graphics for visualizing densities. *Journal of Computational and Graphical Statistics*, 11:137–162, 2002.

[159] Kilian Weinberger, Benjamin Packer, and Lawrence Saul. Nonlinear dimensionality reduction by semidefinite programming and kernel matrix factorization. In Robert G. Cowell and Zoubin Ghahramani, editors, *Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS 2005)*, pages 381–388. Society for Artificial Intelligence and Statistics, 2005.

[160] Kilian Weinberger, Fei Sha, Qihui Zhu, and Lawrence Saul. Graph regularization for Maximum Variance Unfolding with an application to sensor localization. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

[161] Kilian Q. Weinberger and Lawrence K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, 70:77–90, 2006.

[162] Kilian Q. Weinberger, Fei Sha, and Lawrence K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, pages 839–846, New York, NY, USA, 2004. ACM Press.

[163] Li Yang. Distance-Preserving Projection of high-dimensinal data for nonlinear dimensionality reduction. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 26:1243–1246, 2004.

[164] Li Yang. Sammon's nonlinear mapping using geodesic distances. In *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2*, pages 303–306, Washington, DC, USA, 2004. IEEE Computer Society.

[165] Li Yang. Building connected neighborhood graphs for isometric data embedding. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 722–728, New York, NY, USA, 2005. ACM Press.

[166] Li Yang. Building k edge-disjoint spanning trees of minimum total length for isometric data embedding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1680–1683, 2005.

[167] Julie Yang-Peláez and Woodie C. Flowers. Information content measures of visual displays. In *Proceedings of the IEEE Symposium on Information Vizualization 2000, INFOVIS '00*, pages 99–103, Washington, DC, USA, 2000. IEEE Computer Society.

[168] Bin Yu and Per Mykland. Looking at Markov samplers through cusum path plots: a simple diagnostic idea. *Statistics and Computing*, 8:275–286, 1998.

[169] Hongyuan Zha and Zhenyue Zhang. Isometric embedding and continuum isomap. In *Proceedigns of the International Conference on Machine Learning (ICML-2003)*, pages 864–871, 2003.

[170] Jiajie Zhang. A representational analysis of relational information displays. *International Journal of Human-Computer Studies*, 45:59–74, 1996.

[171] Zhenyue Zhang and Hongyuan Zha. Principal manifolds and nonlinear dimensionality reduction via Tangent Space Alignment. *SIAM Journal on Scientific Computation*, 26:313–338, 2004.

[172] Huaiyu Zhu and Richard Rohwer. Information geometric measurement of generalization. Technical Report NCRG/4350, Neural Computing Research Group, Aston University, 1995.

[173] Alexander Zien and Joaquin Qui nonero Candela. Large margin non-linear embedding. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 1060–1067, New York, NY, USA, 2005. ACM Press.

[174] Joseph L. Zinnes and David B. MacKay. Probabilistic multidimensional scaling: complete and incomplete data. *Psychometrica*, 48:27–48, 1983.