Exploring model selection techniques for nonlinear dimensionality reduction

Stefan Harmeling
Edinburgh University, Scotland
stefan.harmeling@ed.ac.uk

Informatics Research Report EDI-INF-RR-0960

SCHOOL *of* INFORMATICS
Institute of Adaptive and Neural Computation

March 2007

Abstract : Nonlinear dimensionality reduction (NLDR) methods have become useful tools for practitioners who are faced with the analysis of high-dimensional data. Of course, not all NLDR methods are equally applicable to a particular dataset at hand. Thus it would be useful to come up with model selection criteria that help to choose among different NLDR algorithms. This paper explores various approaches to this problem and evaluates them on controlled data sets. Comprehensive experiments will show that model selection scores based on stability are not useful, while scores based on Gaussian processes are helpful for the NLDR problem.

Keywords : Nonlinear dimensionality reduction, bootstrap, GPLVM

1 Introduction

High-dimensional data that is contained in some lower-dimensional manifold can be embedded into much lower dimensions by nonlinear dimensionality reduction (NLDR) methods (see [18, 15, 2, 16]). Suppose the data is given as vectors $y_1, \ldots, y_n \in \mathbb{R}^D$ in some high-dimensional space $\mathbb{R}^D$, but actually lies on some much lower, say $d \ll D$, dimensional manifold. Then the task of NLDR is to find d-dimensional vectors $x_1, \ldots, x_n \in \mathbb{R}^d$ such that the entries of x_i are the coordinates along that (possibly quite curvy) manifold. Of course, this is not always possible. Usually it is necessary that the manifold is isomorphic to $\mathbb{R}^d$.

Several methods exist for NLDR, each pursuing a different paradigm, which is often motivated by the NLDR problem but also influenced by computational concerns. Since NLDR is an example of unsupervised learning, the different methods always will give some results when applied to a particular dataset. The difficult task for the practitioner is to judge the different solutions to decide from which solution she can learn something about her dataset. This paper explores and tests model selection techniques for this problem which should help practitioner to assess their solutions. By the end of this paper we will have experimentally shown that:

1. A stability-based approach to model selection for NLDR fails.
2. A Bayesian approach based on Gaussian process latent variable models (GPLVMs) can rank NLDR solution in a meaningful way, that corresponds to some ground truth.

Towards these goals we proceed as follows: after briefly discussing model selection for NLDR in Section 2 we describe in Section 3 situations in which NLDR might fail and in Section 4 how to compare different embeddings. Then we explain in Section 5 stability-based approaches and in Section 6 a Bayesian approach to compare the performance of different NLDR methods. After that in Section 7 we experimentally evaluate those approaches. Finally, we summarize our results in Sec. 8.

2 Model selection for NLDR

In supervised learning problems (like the classification task), we can compare different classification methods by the test error they achieve, when applied to test data. The best method for the problem at hand is the classification procedure that has reached the smallest test error. Such kind of model selection is more difficult for unsupervised learning problems (like clustering and NLDR). The reason is that the goal is usually more *qualitatively* described. For instance for clustering we want a method to find the clusters which somehow group the data points. This is interpretable in many ways. Are we looking for large clusters? Or for clusters of similar size? Are we looking for exactly five clusters? Are we looking for a hierarchy of clusters? Similarly for the NLDR problem, in which the informal goal is to unfold the underlying manifold. Such an imprecise description has the advantage that it opens up many possibilities to approach the problem. On the other hand it makes it quite difficult to compare solutions from different algorithms, because it is quite hard in real-world situations to *objectively* measure whether one embedding is more meaningful than another. Looking at a single NLDR algorithm it is usually possible to compare different solutions (for instance for hyperparameter learning), since a single algorithm optimizes a single fixed cost function. Such a cost function is motivated by the NLDR goal, but also by computational concerns. The solution of a particular NLDR algorithm is then of course favored by its own cost function, but probably not by the cost functions of other NLDR methods. For instance ISOMAP looks for an embedding that preserves all pair-wise geodesic distances along the manifold, whereas LLE seeks an embedding that preserves the local neighborhoods. Each of these methods can measure how well its individual criterion is met, but it is not straightforward to compare the results of different methods, since they try to optimize different things and their cost functions are usually on different scales as well.

Thus in order to implement a useful model selection procedure we need to define *neutral* scores that rank different embeddings for a given dataset obtained from different algorithms. Of course

such scores will not be *objective* either. Similar to the algorithmic decisions, the various NLDR method make in choosing their cost functions, such a neutral score will only capture particular aspects of possible embeddings and will thus be *subjective* as well. However, possibly the choice will be more guided by the abstract description of the NLDR problem and by our views about what embeddings we are looking for and not by computational concerns like whether it can be calculated by an eigenvalue problem. To this end, this paper will describe and test two classes of *subjective* score functions:

Stability-based scores: the intuition behind stability is that, if an embedding reflects the underlying statistical properties of a dataset it should be a robust solution. That is a slight perturbation of the dataset should not change the embedding very much. On the other hand, not-so-meaningful embeddings should be unstable. We will see in the experiments that this criterion is ineffective.

Bayesian scores: one possibility to be subjective about NLDR is to formulate our initial beliefs about possible embeddings as a probability distribution over possible mappings. Such a distribution allows us to compare different solution by asking, which embedding is more likely given our prior beliefs. As we will see in the experiments this approach is useful.

In order to evaluate these score functions, we will design several controlled NLDR datasets in which LLE and/or ISOMAP have or do not have problems to find good embeddings. Note that in this paper we will focus on LLE and ISOMAP, because both methods are easily implemented and they are fast, which makes them particularly useful for comprehensive evaluations.

3 When does NLDR fail?

Even though NLDR methods are quite successful in obtaining interesting embeddings (for instance [18, 15, 2, 16]), there are situations in which common NLDR methods fail. These can have different reasons which we discuss next.

3.1 Proximity-graph related problems

Many NLDR methods, including LLE and ISOMAP, are based on proximity graphs: usually the k-nearest-neighbor graph is employed, the vertices of which are the given data points and the edges connect a data point to its k nearest neighbors. Alternatively, ϵ-ball graphs can be used, the edges of which connect a data point to its neighbors which are not further than ϵ away. The idea is that under the assumption that the data points originate from a low dimensional manifold, both graphs (as do proximity graphs in general) have the property that their edges only go along the underlying manifold for appropriately chosen hyperparameters (here k or ϵ) and for densely enough sampled data. Since the proximity graph is central to LLE and ISOMAP, their results can be easily become corrupted if the employed graph does not go along the underlying manifold, which can happen because of:

- noise,
- sparse sampling.

In Section 7 we will test our model selection scores on such cases.

3.2 Model mismatch

The second group of failures is due to model mismatch. Model mismatch appears if a dataset does not fulfill the (implicitly determined) model of the applied algorithm, in our case of LLE and/or ISOMAP. Proximity-graph related issues can be also interpreted as model mismatches, but there are further reasons why a model mismatch can occur:

- the embedding dimension is chosen too small,
- the data is connected but not convex (a problem for ISOMAP),
- the manifold is non-uniformly sampled (a problem for LLE),
- or the manifold is not flat (the fishbowl problem).

These lists of failures are of course not exhaustive and for other NLDR methods we will be able to identify other failures. We will use these cases to evaluate our model selection scores.

4 Comparing two embeddings

The observed data points $y_1, y_2, \ldots, y_n \in \mathbb{R}^D$ originate from some high dimensional space (with large D) which we call *data space*. For notational brevity we collect those vectors as columns in the $D \times n$ matrix $Y = [y_1, y_2, \ldots, y_n]$. Given Y an NLDR algorithm calculates the lower dimensionally embedded vectors $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ for some $d \ll D$. Similar to Y we collect those vectors in an $d \times n$ matrix X. We call $\mathbb{R}^d$ the *embedding space* or *latent space*. Suppose we obtain another embedding Z. How can we measure how similar they are? We do this in two steps: (i) first we *whiten* the embedded data and then (ii) we calculate some squared distance using the *Procrustes analysis*. Both parts will be detailed in the next two subsections.

4.1 Whitening

First of all we assume that X has been already transformed such that the mean of the columns of X is zero. Additionally whitening a dataset X, forces its covariance to be the identity matrix. Whitening has its name from the fact that the covariance of white noise is also the identity matrix. Let us write the estimated covariance matrix in terms of X,

$$C = XX^\top/n. \tag{1}$$

Let V be the matrix of C's eigenvectors (as column vectors in V) and let Λ be diagonal matrix of C's eigenvalues (along the diagonal), then we have

$$CV = V\Lambda. \tag{2}$$

We whiten X by transforming it with V and Λ,

$$X_w = \Lambda^{-1/2}V^\top X, \tag{3}$$

which can be verified by observing that X_w's (estimated) covariance matrix,

$$X_w X_w^\top/n = \Lambda^{-1/2}V^\top XX^\top V\Lambda^{-1/2}/n = I, \tag{4}$$

is the identity matrix (using the previous Equations and $V^\top V = I$).

4.2 Procrustes analysis

Let us assume that X and Z have already been whitened. Then we apply Procrustes analysis (see for instance [3]) which transforms X such that the sum of the distances between the points (that is the columns) in X and Z,

$$\bar{\rho}(X, Z) = \sum_{i=1}^{n} (z_i - t(x_i))^\top (z_i - t(x_i)). \tag{5}$$

is minimized. Hereby the transformation $t : \mathbb{R}^d \to \mathbb{R}^d$ has the form

$$t(x) = \alpha Ax + b \tag{6}$$

with $\alpha \in \mathbb{R}$ being a scaling factor, A being a rotation matrix and b being an translation vector. The *Procrustes distance* of X and Z is the minimal sum of distances among all possible transformation,

$$\rho(X, Z) = \min_{\alpha, A, b} \bar{\rho}(X, Z). \tag{7}$$

Note that because $t(x)$ is an invertible transformation, ρ is symmetric in X and Z. The reason for initial whitening is that Procrustes analysis allows only to adjust the overall scaling of the two datasets. However, in the NLDR problem we additionally want to ignore the scaling in each of the different embedding dimensions.

5 Stability-based model selection

Intuitively speaking, stability-based model selection scores measure how much a result would change if we had slightly different data. This concept was successfully applied to independent component analysis (ICA, for an overview see [7]). For ICA, the question was whether calculated independent components reflect statistical properties of the given dataset or whether they are due to random variation in the data. [11, 12, 10, 6] were able to show that the most stable components are usually the most relevant ones. Solutions that were purely random or due to over-fitting could be identified. Viewing an obtained ICA solution as a (latent) model of the original data, stability analysis was thus able to perform a certain form of model selection. Both ICA and NLDR are unsupervised learning methods for which model selection in general is difficult. Motivated by the success of stability-based model selection in ICA, we will try in the following to apply stability-based measures to NLDR problems.

Stability analysis in a nutshell can be described as follows: In practical problem we only have a single dataset Y. Thus it is not directly possible to estimate the stability of the solution that some algorithm would give us. However, if we had access to the distribution from which Y originates, we could draw more datasets $Y^{(1)}, Y^{(2)}, \ldots, Y^{(R)}$ and apply the chosen algorithm to each of those samples to obtain several solutions, which we could analyse how similar they are to each other (which we call *scatter*) and to the solution obtained from Y (which we call *offset*). Bootstrap (BT) and noise injection (NI) are two approaches which generate datasets that share certain properties of the given dataset Y.

5.1 Bootstrap

Bootstrap (BT, see [5]) approaches the problem of generating more datasets from the point of view of the distribution of the column vectors, that is of the data points themselves. It takes the empirical distribution given by the set of column vectors in Y and repeatedly samples with replacement from the columns of Y to generate surrogate datasets $Y^{(1)}, Y^{(2)}, \ldots, Y^{(R)}$ of the same size as Y. Of course the empirical distribution as expressed by the columns of Y is a discrete distribution and thus only a coarse approximation to the true distribution of the data points. Furthermore, note that in the surrogate datasets many data points (column vectors) will appear several times.

Using those additional datasets, we can define two scores which measure different aspects of the stability. For notational convenience we denote the currently considered NLDR algorithm by ϕ and view ϕ as a mapping from data space to embedding space, that is the embedded vectors can be written as $X = \phi(Y)$.

1. Calculate the average Procrustes distance of the embeddings of the surrogate datasets to the initial solution:

$$\text{offset} = \frac{1}{R} \sum_{r=1}^{R} \rho(\phi(Y), \phi(Y^{(r)})) \tag{8}$$

2. Additionally, we can calculate pairwise Procrustes distances of the embeddings of the surrogate datasets:

$$\text{scatter} = \frac{1}{R^2} \sum_{r=1}^{R} \sum_{s=1}^{R} \rho(\phi(Y^{(r)}), \phi(Y^{(s)})) \tag{9}$$

Note that "scatter" can be small while "offset" is large, if the solutions for the resampled datasets are very different from Y but very stable. On the other hand, if "scatter" is large, then "offset" will also be large. Informally speaking, "offset" is related to bias and "scatter" to variance.

5.2 Noise injection

An alternative to BT sampling is noise injection (NI, see [6]) which differs from BT in the way the surrogate datasets are generated: instead of resampling the original dataset Y, we obtain surrogate datasets $Y^{(1)}, Y^{(2)}, \ldots, Y^{(R)}$ by adding small amounts of white noise to Y:

$$Y^{(r)} = \sqrt{1 - \sigma^2}\, Y + \sqrt{\sigma^2}\, N^{(r)} \tag{10}$$

with each entry of $N^{(r)}$ being standard normally distributed for all r, and σ^2 being a parameter between zero and one that determines the mixture of true data and noise. Thus instead of using the empirical distribution, NI employs a kernel density estimate of the distribution of data vectors. However, instead of sampling from the overall kernel density estimate (which would be possible as well) in this paper we sample from each Gaussian distribution (centred at each data point) exactly once. Another point of view is that the original dataset Y is shaken a little bit to get a similar but slightly different one. Using again Equations (8) and (9) this induces two NI scores, "offset" and "scatter".

6 Bayesian model selection

Recognising that model selection for unsupervised learning problems is necessarily *subjective*, we suggest to formalize our beliefs about the embedding. A reasonable assumption is that the embedding should be smooth and invertible. Thus let us assume that the mapping from the (low dimensional) embedded vectors to the given (high dimensional) data vectors is smooth (see also [1]). That is, we formulate a prior distribution over the mappings from the low dimensional space to the high dimensional space. The reason for not choosing to model the mapping from data space to embedding space is that there are smooth mappings from data space to embedding space which "fold" the manifold which will result in (an unwanted) loss of information. Having specified a distribution over inverse embeddings, we can compare the likelihoods of different solutions, hereby also taking into account the values of possible hyperparameters. In the following we will formulate such priors using Gaussian processes.

Gaussian processes have been used extensively in machine learning (see [14]). They provide a principled way to state prior beliefs about functions. In the following we introduce Gaussian processes and explain how we can use them to formulate model selection scores for NLDR.

6.1 Gaussian processes

A Gaussian process can be seen as a set of random variables indexed by real numbers (that is as a stochastic process), such that each finite subset of those random variables is distributed according to a multivariate Gaussian distribution. One instance of a Gaussian process (that is one instance of the set of random variables) can be seen as a function $f : \mathbb{R} \to \mathbb{R}$ from the real numbers to the real numbers. Viewing the function f as a random variable itself (as a *random function* to be specific), a Gaussian process defines a distribution over functions. Allowing multidimensional indices, we can similarly define Gaussian processes over real-valued functions of vectors:

Definition 1 *The random function $f : \mathbb{R}^d \to \mathbb{R}$ is distributed according to a Gaussian process with mean function $m : \mathbb{R}^d \to \mathbb{R}$ and covariance function $k : (\mathbb{R}^d \times \mathbb{R}^d) \to \mathbb{R}$, abbreviated*

$$f \sim GP(m, k), \tag{11}$$

if and only if for each finite set of vectors $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ the random vector $(f(x_1), f(x_2), \ldots, f(x_n))^\top$ is distributed according to a multivariate Gaussian distribution with n dimensional mean vector μ with entries

$$\mu_i = m(x_i) \tag{12}$$

and $n \times n$ dimensional covariance matrix Σ with entries

$$\Sigma_{ij} = k(x_i, x_j). \tag{13}$$

The parameters of a Gaussian process are the mean function m and the covariance function k. Note that the vector $(f(x_1), f(x_2), \ldots, f(x_n))^\top$ is random due to the randomness of f.

6.2 Multidimensional Gaussian processes

So far, we considered real-valued functions of vectors. However, the inverse mapping of the NLDR embedding is vector-valued. We can generalize the previous definition to vector-valued functions $f : \mathbb{R}^d \to \mathbb{R}^D$ by viewing f as a vector of real-valued functions of vectors, that is $f = (f_1, f_2, \ldots, f_D)^\top$ with $f_I : \mathbb{R}^d \to \mathbb{R}$ for all I. Similarly view $m : \mathbb{R}^d \to \mathbb{R}^D$ as $m = (m_1, m_2, \ldots, m_D)^\top$ and $k : (\mathbb{R}^d \times \mathbb{R}^d) \to \mathbb{R}^D$ as $k = (k_1, k_2, \ldots, k_D)^\top$. Note that the upper case indices I and J run over dimensions in $\mathbb{R}^D$ and lower case indices i and j run over a finite set of data points in $\mathbb{R}^d$.

Definition 2 *The vector-valued random function $f : \mathbb{R}^d \to \mathbb{R}^D$ is distributed according to a Gaussian process with mean function $m : \mathbb{R}^d \to \mathbb{R}^D$ and covariance function $k : (\mathbb{R}^d \times \mathbb{R}^d) \to \mathbb{R}^D$, abbreviated*

$$f \sim GP(m, k), \tag{14}$$

if and only if for all $I \in \{1, 2, \ldots, D\}$

$$f_I \sim GP(m_I, k_I). \tag{15}$$

Note that the random functions $f_1, f_2, \ldots, f_D$ are independent given m and k. This is not the most general form of a multidimensional Gaussian process: here we employed a vector-valued covariance function, but instead we could consider it to be matrix-valued, $k : (\mathbb{R}^d \times \mathbb{R}^d) \to \mathbb{R}^{D \times D}$. The latter form could take possible correlations between different dimensions in $\mathbb{R}^D$ into account. Definition 2 is a special case thereof assuming zero correlations between different dimensions. The more general form is beyond the scope of this paper and for our purposes Definition 2 is sufficient. We even simplify it further and will write $f \sim GP(m, k)$ for real-valued covariance function $k : (\mathbb{R}^d \times \mathbb{R}^d) \to \mathbb{R}$. In that case for each dimension I we will use the same covariance function k.

6.3 Gaussian process latent variable model

The Gaussian process approach to the NLDR problem introduces the *Gaussian process latent variable model* (GPLVM, see [8]) which is a latent variable model for the observed data points $y_1, y_2, \ldots, y_n \in \mathbb{R}^D$ using Gaussian processes: the GPLVM assumes a Gaussian process prior for the mapping $f : \mathbb{R}^d \to \mathbb{R}^D$ from the low dimensional space $\mathbb{R}^d$ (the latent space) to the high dimensional space $\mathbb{R}^D$ (the data space):

$$f \sim GP(0, k) \tag{16}$$

Note that we fix the mean function m to be zero everywhere which is all right if nothing more is known. The covariance function k can be freely chosen as well. In this paper we choose the weighted sum of the squared exponential covariance function and white noise covariance function,

$$k(x_i, x_j) = \theta_{\text{rbf}} \exp(-\frac{(x_i - x_j)^\top (x_i - x_j)}{2\sigma^2}) + \theta_{\text{noise}} \delta_{ij}, \tag{17}$$

for data points $x_i, x_j \in \mathbb{R}^d$ with $\delta_{ij} = 1$ for $i = j$ and zero otherwise, and $\theta_{\text{rbf}} > 0$ and $\theta_{\text{noise}} > 0$ being the signal and noise variance, and $\sigma > 0$ being the kernel width. The first term, the squared exponential covariance function, captures the following intuition: suppose a and b are close in the embedding space $\mathbb{R}^d$. Then $\exp(\ldots)$ will be large and with it $k(x_i, x_j)$ will be large. Hence their images $f(x_i)$ and $f(x_j)$ will be correlated and thus close as well. We will assume k to fixed and not a random function. We collect all those hyperparameters of the covariance function in the variable $\theta = (\sigma, \theta_{\text{rbf}}, \theta_{\text{noise}})$. Since we share a single covariance function k for all dimensions I, also the hyperparameters θ are shared among all dimensions I.

Having defined a prior for f, we generate a data point y_i given some latent point x_i simply by mapping x_i via f,

$$y_i = f(x_i). \tag{18}$$

Notation-wise we collect (again) all observed data points in a $D \times n$ matrix $Y = [y_1, y_2, \ldots, y_n]$, similarly for the latent data points, $X = [x_1, x_2, \ldots, x_n]$ (in the embedding space). Now we want an expression for how likely it is to observe Y given X under the Gaussian process prior for f. By the defining property of the Gaussian process each row Y_I of Y is distributed according to a Gaussian distribution and since the rows are independent we obtain a product of those distributions,

$$p(Y|X, \theta) = \prod_{I=1}^{D} N(Y_I; 0, K) \tag{19}$$

with K being the covariance matrix[1] with entries

$$K_{ij} = k(x_i, x_j). \tag{20}$$

Note that K depends on the hyperparameter θ. Viewing also the matrix X as the parameter to optimize we can call $p(Y|X, \theta)$ the likelihood of X and θ. Taking logs we obtain the log-likelihood,

$$\log p(Y|X, \theta) = -\frac{Dn}{2} \log 2\pi - \frac{D}{2} \log |K| - \frac{1}{2} \operatorname{tr}(K^{-1} Y Y^\top) \tag{21}$$

which is identical to Equation (6) in [8]. Probabilistic nonlinear PCA which is introduced in [8] directly maximizes $\log p(Y|X, \theta)$ with respect to X in order to find an embedding of Y. In that manner GPLVMs are used for nonlinear dimensionality reduction. However, $\log p(Y|X, \theta)$ is prone to many local minima, so direct minimization is often not straightforward (see also [9] for recent work on this). Slightly differently, we propose in this paper to use GPLVMs to evaluate the performance of LLE and ISOMAP.

6.4 GPLVMs as a model selection criterion for NLDR

In order to compare the results of different algorithms we need to agree on a criterion that somewhat captures our (informal) notion of what a good embedding is. We suggest that a good embedding leads to a smooth mapping from the embedded data points to the observed points. Thus we will assume a Gaussian process prior with a squared exponential covariance function (and a noise term) for that mapping. Note that as discussed in [9] a GPLVM makes sure that far-away points in data space are far away in latent space, which corresponds to the intuition that the embedding should not fold parts of the manifold together.

[1] In the "kernel-methods" world K is called the kernel matrix (e.g. [17, 13]).

Suppose we are observing some data which we (as before) represent as column vectors of the $D \times n$ matrix Y. An NLDR algorithm calculates a low dimensional embedding which consists of a $d \times n$ matrix X of column vectors. Then we can define the GPLVM scores as follows:

Definition 3 *The GPLVM scores calculate for the fixed covariance function k in Equation (17) that depends the hyperparameter $\theta = (\sigma, \theta_{rbf}, \theta_{noise})$ (i) how likely it is that Y has been generated from X by some random function $f \sim GP(0, k)$, that is the log-likelihood,*

$$\text{lml} = \max_{\sigma, \theta_{rbf}, \theta_{noise}} \log p(Y|X, \theta), \tag{22}$$

and (ii) the corresponding optimal values of the hyperparameters,

$$(\text{width, sigvar, noivar}) = \arg \max_{\sigma, \theta_{rbf}, \theta_{noise}} \log p(Y|X, \theta) \tag{23}$$

Before calculating these values we whiten X and Y. This is important because otherwise the hyperparameters for different runs would not be comparable. Note that while the hyperparameters change with linear transformations of X and Y, the log-likelihood $\log p(Y|X, \theta)$ does not.

In order to calculate these quantities, the hyperparameters are optimized using standard conjugate gradient descent algorithm (as implemented by Carl Rasmussen's Matlab function minimize.m, which is part of the Gaussian process package available at http://www.gaussianprocess.org/gpml/code/matlab/doc/). In principle, of course, this optimization might run into local minima. However, our experience during the experiments described in the following was that various randomly initialized runs lead to the same results.

By now we defined various scores which we will analyse further in the following. Of course we could use alternative covariance functions with alternative hyperparameter dependent on the problem we look at.

7 Experiments

To find out which of these scores are useful for model selection in NLDR problem, we applied them to various (controlled) examples that correspond to the cases (discussed in Section 3) in which NLDR methods often fail.

For artificial datasets like the one we use in the following, we often know from the data generation process the "true" embedding, which we denote by the $d \times n$ matrix $Z = [z_1, z_2, \ldots, z_n]$. The word "true" is in quotation marks, since that is only one possible embedding and we could produce a different one for instance by monotonically transforming each coordinate. However, we tried to choose a more or less *canonical* embedding that best matches the one we would expect.

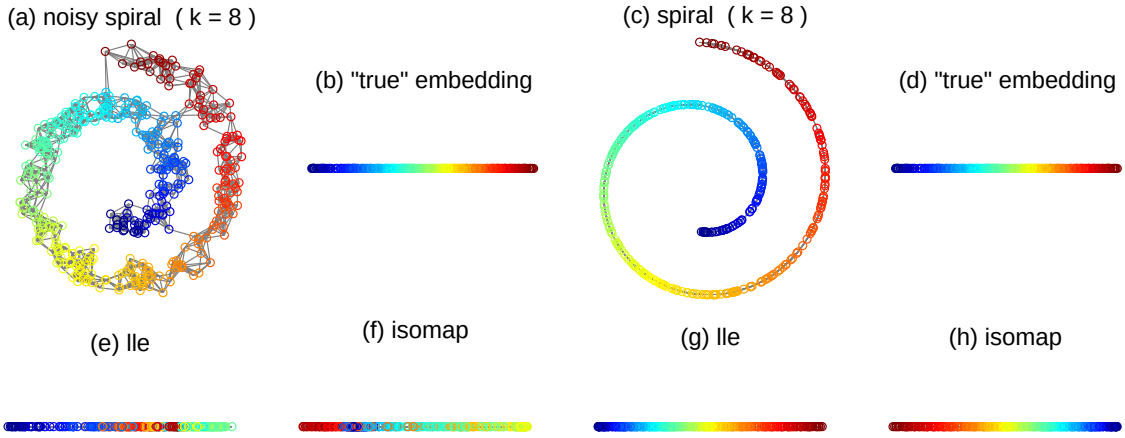
Since we do try to get the best possible results for the various algorithms, we did not search for the best value k (number of neighbors in the proximity graph). Also we did not apply all available algorithms to all problems, which would have been computationally demanding for some of the indices. Instead we focused on using LLE and ISOMAP, which run reasonably fast on the problems we looked at.

7.1 Proximity graph related issues

The first two experiments are constructed to contrast a dataset with a proximity graph that does not go along the manifold, against an example with a correct proximity graph. Then we can compare our qualitative expectations with the scores we calculate.

7.1.1 Noise

The first way to corrupt the proximity graph is by adding noise to the dataset. In the first row of plots in Figure 1 we show from left to right the noisy spiral dataset (with 400 points), its "true"



		(a/b) noisy spiral		(c/d) spiral	
		(e) lle	(f) isomap	(g) lle	(h) isomap
procrustes	dist	0.76914	0.99662	0.016086	0.017243
gplvm	nlml	890.0938	880.4318	-6198.4008	-7263.0611
	width	0.25662	0.34146	0.10483	0.38439
	sigvar	0.61935	0.69599	0.5841	0.84122
	noivar	0.70246	0.69962	2.2109e-05	1.5671e-05
bootstrap 10	scatter	0.77297	3.8636e-16	0.76226	8.7486e-16
	offset	0.73397	3.3307e-16	0.62785	9.1038e-16
bootstrap 50	scatter	0.89553	4.3059e-16	0.77991	8.1766e-16
	offset	0.85861	5.5511e-16	0.5902	8.7486e-16
noise inj 10 0.3	scatter	0.77199	4.3521e-16	0.54319	7.3275e-16
	offset	0.81871	3.3307e-16	0.513	1.1102e-15
noise inj 10 0.6	scatter	0.84088	3.1974e-16	0.667	8.2157e-16
	offset	0.79698	9.992e-16	0.54336	9.1038e-16

Figure 1: (a) Noisy spiral data with (b) its "true embedding" and (e) LLE's and (f) ISOMAP's embeddings, (c) clean spiral data with (c) its "true embedding" and (g) LLE's and (h) ISOMAP's embeddings. The table shows all scores for these datasets.

embedding, and the clean spiral dataset (with 400 points) with its "true" embedding. Note that the graph (for $k = 8$) of the noisy spiral has several shortcuts which are due to the noise, while the clean spiral has none. Those shortcuts are the reason why the embeddings of LLE and ISOMAP (both with $k = 8$) are not perfect (the two left-most plots in the second row), while on the clean dataset the embeddings are perfect (the two right-most plots in the second row). These findings by visual inspection are matched by the Procrustes distance that compares the embeddings found by the algorithms to the respectively "true" embeddings. However, the calculation of the Procrustes distance requires access to the "true" embeddings, which are of course not available in real-world datasets. Thus let us see which of the indices best match to our visual results and the Procrustes distance: the log-likelihood of the GPLVM model and the noise-variance are large for the noisy spiral and very small for the clean spiral which is in correspondence to the Procrustes-score. However, note that for both the clean and noisy spiral we have

$$\text{lml(LLE)} > \text{lml(ISOMAP)} \tag{24}$$

$$\text{noivar(LLE)} > \text{noivar(ISOMAP)} \tag{25}$$

while

$$\text{procrustes(LLE)} < \text{procrustes(ISOMAP)}. \tag{26}$$

What about the stability-based scores? They all fail to capture the quality of the solution: we tried BT with 10 and 50 repetitions and NI with 10 repetitions and two levels of added noise (0.3 and 0.6). However, the scores for LLE are all quite large, while the values for ISOMAP are very small, indicating that ISOMAP is not very sensitive to slight variations of the dataset while LLE is. Thus all those scores are here not useful for model comparison, since they are affected by the different algorithms in different ways.

7.1.2 Sparse sampling

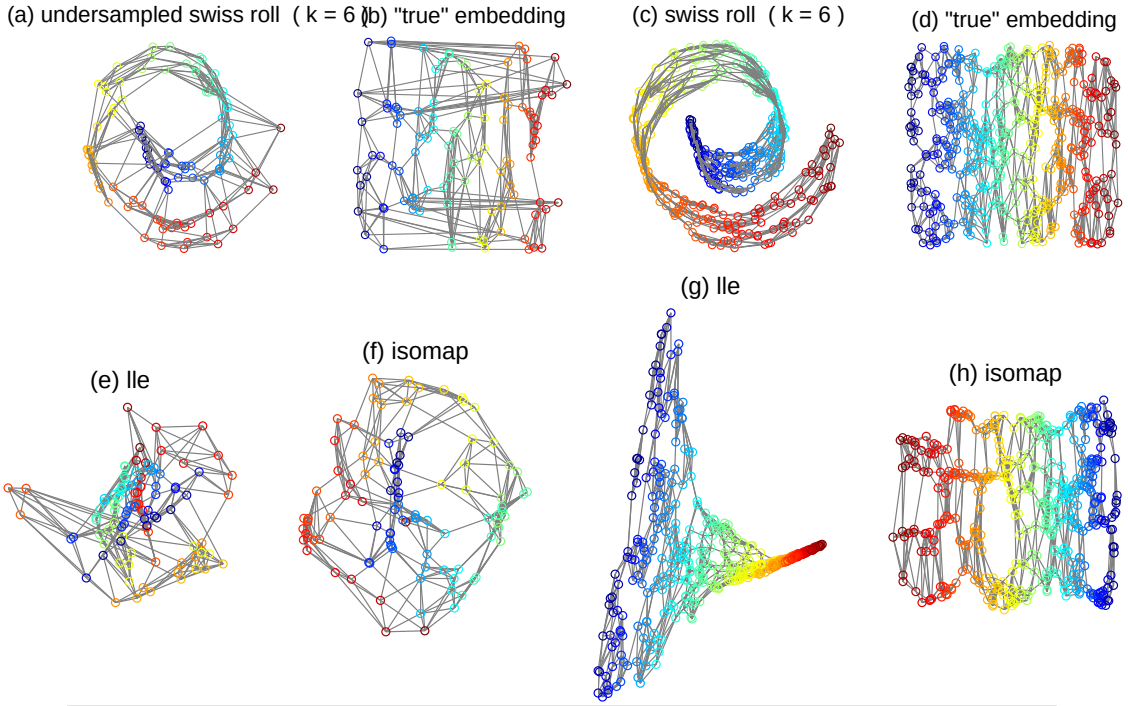
The second way to corrupt the proximity graph is by sparse sampling. Figure 2 shows the datasets and the results: the left dataset is an undersampled swiss roll with only 100 data points. This leads to a lot of shortcuts as can be seen in the upper left plot, indicating that the graph does not go along the manifold. The second dataset is sampled with 400 data points which does not have any shortcuts for $k = 6$. Of course, ISOMAP (with $k = 6$) fails on the first undersampled swiss roll, while it somewhat gets a reasonable embedding on the classic swiss roll. These findings are reflected by the Procrustes score, which requires knowledge of the "true" embedding, and also by the log-likelihood, the lml-score, and the noise-variance, the noivar-score. Again the methods based on resampling fail. Intuitively speaking, the ISOMAP embedding of undersampled swiss roll is quite wrong, but at the same time very stable.

7.2 Model mismatch

The second category of examples in which LLE and/or ISOMAP fail are due to model mismatch: in these situation we apply those NLDR methods to datasets for which they were not designed to work. Of course this is of practical relevance, since the practitioner does not know in an unsupervised learning problem, whether the model assumptions of the applied algorithms are fulfilled. Thus it is interesting to explore model selection scores for these situations as well.

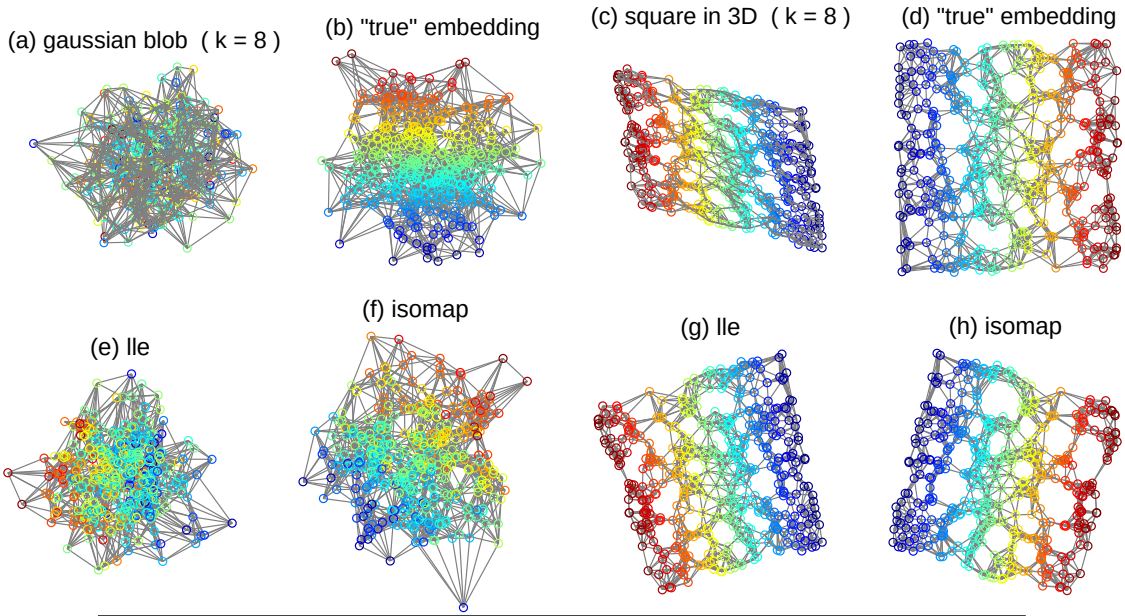
7.2.1 Embedding dimension too small

The simplest form of model mismatch in NLDR problem is a dataset in which there is no manifold to find. For this we created two datasets (both 400 data points, see Figure 3), (i) with a three dimensional Gaussian blob that does not have any manifold structure (left-most plots in the first row), and (ii) a uniformly sampled square that is embedded into three dimensions with some rotation (right-most plots in the first row). Of course, LLE and ISOMAP (both with $k = 8$)



		(a/b) undersampled swiss roll		(c/d) swiss roll	
		(e) lle	(f) isomap	(g) lle	(h) isomap
procrustes	dist	0.82196	0.75055	0.31727	0.030654
gplvm	nlml	287.893	264.9918	-512.666	-1220.4645
	width	0.49121	0.41378	0.42306	0.61248
	sigvar	0.95262	0.93925	3.9455	1.1221
	noivar	0.40877	0.33329	0.073515	0.057237
bootstrap 10	scatter	0.88044	2.6645e-17	0.85743	-2.2649e-16
	offset	0.97651	6.6613e-17	0.88859	-7.1054e-16
bootstrap 50	scatter	0.96214	7.9226e-17	0.94146	-1.144e-16
	offset	0.96383	3.9968e-17	0.86604	1.0658e-16
noise inj 10 0.3	scatter	0.80172	1.7764e-17	0.76175	-1.1102e-16
	offset	0.81203	6.6613e-17	0.72157	-3.9968e-16
noise inj 10 0.6	scatter	0.84919	8.8818e-17	0.8444	-2.2204e-16
	offset	0.89058	8.8818e-17	0.83021	-7.1054e-16

Figure 2: (a) Undersampled swiss roll data with (b) its "true embedding" and (e) LLE's and (f) ISOMAP's embeddings, (c) classic swiss roll data with (d) its "true embedding" and (g) LLE's and (h) ISOMAP's embeddings. The table shows all scores for these datasets.



		(a/b) gaussian blob		(c/d) square in 3D	
		(e) lle	(f) isomap	(g) lle	(h) isomap
procrustes	dist	0.60062	0.59199	0.0016892	0.0019481
gplvm	nlml	1081.6954	1094.3383	-5487.522	-2483.3359
	width	5.0058	15.077	1.2314	1.4083
	sigvar	2.3292	6.3859	0.86211	1.1214
	noivar	0.57605	0.58514	0.0014166	0.024187
bootstrap 10	scatter	0.89546	3.9524e-16	0.88688	3.8192e-16
	offset	0.9805	3.9968e-16	0.92139	7.9936e-16
bootstrap 50	scatter	0.97542	5.4587e-16	0.96297	6.7111e-16
	offset	0.98346	-4.2188e-16	0.91418	4.3521e-16
noise inj 10 0.3	scatter	0.77486	3.5971e-16	0.83788	4.0856e-16
	offset	0.69503	1.3323e-16	0.75126	1.1102e-16
noise inj 10 0.6	scatter	0.86647	2.3981e-16	0.88	4.5297e-16
	offset	0.85889	5.107e-16	0.88229	0

Figure 3: (a) Three-dimensional Gaussian blob data with (b) its "true embedding" and (e) LLE's and (f) ISOMAP's embeddings, (c) two-dimensional square data with (d) its "true embedding" and (g) LLE's and (h) ISOMAP's embeddings. The table shows all scores for these datasets.

fail on the Gaussian blob to find something reasonable, while at least the color coding from the "true" embedding is somewhat correct. However, still a two dimensional embedding of an three dimensional Gaussian blob has to squash one dimension which should be reflected in our scores. On the other hand, the square in three dimensions is easily found by LLE and ISOMAP (again both with $k = 8$). All those visual findings are nicely matched by the Procrustes score, the lml- and the noivar-score. In this example lml- and also noivar-score are also able to distinguish between the embeddings of the square in three dimensions obtained by LLE and ISOMAP. Different from the experiment in Section 7.1.1 now LLE gets a better lml- and a better noivar-score than ISOMAP. Again the stability-based scores are ineffective as in the previous experiments.

7.2.2 Connected but not convex

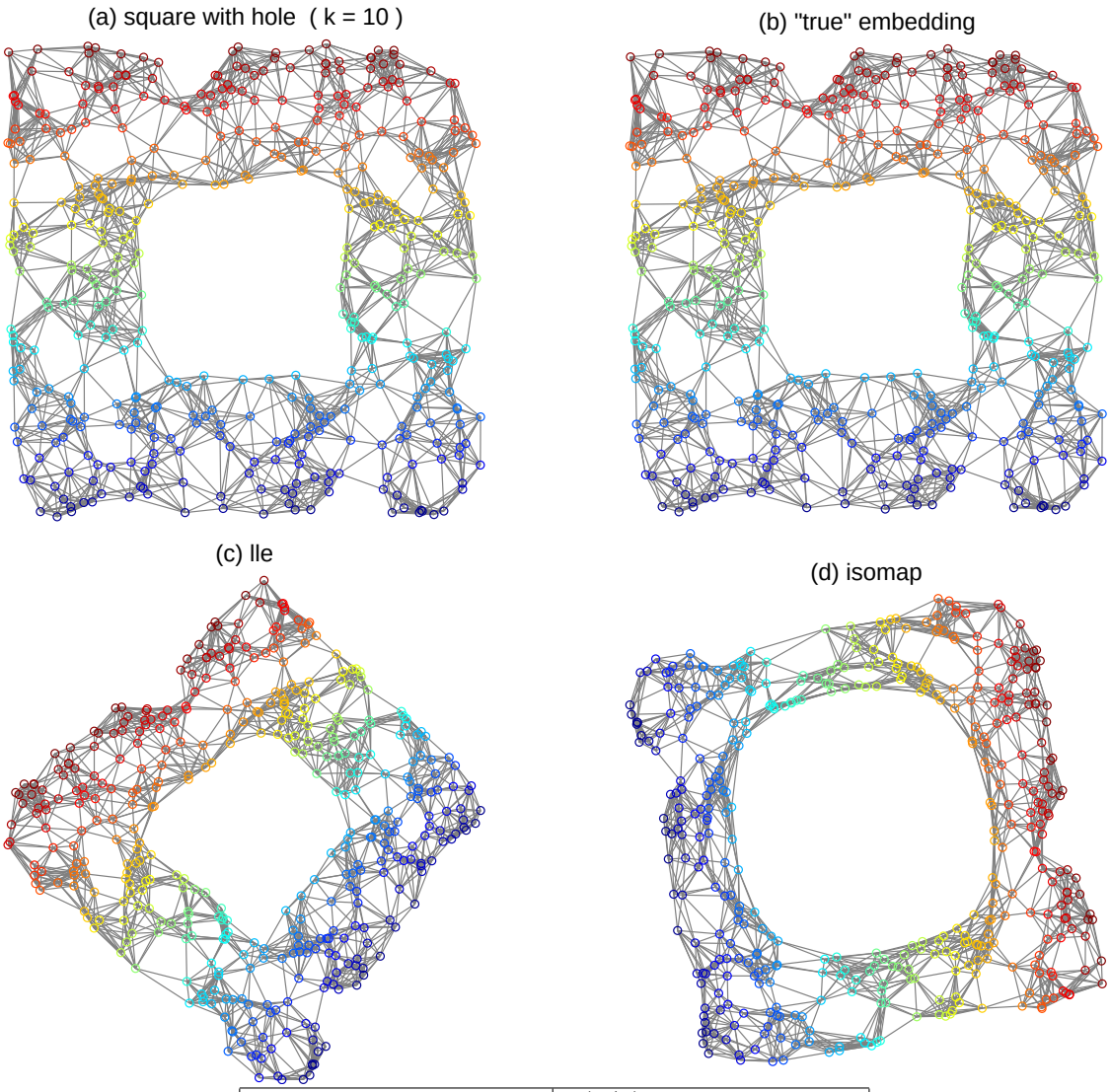
As pointed out by several authors (for instance [4]) ISOMAP requires the dataset to be connected and convex, that is the proximity graph should only have a single connected component and there should not be big holes in there. Otherwise, the theoretical guarantees of ISOMAP are not valid. To study whether our scores can detect such situations we generated a uniformly sampled square in two dimensions (with 400 datapoints) that has a square hole in the middle (see plots in the first row of Figure 4). Applying LLE and ISOMAP (both with $k = 10$) we compare the results. Note that the dataset and its "true" embedding are identical. Of course we could embed the two dimensional dataset into higher dimension using some high dimensional rotation, but the distances between the data points would not change and thus the dataset from the perspective of LLE and ISOMAP (which both consider only the distances) would not be different. LLE is able to embed the data, that is it finds a two dimensional representation that does not change the shape of the hole (left plot in the second row). On the other hand, the embedding found by ISOMAP has made the shape of the hole rounder (right plot in the second row). This behaviour is due to the fact that ISOMAP calculates all pair-wise distances along the graph. The length of paths that go around the corners are overestimated.

These findings are reflected by the Procrustes-, lml- and noivar-score as in the other experiments. Again, the stability-based scores fail.

7.2.3 Non-uniform sampling problem

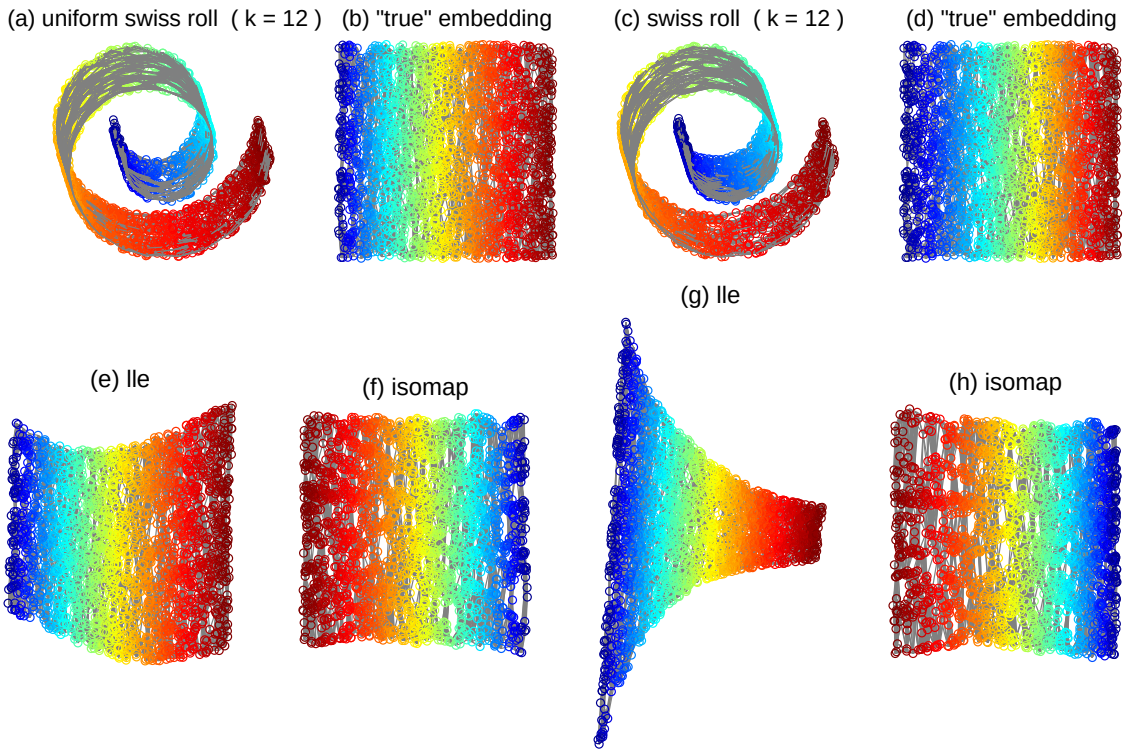
It is also well-known that the performance of LLE is influenced by the sampling distribution: the average local distance between data points of classic swiss roll (with 2000 data points, see two right-most plots in the first row of Figure 5) is much smaller in the inside loop than in the outside loop. This misleads LLE which prefers a uniform distribution in embedding space as can be seen from the result we get by applying LLE (with $k = 12$, see third plot from the left in the second row). ISOMAP (with $k = 12$ as well) does not depend on the embedding distribution (see right-most plot in the second row) and gives a good result. We can help LLE by changing the sampling along the manifold. The two left-most plots of the first row show a swiss roll (with 2000 data points) along which the density of the data points is kept constant. With this dataset LLE (again with $k = 12$) has less problems and calculates a reasonable embedding (see left-most plot in the second row). The embedding found by ISOMAP (again $k = 12$) is similarly good as for the non-uniformly sampled swiss roll.

These qualitative findings are matched by the Procrustes-, the lml- and the noivar-score. They all give similar scores to the two ISOMAP solutions and they are able to express the qualitative difference between the two LLE solutions. However, the Procrustes-score favors overall the ISOMAP solution of the left dataset, while the lml-score and the noivar-score favor the LLE-solution of the left dataset. Again the stability-based scores fail.



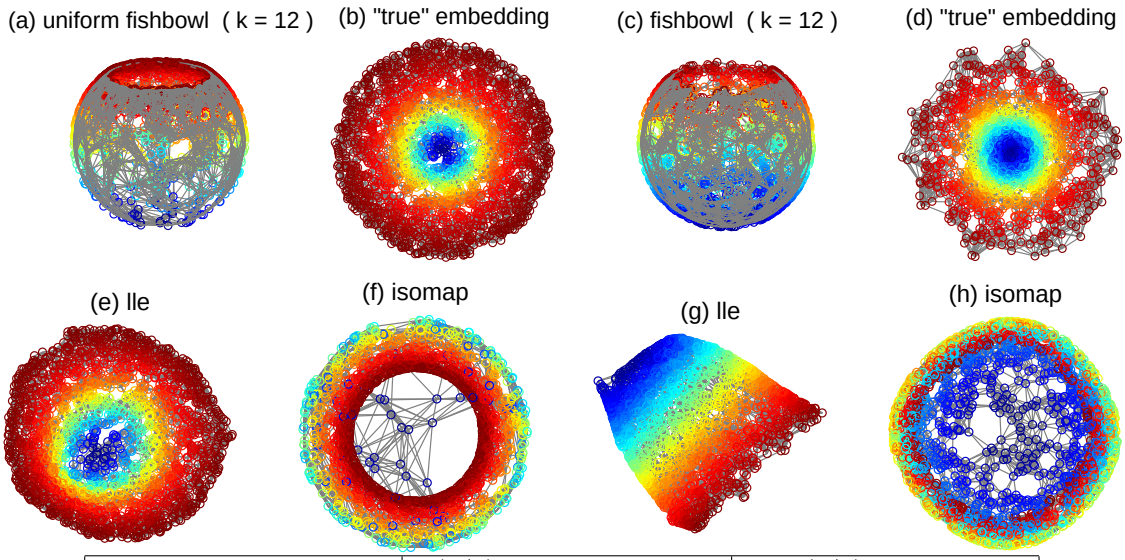
		(a/b) square with hole	
		(c) lle	(d) isomap
procrustes	dist	0.0041376	0.0059498
gplvm	nlml	-3782.1966	-1717.2723
	width	0.89443	0.58406
	sigvar	0.85311	0.86816
	noivar	0.00095137	0.016316
bootstrap 10	scatter	0.87956	-1.8208e-16
	offset	0.88357	-1.1102e-15
bootstrap 50	scatter	0.95198	-1.3429e-16
	offset	0.86755	1.4211e-16
noise inj 10 0.3	scatter	0.75738	-5.5067e-16
	offset	0.62082	-3.3307e-16
noise inj 10 0.6	scatter	0.86222	-1.5099e-16
	offset	0.84206	-7.5495e-16

Figure 4: (a) Uniform square with a hole with (b) its "true embedding" and (c) LLE's and (d) ISOMAP's embeddings. The table shows all scores for this dataset.



		(a/b) uniform swiss roll		(c/d) swiss roll	
		(e) lle	(f) isomap	(g) lle	(h) isomap
procrustes	dist	0.017211	0.0015772	0.15967	0.013028
gplvm	nlml	-33093.2119	-14377.6287	-27180.8851	-14689.2611
	width	0.42351	0.3172	0.46517	0.31701
	sigvar	0.72068	0.58185	0.96465	0.61964
	noivar	0.00048579	0.014584	0.0017158	0.013712
bootstrap 10	scatter	0.87535	-3.9968e-17	0.8727	-1.3323e-15
	offset	0.91465	-7.9936e-16	0.89428	-1.954e-15
bootstrap 50	scatter	0.95748	-1.5916e-16	0.95453	-1.4175e-15
	offset	0.91444	1.0658e-16	0.89838	-1.9895e-15
noise inj 10 0.3	scatter	0.74525	1.0658e-16	0.81463	-1.4211e-15
	offset	0.7589	1.9984e-16	0.7927	-1.2434e-15
noise inj 10 0.6	scatter	0.83904	-7.9936e-17	0.81486	-1.4566e-15
	offset	0.83888	-4.4409e-16	0.80041	-1.4655e-15

Figure 5: (a) Uniform swiss roll data with (b) its "true embedding" and (e) LLE's and (f) ISOMAP's embeddings, (c) classic swiss roll data with (d) its "true embedding" and (g) LLE's and (h) ISOMAP's embeddings. The table shows all scores for these datasets.



		(a/b) uniform fishbowl		(c/d) fishbowl	
		(e) lle	(f) isomap	(g) lle	(h) isomap
procrustes	dist	0.010063	0.24375	0.776	0.25521
gplvm	nlml	-17702.4822	1517.6244	5181.2392	4144.5873
	width	0.20999	0.33446	1.0127	0.43557
	sigvar	0.45094	1.1383	0.80622	0.77524
	noivar	0.0049896	0.27527	0.56249	0.45898
bootstrap 10	scatter	0.89019	7.5495e-17	0.89598	3.5527e-17
	offset	0.92727	1.9318e-15	0.96414	4.885e-16
bootstrap 50	scatter	0.9732	6.6969e-17	0.97532	2.5562e-16
	offset	0.94479	-7.3275e-16	0.96086	5.0182e-16
noise inj 10 0.3	scatter	0.84829	3.5083e-16	0.87082	2.6201e-16
	offset	0.88095	8.8818e-16	0.86952	4.6629e-16
noise inj 10 0.6	scatter	0.88313	3.1974e-16	0.88656	-4.5297e-16
	offset	0.91201	1.1546e-15	0.94562	4.6629e-16

Figure 6: (a) Uniform fishbowl data with (b) its "true embedding" and (e) LLE's and (f) ISOMAP's embeddings, (c) fishbowl data with (d) its "true embedding" and (g) LLE's and (h) ISOMAP's embeddings. The table shows all scores for these datasets.

7.2.4 Fishbowl problems

The "fishbowl" dataset[2] that is uniformly sampled in data space (with 2000 data points, see the two right-most plot in the first row of Figure 6) is difficult to embed for LLE and ISOMAP (both with $k = 12$) as can be seen by their resulting embeddings (see the two right-most plots in the second row). The reason for this failure for ISOMAP is that it tries to keep the original distances in data space also in embedding space. However, an embedding of the fishbowl must open up the top rim to make it flat. LLE does not try to preserve all distances but only local ones. Since LLE is also happy with an embedding that collapses far away points to the same location in embedding space (if it has to), we obtain a solution that looks like the fishbowl is squashed from the side. Of course both embeddings do not capture the "true" embedding (shown in the upper right corner). Changing the sampling of the fishbowl, such that it is more densely sampled at the rim of the bowl than at the bottom (with 2000 data points, see the two left-most plots in the first row of Figure 6), LLE (again with $k = 12$) is able to find a good solution that corresponds closely to the "true" embedding, which is actually uniformly sampled in embedding space. On the other hand, ISOMAP (also with $k = 12$) fails as well on the left dataset.

This qualitative discussion is matched again by Procrustes score (which requires the knowledge of the "true" embedding), by the log-likelihood, the lml-score, and the noivar-score. The ranking of all those scores agree, which again show that the GPLVM is useful for model selection of NLDR problem. Again the stability-based score depend on the algorithm and do not help.

7.3 Summary

First of all we note that the Procrustes-score that takes into account the unobservable "true" embedding best matches the visual qualitative judgements that the author made. Secondly, two scores based on the GPLVM, namely the negative log-likelihood and the noise-variance closely follow the Procrustes-score with some minor disagreements. However, noting that the scores based on the GPLVM are calculated without access to the ground truth, we suggest to use the GPLVM scores for model selection in the NLDR problem. On the other hand, the scores based on the stability idea failed. The reason for this failure is the fact that wrong solutions to the NLDR problem are often quite stable and that the algorithms react differently on the resampling procedures.

8 Conclusion

Model selection for unsupervised learning problems is generally difficult. This report explores and evaluates different scores for the model selection problem in the NLDR domain which is an example of an unsupervised problem. Extensive experiments that concentrate on situations in which common NLDR algorithms do or do not have problems, show that scores based on stability are not useful, while scores based on Gaussian processes are quite good at matching an "informed" score that takes into account the data generation process. The later scores are promising candidates that might help practitioners to judge their results they get from different off-the-shelf NLDR algorithm on their real-world datasets and hereby help to understand their data.

Another observation from this study is that while GPLVM provides a good score to evaluate the quality of an embedding at hand it is usually difficult to optimize the GPLVM criterion directly. This suggests for practitioner to first run LLE or ISOMAP and then secondly use the result as an initialization for GPLVM, which has the chance to overhaul the embedding, hereby to straighten out the disadvantages of LLE or ISOMAP.

[2]The data is sampled from the surface of a sphere of which the top cap is removed. Such a capped sphere looks like a fishbowl.

Acknowledgements

The author is grateful for valuable discussions with Chris Williams and Amos Storkey. This research was supported by the EU-PASCAL network of excellence (IST- 2002-506778) and through a European Community Marie Curie Fellowship (MEIF-CT-2005-025578).

References

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

[2] C.J.C. Burges. Geometric methods for feature extraction and dimensional reduction. *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers. Kluwer Academic Publishers*, 2005.

[3] T.F. Cox and M.A.A. Cox. *Multidimensional Scaling*. Chapman & Hall/CRC, 2001.

[4] D.L. Donoho and C. Grimes. When does Isomap recover the natural parameterization of families of articulated images. *Department of Statistics, Stanford University, Tech. Rep*, 27, 2002.

[5] B. Efron and R.J. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, New York, 1994.

[6] S. Harmeling, F. Meinecke, and K.-R. Müller. Injecting noise for analysing the stability of ICA components. *Signal Processing*, 84:255–266, 2004.

[7] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, 2001.

[8] N.D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. *Advances in Neural Information Processing Systems*, 16:329–336, 2004.

[9] N.D. Lawrence and J. Quiñonero-Candela. Local distance preservation in the GP-LVM through back constraints. *Proceedings of the 23rd International Conference on Machine Learning*, pages 513–520, 2006.

[10] F. Meinecke. Resampling-techniken für ICA und ihre anwendungen in der biomedizinischen datenanalyse. Diplomarbeit, Institut für Physik, Universität Potsdam, 2003.

[11] F. Meinecke, A. Ziehe, M. Kawanabe, and K.-R. Müller. Estimating the reliability of ICA projections. In T.G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2002.

[12] F. Meinecke, A. Ziehe, M. Kawanabe, and K.-R. Müller. A resampling approach to estimate the stability of one-dimensional or multidimensional independent components. *IEEE Transactions on Biomedical Engineering*, 49:1514–1525, Dez 2002.

[13] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

[14] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[15] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[16] L.K. Saul, K.Q. Weinberger, J. Ham, F. Sha, and D.D. Lee. Spectral methods for dimensionality reduction. In O. Chapelle, B. Schoelkopf, and A. Zien, editors, *Semisupervised Learning*. MIT Press, 2006.

[17] B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

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