Probabilistic Non-linear Principal Component Analysis with Gaussian Process Latent Variable Models

Neil Lawrence NEIL@DCS.SHEF.AC.UK
Department of Computer Science, University of Sheffield
Regent Court, 211 Portobello Street
Sheffield, S1 4DP, U.K.

Editor: Aapo Hyvärinen

Abstract

Summarising a high dimensional data set with a low dimensional embedding is a standard approach for exploring its structure. In this paper we provide an overview of some existing techniques for discovering such embeddings. We then introduce a novel probabilistic interpretation of principal component analysis (PCA) that we term dual probabilistic PCA (DPPCA). The DPPCA model has the additional advantage that the linear mappings from the embedded space can easily be non-linearised through Gaussian processes. We refer to this model as a Gaussian process latent variable model (GP-LVM). Through analysis of the GP-LVM objective function, we relate the model to popular spectral techniques such as kernel PCA and multidimensional scaling. We then review a practical algorithm for GP-LVMs in the context of large data sets and develop it to also handle discrete valued data and missing attributes. We demonstrate the model on a range of real-world and artificially generated data sets.

Keywords: Gaussian processes, latent variable models, principal component analysis, spectral methods, unsupervised learning, visualisation

1. Introduction

Machine learning is often split into three categories: supervised learning, where a data set is split into inputs and outputs; reinforcement learning, where typically a reward is associated with achieving a set goal, and unsupervised learning where the objective is to understand the structure of a data set. One approach to unsupervised learning is to represent the data, $\mathbf{Y}$, in some lower dimensional embedded space, $\mathbf{X}$. In a probabilistic model the variables associated with such a space are often known as latent variables. In this paper our focus will be on methods that represent the data in this latent (or embedded, we shall use the terms interchangeably) space.

Our approach is inspired by probabilistic latent variable models. It has roots in previously proposed approaches such as density networks (MacKay, 1995) where a multi-layer perceptron (MLP) is used to provide a mapping from the latent projections, $\mathbf{X}$, to the observed data, $\mathbf{Y}$. A prior distribution is placed over the latent-space and the latent-space's posterior distribution is approximated by sampling. Density networks made use of the MLP to perform the mapping, Bishop et al. (1996) replaced the MLP with a radial basis function (RBF) network with the aim of decreasing the training time for the model. This model evolved (Bishop et al., 1998) into the generative topographic mapping (GTM) where the latent-space was now sampled on a uniform grid, and importance sampling is reinterpreted as the fitting of a mixture model via the expectation-maximisation (EM) algorithm.

This allows the points in the latent-space to be laid out on a uniform grid[1] (rather than sampled). This grid layout is shared with the self organising map (SOM) (Kohonen, 1990) and in Bishop et al. (1997) it was argued that the GTM provides a principled alternative to the self organising map.

The models outlined above are typically designed to embed a data set in two dimensions, they rely on either importance sampling, or a grid of points in the latent-space to achieve this embedding, this causes problems when the dimensionality of the latent-space increases. Point representations of the latent-space are useful because they allow for non-linear models: each point is easy to propagate through the non-linear mapping to the data-space. These non-linear mappings are designed to address the weaknesses in visualising data sets that arise when using standard statistical tools that rely on linear mappings, such as principal component analysis (PCA) and factor analysis (FA): with a linear mapping it may not be possible to reflect the structure of the data through a low dimensional embedding.

Principal component analysis seeks a lower dimensional sub-space (typically represented by its orthonormal basis) in which the projected variance of the data is maximised. If a two dimensional sub-space is sought then the projections may be visualised; but it may be necessary to include more latent dimensions to capture the variability (and therefore hopefully, but by no means necessarily the structure) in the data. Principal component analysis also has a latent variable model representation (Tipping and Bishop, 1999) which is strongly related to Factor Analysis (FA) (Bartholomew, 1987; Basilevsky, 1994). Both are linear-Gaussian latent variable models, but FA allows for a richer noise model than PCA (for recent work on non-linear factor analysis see Honkela and Valpola, 2005).

Naturally statisticians have not constrained themselves to linear methods when visualising data and in the next section we shall briefly review multidimensional scaling and related techniques that rely on *proximity data*.

1.1 Multidimensional Scaling and Kernel PCA

We have already mentioned several visualisation techniques which rely on learning a mapping from a latent-space (the embedded space) to the data-space. In this section we will briefly review methods that use *proximity data* to obtain a visualisation or embedding. Broadly speaking these methods are all variants or enhancements of the technique known as multidimensional scaling (MDS). In these methods, rather than observing data directly, information about the data set is summarised in an $N \times N$ matrix of either similarities or dissimilarities. Examples include distance matrices (a dissimilarity matrix) and kernel matrices (a similarity matrix). Each method we review provides answers to at least one of two questions.

1. How is the proximity matrix compiled?

2. How is the embedding developed from the proximity matrix?

Most of the variants of multidimensional scaling (Mardia et al., 1979) appear to focus on the second question. In classical MDS (Torgerson, 1952) an eigendecomposition of the centred similarity matrix[2] is performed. This is sometimes viewed as minimising a particular *stress function* where distances in the visualised space are matched to those in the data space. Attempting to preserve these distances is known as *metric* MDS, in *non-metric* MDS only the ordering of distances is preserved.

[1] When sampling techniques are used the latent points will be in random positions.

[2] When the data is presented in the form of a distance or dissimilarity matrix a simple conversion may be performed to obtain a similarity matrix.

There are strong connections between MDS and kernel PCA (Schölkopf et al., 1998), some of which are formalised in Williams (2001). Kernel PCA also provides an answer to the first question—the suggestion is that the proximity data is provided by a positive semi-definite Mercer kernel that is computed on the data, $\mathbf{Y}$. The use of this kernel implies the existence of a non-linear mapping[3] from the data-space to the latent-space (recall that the GTM and density networks perform the non-linear mapping in the opposite direction). The existence of this function is important as it allows data points which were not in the training set to be mapped to a position in the latent space without re-solving the eigenvalue problem. However, for both kernel PCA and MDS methods, it is not obvious how to project back from the latent-space to the data-space (this is known as the pre-image problem). Neither is it clear how to handle missing data[4] as the proximity data matrix cannot normally be computed consistently if a particular attribute is not available.

Sammon mappings (Sammon, 1969) also attempt to match the embedded distances between points with the distances in the observed space (therefore they are a form of MDS). They suffer from the same weakness as MDS in that projection of data points which were not in the original data set can be computationally demanding, *i.e.* despite their name they do not provide an explicit mapping between the data and latent-space. The lack of a mapping was addressed by the Neuroscale algorithm of Lowe and Tipping (1996) a version of which was also suggested for MDS (Tipping, 1996).

Other recent work of importance which has focussed on forming the proximity matrix includes Isomap (Tenenbaum et al., 2000), where an approximation to geodesic distance is used and spectral clustering (see *e.g.* Shi and Malik, 2000) where the proximity data is derived from a graph.

In Table 1 we have summarised some of the properties of these algorithms/models. We have also included the model that is the subject of this paper, the Gaussian process latent variable model (GP-LVM).

In the remainder of this paper we will introduce the GP-LVM from the latent variable model perspective. The GP-LVM belongs to the same class of methods as density networks and the GTM, however there are also connections to classical MDS and kernel PCA. In particular, in the next section, we show that the approaches share an objective function. In Section 3 we will cover some of the algorithmic issues that arise with the model. The framework within which our GP-LVM is developed makes it straightforward to modify the approach for data for which a Gaussian noise model is not appropriate (such as binary or ordinal), this is discussed in Section 5. Handling of missing data attributes is also straightforward (Section 6). The algorithm's characteristics are explored empirically in Section 7.

2. Gaussian Process Latent Variable Models

In this paper we present the Gaussian process latent variable model. As we shall see, the model is strongly related to many of the approaches that we have outlined above. There is a point representation in the latent-space (as there was for the GTM and density networks) and we will minimise an objective function that can be related to classical MDS and kernel PCA (see Section 2.6). Our starting point, however, will be a novel probabilistic interpretation of principal component analysis

[3] A good reference which introduces Mercer kernels is Schölkopf and Smola (2001) Chapter 2.

[4] Here, by missing data, we mean missing attributes which would normally be used in computing the proximity data matrix. For proximity data methods missing data can also mean elements missing from the proximity matrix, we do not discuss this case.

	Proximity	$\mathbf{X} \rightarrow \mathbf{Y}$	$\mathbf{Y} \rightarrow \mathbf{X}$	Non-linear	Probabilistic	Convex
PCA	I	Y	Y		I	Y
FA		Y	Y		Y	Y
Kernel PCA	Y		Y	Y		Y
MDS	Y			Y		
Sammon mapping	Y			Y		
Neuroscale	Y		Y	Y		
Spectral clustering	Y			Y		Y
Density Networks		Y		Y	Y	
GTM		Y		Y	Y	
GP-LVM	I	Y		Y	Y	

Table 1: Overview of the relationship between algorithms. A ‘Y’ indicates the algorithm exhibits that property, an ‘I’ indicates that there is an interpretation of the algorithm that exhibits the associated property. The characteristics of the algorithm are: *proximity*: is the method based on proximity data? $\mathbf{X} \rightarrow \mathbf{Y}$: does the method lead to a mapping from the embedded to the data-space? $\mathbf{Y} \rightarrow \mathbf{X}$: does the method lead to a mapping from data to embedded space? *Non-linear*: does the method allow for non-linear embeddings? *Probabilistic*: is the method probabilistic? *Convex:* algorithms that are considered convex have a unique solution, for the others local optima can occur.

which we will refer to as dual probabilistic principal component analysis (DPPCA). Dual probabilistic principal component analysis turns out to be a special case of the more general class of models we refer to as GP-LVMs.

2.1 Latent Variable Models

Typically we specify a latent variable model relating a set of latent variables, $\mathbf{X} \in \Re^{N\times q}$, to a set of observed variables, $\mathbf{Y} \in \Re^{N\times D}$, through a set of parameters. The model is defined probabilistically, the latent variables are then marginalised and the parameters are found through maximising the likelihood.

Here we consider an alternative approach: rather than marginalising the latent variables and optimising the parameters we marginalise the parameters and optimise the latent variables. We will show how the two approaches can be equivalent: for a particular choice of Gaussian likelihood and prior both approaches lead to a probabilistic formulation of principal component analysis (PCA). In the next section we will review the standard derivation of probabilistic PCA (Tipping and Bishop, 1999), then we will show how an alternative probabilistic formulation may be arrived at (see also Appendix A).

2.2 Probabilistic PCA

Probabilistic PCA (PPCA) is a latent variable model in which the maximum likelihood solution for the parameters is found through solving an eigenvalue problem on the data’s covariance matrix

(Tipping and Bishop, 1999). Let's assume that we are given a set of centred D-dimensional data $\mathbf{Y} = [\mathbf{y}_1 \ldots \mathbf{y}_N]^{\mathrm{T}}$. We denote the q-dimensional latent variable associated with each data point by $\mathbf{x}_n$. The relationship between the latent variable and the data point is linear with noise added,

$$\mathbf{y}_n = \mathbf{W}\mathbf{x}_n + \eta_n$$

where the matrix $\mathbf{W} \in \Re^{D\times q}$ specifies the linear relationship between the latent-space and the data space and the noise values, $\eta_n \in \Re^{D\times 1}$, are taken to be an independent sample from a spherical Gaussian distribution[5] with mean zero and covariance $\beta^{-1}\mathbf{I}$,

$$p(\eta_n) = N\left(\eta_n|\mathbf{0}, \beta^{-1}\mathbf{I}\right).$$

The likelihood for a data point can then be written as

$$p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{W}, \beta) = N\left(\mathbf{y}_n|\mathbf{W}\mathbf{x}_n, \beta^{-1}\mathbf{I}\right). \tag{1}$$

To obtain the marginal likelihood we integrate over the latent variables,

$$p(\mathbf{y}_n|\mathbf{W}, \beta) = \int p(\mathbf{y}_n|\mathbf{x}_n, \mathbf{W}, \beta)\, p(\mathbf{x}_n)\, d\mathbf{x}_n, \tag{2}$$

which requires us to specify a prior distribution over $\mathbf{x}_n$. For probabilistic PCA the appropriate prior is a unit covariance, zero mean Gaussian distribution,

$$p(\mathbf{x}_n) = N(\mathbf{x}_n|\mathbf{0}, \mathbf{I}).$$

The marginal likelihood for each data point can then be found analytically (through the marginalisation in (2)) as

$$p(\mathbf{y}_n|\mathbf{W}, \beta) = N\left(\mathbf{y}_n|\mathbf{0}, \mathbf{W}\mathbf{W}^{\mathrm{T}} + \beta^{-1}\mathbf{I}\right).$$

Taking advantage of the independence of the data points, the likelihood of the full data set is given by

$$p(\mathbf{Y}|\mathbf{W}, \beta) = \prod_{n=1}^{N} p(\mathbf{y}_n|\mathbf{W}, \beta). \tag{3}$$

The parameters $\mathbf{W}$ can then be found through maximisation of (3). Tipping and Bishop (1999) showed that there is an analytic solution to this maximisation. This solution is achieved when the matrix $\mathbf{W}$ spans the principal sub-space of the data. This model therefore has an interpretation as a *probabilistic* version of PCA.

Marginalising the latent variables and optimising the parameters via maximum likelihood is a standard approach for fitting latent variable models. In the next section we will introduce an alternative approach. Instead of optimising parameters and marginalising latent variables we will suggest the dual approach of marginalising parameters, $\mathbf{W}$, and optimising with respect to latent variables, $\mathbf{X}$. For a particular choice of prior distribution on $\mathbf{W}$ this probabilistic model will also turn out to be equivalent to PCA.

[5] We use the notation $N(\mathbf{z}|\mu, \Sigma)$ to denote a Gaussian distribution over $\mathbf{z}$ with mean μ and covariance Σ.

2.3 Probabilistic PCA through Latent Variable Optimisation

In the Bayesian framework parameters, such as $\mathbf{W}$, are viewed as random variables. The Bayesian methodology requires a suitable choice of prior for $\mathbf{W}$, and then proceeds to treat the parameters as latent variables. A simple choice of prior that is conjugate to (1) would be a spherical Gaussian distribution:

$$p(\mathbf{W}) = \prod_{i=1}^{D} N(\mathbf{w}_i|\mathbf{0},\mathbf{I})$$

where $\mathbf{w}_i$ is the ith row of the matrix $\mathbf{W}$. Unfortunately[6] marginalisation of both $\mathbf{W}$ and $\mathbf{X} = [\mathbf{x}_1 \ldots \mathbf{x}_N]^{\mathbf{T}}$ is intractable. If we wish to proceed without turning to approximate methods we are faced with a choice over what to marginalise. The natural choice seems to be to marginalise $\mathbf{X} \in \Re^{N\times q}$ as typically it will be of larger dimension[7] than $\mathbf{W} \in \Re^{D\times q}$. In practice though, it turns out that the two approaches are equivalent.

Marginalisation of $\mathbf{W}$ is straightforward due to our choice of a conjugate prior. The resulting marginalised likelihood takes the form

$$p(\mathbf{Y}|\mathbf{X},\beta) = \prod_{d=1}^{D} p(\mathbf{y}_{:,d}|\mathbf{X},\beta), \tag{4}$$

where we use $\mathbf{y}_{:,d}$ to represent the dth column of $\mathbf{Y}$ and

$$p(\mathbf{y}_{:,d}|\mathbf{X},\beta) = N\left(\mathbf{y}_{:,d}|\mathbf{0},\mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta^{-1}\mathbf{I}\right). \tag{5}$$

We now look to optimise with respect to the latent variables. As might be expected from the duality of (3) and (4), this optimisation is very similar to that presented in Tipping and Bishop (1999). Our objective function is the log-likelihood,

$$L = -\frac{DN}{2}\ln 2\pi - \frac{D}{2}\ln|\mathbf{K}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right), \tag{6}$$

where

$$\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta^{-1}\mathbf{I}.$$

The gradients of (6) with respect to $\mathbf{X}$ may be found (Magnus and Neudecker, 1999) as,

$$\frac{\partial L}{\partial \mathbf{X}} = \mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{X} - D\mathbf{K}^{-1}\mathbf{X},$$

a fixed point where the gradients are zero is then given by

$$\frac{1}{D}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{X} = \mathbf{X}.$$

In Appendix B we show how the values for $\mathbf{X}$ which maximise the likelihood are given by

$$\mathbf{X} = \mathbf{U}\mathbf{L}\mathbf{V}^{\mathrm{T}}$$

[6] If it were possible to marginalise both the parameters and latent variables analytically we could use Bayes factors to perform model selection (see, for example, Bishop, 1999).

[7] The matrix $\mathbf{X}$ will be of larger dimension than $\mathbf{W}$ unless $D > N$, *i.e.* there are more features than data points.

where $\mathbf{U}$ is an $N \times q$ matrix whose columns are the first q eigenvectors of $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$, $\mathbf{L}$ is a $q \times q$ diagonal matrix whose jth element is $l_j = \left(\lambda_j - \frac{1}{\beta}\right)^{-\frac{1}{2}}$ where λ_j is the eigenvalue associated with the jth eigenvector of $D^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ and $\mathbf{V}$ is an arbitrary $q \times q$ rotation matrix. Here, and in what follows, we will assume that these eigenvalues are ordered according to magnitude with the largest being placed first. Note that the eigenvalue problem we have developed can easily be shown to be equivalent to that solved in PCA (see Appendix C), indeed the formulation of PCA in this manner is a key step in the development of kernel PCA (Schölkopf et al., 1998) where the matrix of inner products $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ is replaced with a kernel (see Tipping (2001) for a concise overview of this derivation). Our probabilistic PCA model shares an underlying structure with that of Tipping and Bishop (1999) but differs in that where they optimise we marginalise and where they marginalise we optimise.

2.4 Gaussian Processes

Gaussian processes (O'Hagan, 1992; Williams, 1998) are a class of probabilistic models which specify distributions over function spaces. While a function is an infinite dimensional object, a distribution over the function space can be considered by focussing only on points where the function is instantiated. In Gaussian processes the distribution over these instantiations is taken to be Gaussian. Modelling with Gaussian processes consists of first specifying a Gaussian process prior. Usually a Gaussian distribution is parameterised by a mean and a covariance. In the case of Gaussian processes the mean and covariance must be functions of the space on which the process operates. Typically the mean function is taken to be zero, while the covariance function is necessarily constrained to produce positive definite matrices.[8]

Consider a simple Gaussian process prior over the space of functions that are fundamentally linear, but are corrupted by Gaussian noise of variance $\beta^{-1}\mathbf{I}$. The covariance function, or kernel, for such a prior is given by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j + \beta^{-1}\delta_{ij}, \tag{7}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ are vectors from the space of inputs to the function and δ_{ij} is the Kronecker delta. If these inputs were taken from our embedding matrix, $\mathbf{X}$, and the covariance function was evaluated at each of the N points we would recover a covariance matrix of the form

$$\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta^{-1}\mathbf{I}, \tag{8}$$

where the element at the ith row and jth column of $\mathbf{K}$ is given by (7). This is recognised as the covariance associated with each factor of the marginal likelihood for dual probabilistic PCA (5). The marginal likelihood for dual probabilistic PCA is therefore a product of D independent Gaussian processes. In principal component analysis we are optimising the parameters *and* input positions of a Gaussian process prior distribution where the (linear) covariance function for each dimension is given by $\mathbf{K}$.

2.5 Gaussian Process Latent Variable Models

The dual interpretation of probabilistic PCA described above points to a new class of models which consist of Gaussian process mappings from a latent space, $\mathbf{X}$, to an observed data-space, $\mathbf{Y}$. Dual

[8] The positive definite constraint implies that these covariance functions are also valid Mercer kernels. It is therefore common to refer to the covariance function as a kernel. In this paper we shall use the two terms interchangeably.

probabilistic PCA is the special case where the output dimensions are *a priori* assumed to be linear, independent and identically distributed. However, each of these assumptions can be infringed to obtain *new* probabilistic models. Independence can be broken, for example, by allowing an arbitrary rotation on the data matrix $\mathbf{Y}$, the 'identically distributed' assumption can be broken by allowing different covariance functions for each output dimension.[9] In this paper we focus on the third assumption, linearity. By replacing the inner product kernel with a covariance function that allows for non-linear functions we can obtain a non-linear latent variable model. Due to the close relationship with the linear model, which has an interpretation as probabilistic PCA, such a model can be interpreted as a non-linear probabilistic version of PCA.

2.6 Proximity Data and the GP-LVM

We indicated in the introduction that the GP-LVM has connections with proximity data based methods such as kernel PCA and classical MDS. These connections are through a unifying objective function which embraces all three models. In the next section we briefly introduce this objective function.

2.6.1 A Unifying Objective Function

Classical MDS and kernel PCA rely on proximity data, such as similarity matrices. Let's denote the matrix of similarities for these methods by $\mathbf{S}$. For the case of positive definite[10] similarity measures the matrix $\mathbf{S}$ can be interpreted as a covariance (or covariance function). The cross entropy between this Gaussian and the Gaussian process whose marginal likelihood was given in (4) is

$$-\int N(\mathbf{z}|\mathbf{0},\mathbf{S}) \ln N(\mathbf{z}|\mathbf{0},\mathbf{K})\, d\mathbf{z} = \frac{N}{2}\ln 2\pi + \frac{1}{2}\ln|\mathbf{K}| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-\mathbf{1}}\mathbf{S}\right). \tag{9}$$

If we substitute $\mathbf{S} = D^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ we see, up to a scaling of $-D$, that (9) becomes identical to (6). Taking $\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta^{-1}\mathbf{I}$ and minimising (9) with respect to $\mathbf{X}$ leads to a solution (Appendix B) of the form

$$\mathbf{X} = \mathbf{U}\mathbf{L}\mathbf{V}^{\mathrm{T}}.$$

Where the matrix $\mathbf{U} \in \Re^{N\times q}$ has columns which are the eigenvectors of $\mathbf{S}$. For the specific case[11] where $\mathbf{S} = D^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ the optimisation is identical to that of dual probabilistic PCA. However in the more general case where $\mathbf{S}$ is either a kernel function or simply a positive definite matrix of similarities kernel PCA and classical MDS are recovered. We also note that the entropy of $N(\mathbf{z}|\mathbf{0},\mathbf{S})$ is constant in $\mathbf{X}$, we therefore may subtract it from our objective function without affecting the optimisation with respect to $\mathbf{X}$. The resulting objective function is then the Kullback-Leibler divergence (Kullback and Leibler, 1951) between the two Gaussians,

$$\begin{aligned}\mathrm{KL}\left(N(\mathbf{z}|\mathbf{0},\mathbf{S})\,||N(\mathbf{z}|\mathbf{0},\mathbf{K})\right) &= -\int N(\mathbf{z}|\mathbf{0},\mathbf{S}) \ln \frac{N(\mathbf{z}|\mathbf{0},\mathbf{K})}{N(\mathbf{z}|\mathbf{0},\mathbf{S})} d\mathbf{z} \\ &= \frac{1}{2}\ln|\mathbf{K}| - \frac{1}{2}\ln|\mathbf{S}| + \frac{1}{2}\mathrm{tr}\left(\mathbf{S}\mathbf{K}^{-1}\right) - \frac{N}{2}.\end{aligned}$$

[9] A very simple example of this idea would be to allow different noise distributions on each output direction. The probabilistic model underlying factor analysis allows this flexibility (see, for example, Tipping and Bishop 1999).

[10] The analysis that follows can be extended to positive semi-definite $\mathbf{S}$ by adding a diagonal term, $\sigma^2\mathbf{I}$ to $\mathbf{S}$ and considering the limit as $\sigma^2 \to 0$.

[11] In the MDS literature this is also sometimes referred to as principal co-ordinate analysis.

With appropriate choice of **S** and **K** this is a valid objective function for PCA, kernel PCA, classical MDS and the GP-LVM. For kernel PCA **S** is a 'non-linear kernel' and **K** is the 'linear kernel'. For the GP-LVM **S** is the 'linear kernel' whereas **K** is a 'non-linear kernel'. In practice this means that the GP-LVM is harder to optimise (solving an eigenvalue problem is no longer sufficient) but the GP-LVM maintains a probabilistic interpretation that kernel PCA doesn't have.

The methods overlap when both **K** and **S** are based on inner product matrices (as outlined for DPPCA above).

Note that when the similarity measure, **S**, is not of the form of the inner product kernel the objective function no longer has an interpretation as a likelihood. Therefore our approach is *not* a probabilistic interpretation of multidimensional scaling: we refer the reader to MacKay and Zinnes (1986) and Oh and Raftery (2001) for details of probabilistic MDS methods.

2.6.2 A NOTE ON REVERSING THE KULLBACK-LEIBLER DIVERGENCE

The Kullback-Leibler divergence is an asymmetric measure of distribution divergence so it is natural to consider the effect of reversing the role of the distributions and taking expectations under the distribution governed by **K** rather than that governed by **S**. For this special case, the reversed KL divergence is very similar to the original, only all matrices **K** and **S** are now replaced with their inverses. So the new objective function is

$$L = \frac{1}{2}\ln|\mathbf{S}| - \frac{1}{2}\ln|\mathbf{K}| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}\mathbf{S}^{-1}\right) - \frac{N}{2},$$

The minimum can again be found through an eigenvalue problem, but now the retained eigenvalues from **K** are the smallest, rather than the largest. In this respect the model leads to *minor component analysis*.

3. Fitting a Non-linear GP-LVM

We saw in the previous section how PCA can be interpreted as a Gaussian process that maps latent-space points to points in data-space. The positions of the points in the latent-space can be determined by maximising the process likelihood with respect to **X**. It is natural, therefore, to consider alternative GP-LVMs by introducing covariance functions which allow for non-linear processes. The resulting models will not, in general, be optimisable through an eigenvalue problem.

3.1 Optimisation of the Non-linear Model

In the previous section we saw for the linear kernel that a closed form solution could be obtained up to an arbitrary rotation matrix. Typically, for non-linear kernels, there will be no such closed form solution and there are likely to be multiple local optima. There is a wide choice of non-linear covariance functions, some of which will be reviewed in Section 7.1. To use a particular kernel in the GP-LVM we first note that gradients of (6) with respect to the latent points can be found through first taking the gradient with respect to the kernel,

$$\frac{\partial L}{\partial \mathbf{K}} = \mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}^{-1} - D\mathbf{K}^{-1}, \tag{10}$$

and then combining it with $\frac{\partial \mathbf{K}}{\partial x_{n,j}}$ through the chain rule. As computation of (10) is straightforward and independent of the kernel choice we only require that the gradient of the kernel with respect to the latent points can be computed. These gradients may then be used in combination with (6) in a non-linear optimiser to obtain a latent variable representation of the data. Furthermore, gradients with respect to the parameters of the kernel matrix may be computed and used to jointly optimise **X** and the kernel's parameters.

The log-likelihood is a highly non-linear function of the embeddings and the parameters. We are therefore forced to turn to gradient based optimisation of the objective function. Scaled conjugate gradient (Møller, 1993) is an approach to optimisation which implicitly considers second order information while using a scale parameter to regulate the positive definitiveness of the Hessian at each point. We made use of scaled conjugate gradient (SCG) for our experiments.

3.2 Illustration of GP-LVM via SCG

To illustrate a simple Gaussian process latent variable model we turn to the 'multi-phase oil flow' data (Bishop and James, 1993). This is a twelve dimensional data set containing data of three known classes corresponding to the phase of flow in an oil pipeline: stratified, annular and homogeneous. In Bishop et al. (1998), see also Section 7.2.1, this data was used to demonstrate the GTM algorithm. The data set is artificially generated and therefore is known to lie on a lower dimensional manifold. Here we use a sub-sampled version of the data (containing 100 data points) to demonstrate the fitting of a GP-LVM with a simple radial basis function (RBF) kernel.

As we saw in Section 2.3, seeking a lower dimensional embedding with PCA is equivalent to a GP-LVM model with a linear kernel,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j + \beta^{-1} \delta_{ij},$$

where $k(\mathbf{x}_i, \mathbf{x}_j)$ is the element in the ith row and the jth column of the kernel matrix **K** and δ_{ij} is the Kronecker delta function.

For comparison we visualised the data set using several of the approaches mentioned in the introduction. In Figure 1(a) we show the first two principal components of the data. Figure 1(b) then shows the visualisation obtained using the GP-LVM with the RBF kernel,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \theta_{\mathrm{rbf}} \exp\left(-\frac{\gamma}{2}(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)\right) + \theta_{\mathrm{bias}} + \theta_{\mathrm{white}} \delta_{ij}.$$

To obtain this visualisation the log likelihood was optimised jointly with respect to the latent positions **X** and the kernel parameters θ_{bias}, θ_{white}, θ_{rbf} and γ. The kernel was initialised using PCA to set **X**, the kernel parameters were initialised as $\theta_{\mathrm{rbf}} = \gamma = 1$ and $\theta_{\mathrm{white}} = \theta_{\mathrm{bias}} = \exp(-1)$.

Note that there is a redundancy in the representation between the overall scale of the matrix **X** and the value of γ. This redundancy was removed by penalising the log likelihood (6) with half the sum of the squares of each element of **X**: this implies we were actually seeking a MAP solution[12] with a Gaussian prior for **X**,

$$p(\mathbf{X}) = \prod_{n=1}^{N} N(\mathbf{x}_n | \mathbf{0}, \mathbf{I}).$$

The likelihood for the RBF kernel was optimised using scaled conjugate gradient (see `http://www.dcs.shef.ac.uk/~neil/gplvmapp/` for the code used).

[12] Multiplying the likelihood by this prior leads to a joint distribution over data points and latent points. As a function of **X** this joint distribution is proportional to the posterior distribution $p(\mathbf{X}|\mathbf{Y})$, therefore maximising the joint distribution is equivalent to seeking a MAP solution.

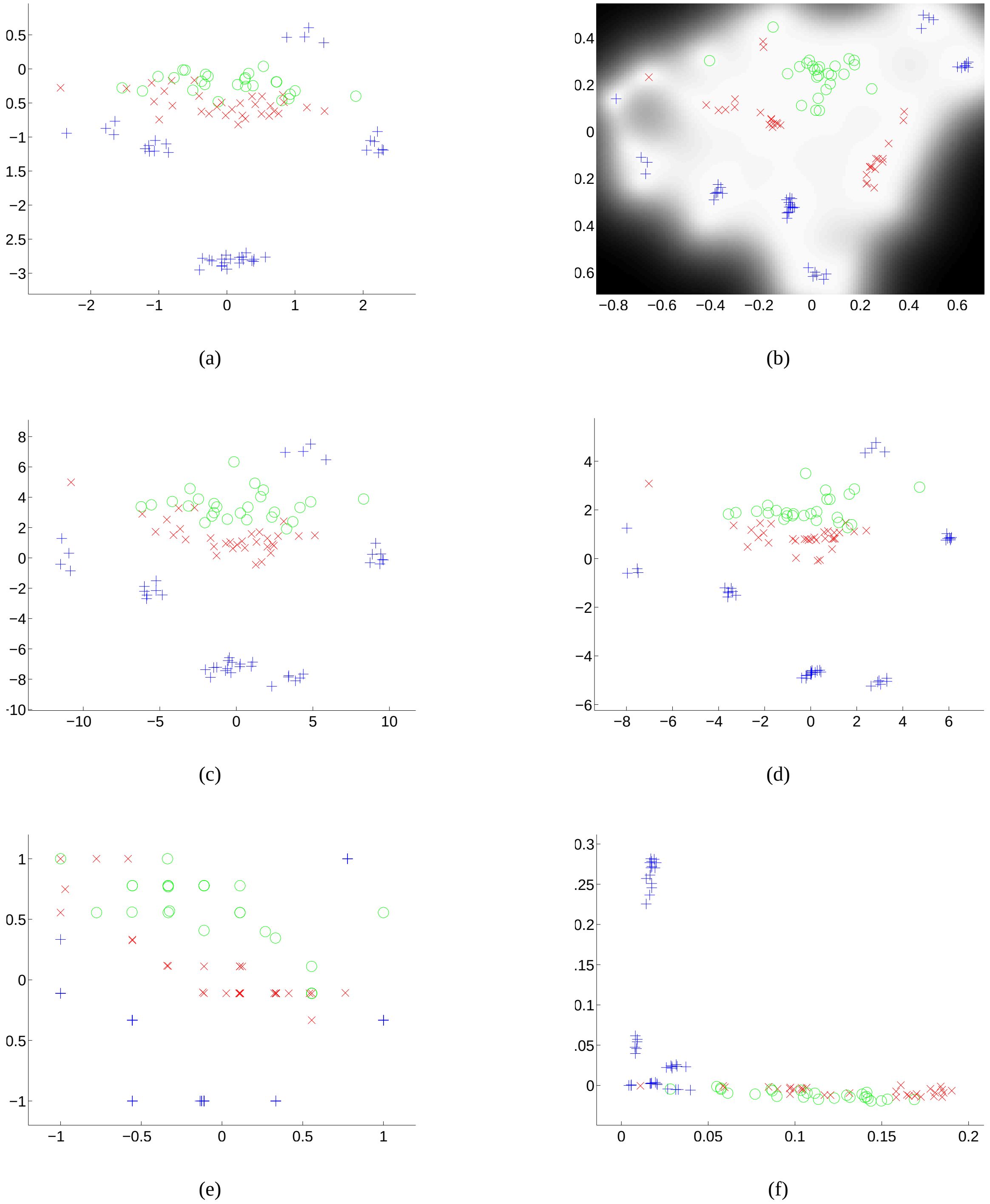


Figure 1: Visualisation of the Oil data with (a) PCA (a linear GP-LVM) and (b) A GP-LVM which uses an RBF kernel, (c) Non-metric MDS using Kruskal's stress, (d) Metric MDS using the 'Sammon Mapping', (e) GTM and (f) kernel PCA. Red crosses, green circles and blue plus signs represent stratified, annular and homogeneous flows respectively. The greyscales in plot (b) indicate the precision with which the manifold was expressed in data-space for that latent point.

Method	PCA	GP-LVM	Non-metric MDS	Metric MDS	GTM*	kernel PCA*
Errors	20	4	13	6	7	13

Table 2: Errors made by the different methods when using the latent-space for nearest neighbour classification in the latent space. Both the GTM and kernel PCA are given asterisks as the result shown is the best obtained for each method from a range of different parameterisations.

We also provide visualisations of the data using the range of algorithms we reviewed in the introduction. In Figure 1(c) we show the result of non-metric MDS using the stress criterion of Kruskal (1964). Figure 1(d) shows the result from metric MDS using the criterion of Sammon (1969). To objectively evaluate the quality of the visualisations we classified each data point according to the class of its nearest neighbour in the two dimensional latent-space supplied by each method. The errors made by such a classification are given in Table 2. For the GTM and kernel PCA some selection of parameters is required. For GTM we varied the size of the latent grid between 3×3 and 15×15, and the number of hidden nodes in the RBF network was varied between 4 and 36. The best result was obtained for a 10×10 latent grid with 25 nodes in the RBF network, it is shown in Figure 1(e). Note the characteristic gridding effect in the GTM's visualisation which arises from the layout of the latent points. For kernel PCA we used the RBF kernel and varied the kernel width between 0.01 and 100. The best result was obtained for a kernel width of 0.75, the associated visualisation is shown in Figure 1(f).

The gradient based optimisation of the RBF based GP-LVM's latent-space shows results which are clearly superior (in terms of separation between the different flow phases) to those achieved by the linear PCA model. The GP-LVM approach leads to a number of errors that is the smallest of all the approaches used. Additionally the use of a Gaussian process to perform our 'mapping' means that we can express uncertainty about the positions of the points in the *data* space. For our formulation of the GP-LVM the level of uncertainty is shared across all D dimensions and thus may be visualised in the latent-space.

3.2.1 VISUALISING THE UNCERTAINTY

Recall that the likelihood (4) is a product of D separate Gaussian processes. In this paper we chose to retain the implicit assumption in PCA that *a priori* each dimension is identically distributed by assuming that the processes shared the same covariance/kernel function $\mathbf{K}$. Sharing of the covariance function also leads to an *a posteriori* shared level of uncertainty in each process. While it is possible to use different covariance functions for each dimension and may be necessary when each of the data's attributes have different characteristics;[13] the more constrained model implemented here allows us to visualise the uncertainty in the latent space and will be preferred for our empirical

[13] A simple example of this is given by Grochow et al. (2004) with the 'scaled GP-LVM', where a scale parameter is associated with each dimension of the data.

studies.[14] In Figure 1(b) (and subsequently) the uncertainty is visualised by varying the intensity of the background pixels. The lighter the pixel the higher the precision of the mapping.

3.2.2 COMPUTATIONAL COMPLEXITY

While the quality of the results seem good, a quick analysis of the algorithmic complexity shows that each gradient step requires an inverse of the kernel matrix (see (10)), an $O\left(N^3\right)$ operation, rendering the algorithm impractical for many data sets of interest. In the next section we will show how a practical algorithm may be developed which circumvents this problem through maximising a sparse approximation to (6).

4. A Practical Algorithm for GP-LVMs

So far we have shown that PCA can be viewed probabilistically from two perspectives, the first involves integrating latent variables and the second optimising them. Using the latter perspective we can develop a non-linear probabilistic version of PCA. Unfortunately the optimisation problem we are faced with is then non-linear and high dimensional (Nq interdependent parameters/latent-variables before we consider the parameters of the kernel). In this section we will describe an approximation that relies on a forced 'sparsification' of the model. The resulting computational advantages make visualisation of large numbers of data points practical. We base our approach on the informative vector machine algorithm (Lawrence et al., 2003). As we will see in Section 5, this machinery has the added advantage of allowing us to extend our non-linear PCA model to non-Gaussian noise models.

4.1 Sparsification

Kernel methods may be sped up through sparsification, *i.e.* representing the data set by a subset, I, of d points known as the *active set.* The remaining points are denoted by J. We make use of the informative vector machine (IVM) which selects points sequentially according to the reduction in the posterior process's entropy that they induce: implementation details for the IVM algorithm are given in Lawrence et al. (2003).

A consequence of this enforced sparsification is that optimisation of the points in the active set (with $d < N$) proceeds much quicker than the optimisation of the full set of latent variables: the likelihood of the active set is given by

$$p\left(\mathbf{Y}_I\right) = \frac{1}{(2\pi)^{\frac{D}{2}}\left|\mathbf{K}_{I,I}\right|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_{I,I}^{-1}\mathbf{Y}_I\mathbf{Y}_I^{\mathrm{T}}\right)\right), \tag{11}$$

which can be optimised with respect to the kernel's parameters and $\mathbf{X}_I$ with gradient evaluations costing $O\left(d^3\right)$ rather than the prohibitive $O\left(N^3\right)$ which would arise in the full model. The dominant cost (asymptotically) becomes that of the active selection which is $O\left(d^2N\right)$.

[14] The two approaches, constraining each data direction to the same kernel and allowing each data dimension to have its own kernel are somewhat analogous to the difference between probabilistic PCA, where each output data shares a variance, and factor analysis, where each data dimension maintains its own variance.

Algorithm 1 An algorithm for visualisation with a GP-LVM.

```
Require: A size for the active set, $d$. A number of iterations, $T$.
  Initialise $\mathbf{X}$ through PCA.
  for $T$ iterations. do
    Select a new active set using the IVM algorithm.
    Optimise (11) with respect to the parameters of $\mathbf{K}$ (and optionally the latent positions $\mathbf{X}_I$) using
    scaled conjugate gradients.
    Select a new active set.
    for each point not in active set $j$. do
      Optimise (12) with respect to $\mathbf{x}_j$ using scaled conjugate gradients.
    end for
  end for
```

4.2 Latent Variable Optimisation

We are interested in visualising all points in the data set, so while there is a significant speed advantage to selecting an active set, we still need to optimise the *inactive points.* Fortunately, active set selection allows us to optimise each of these points independently as, given a fixed active set, the individual data points are no longer interdependent. A standard result for Gaussian processes (see *e.g.* Williams, 1998) is that a point, j, from the inactive set can be shown to project into the data-space as a Gaussian distribution

$$p\left(\mathbf{y}_j|\mathbf{x}_j\right) = N\left(\mathbf{y}_j|\mu_j, \sigma_j^2\mathbf{I}\right) \tag{12}$$

whose mean is

$$\mu_j = \mathbf{Y}^{\mathrm{T}}\mathbf{K}_{I,I}^{-1}\mathbf{k}_{I,j}$$

where $\mathbf{K}_{I,I}$ denotes the kernel matrix developed from the active set and $\mathbf{k}_{I,j}$ made up of rows in I from the jth column of $\mathbf{K}$, and the variance[15] is

$$\sigma_j^2 = k\left(\mathbf{x}_j, \mathbf{x}_j\right) - \mathbf{k}_{I,j}^{\mathrm{T}}\mathbf{K}_{I,I}^{-1}\mathbf{k}_{I,j}.$$

Gradients with respect to $\mathbf{x}_j$ do not depend on other data in J, we can therefore independently optimise the likelihood of each $\mathbf{y}_j$ with respect to corresponding $\mathbf{x}_j$. Thus the full set $\mathbf{X}_J$ can be optimised with one pass through the data. The active set is then reselected, and the process is repeated again.

Algorithm 1 summarises the order in which we implemented these steps. The active set is first selected, then the kernel parameters and active set positions are optimised. The active set is then re-selected and then the latent positions of the points not in the active set are optimised. In each iteration we perform two active set selections because the choice of active set is dependent on both the kernel parameters and the latent point positions. Note also, that for some data sets (when $N >> d$) it may not be necessary to optimise $\mathbf{X}_I$ because the active set is regularly being reselected.

[15] This fixed variance for all output dimensions is a consequence of sharing the same kernel for each output as was discussed in Section 3.2.1.

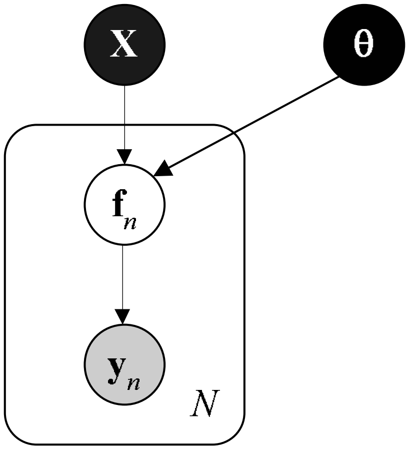


Figure 2: The Gaussian process as a latent variable model.

5. Alternative Noise Models

So far we have considered the GP-LVM for the particular case where we have Gaussian noise in each dimension with variance β^{-1}. In this section we consider extensions to this noise model. To this end we firstly reformulate our Gaussian process so that it contains an additional latent variable $\mathbf{F} = [\mathbf{f}_1 \ldots \mathbf{f}_N]^{\mathbf{T}}$ between $\mathbf{X}$ and $\mathbf{Y}$.

$$p(\mathbf{Y}|\mathbf{X},\theta) = \int \prod_{n=1}^{N} p(\mathbf{y}_n|\mathbf{f}_n)\, p(\mathbf{F}|\mathbf{X},\theta)\, d\mathbf{F}. \tag{13}$$

Thus far we have been considering the case where

$$p(\mathbf{y}_n|\mathbf{f}_n) = \prod_{i=1}^{D} N\left(y_{ni}|\, f_{ni}, \beta^{-1}\right),$$

it is also straightforward to realise the slightly more general case where the variance is dependent on both the data point and the output dimension,

$$p(\mathbf{y}_n|\mathbf{f}_n) = \prod_{i=1}^{D} N\left(y_{ni}|\, f_{ni}, \beta_{ni}^{-1}\right). \tag{14}$$

Our approach to different noise models will be to approximate them with a Gaussian noise model of this form (see also Csató, 2002; Minka, 2001). The noise models we consider in this paper will be independent across the dimensions,

$$p(\mathbf{y}_n|\mathbf{f}_n) = \prod_{i=1}^{D} p(y_{ni}|\, f_{ni}),$$

giving approximations of the form

$$p(y_{ni}|\, f_{ni}) \approx N\left(m_{ni}|\, f_{ni}, \beta_{ni}^{-1}\right).$$

The approximation to the noise model leads to a Gaussian approximation to the posterior distribution,

$$q(\mathbf{F}) \approx p(\mathbf{F}|\mathbf{X}, \mathbf{Y}),$$

where

$$q(\mathbf{F}) = N\left(\mathbf{f}|\bar{\mathbf{f}}, \Sigma\right)$$

where $\mathbf{f}$ is a vector constructed by stacking the columns of $\mathbf{F}$, and $\bar{\mathbf{f}}$ is constructed by stacking the columns of the matrix $\bar{\mathbf{F}} = \left[\bar{\mathbf{f}}_1 \ldots \bar{\mathbf{f}}_N\right]^{\mathrm{T}}$. The covariance matrix has a block diagonal structure[16]

$$\Sigma = \begin{bmatrix} \Sigma_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \Sigma_D \end{bmatrix}.$$

It can be shown (see *e.g.* Csató 2002; Minka 2001) that the parameters of the approximation are given by

$$\beta_{ni} = \frac{\nu_{ni}}{1 - \nu_{ni}\varsigma_{ni}} \tag{15}$$

$$m_{ni} = \frac{g_{ni}}{\nu_{ni}} + \bar{f}_{ni} \tag{16}$$

where ς_{ni} is nth diagonal element of Σ_i, $g_{ni} = \frac{\partial}{\partial f_{ni}} \ln Z_{ni}$ and $\nu_{ni} = g_{ni}^2 - 2\frac{\partial}{\partial \varsigma_{ni}} \ln Z_{ni}$ where

$$Z_{ni} = \int p(y_{ni}|f_{ni})\, q(\mathbf{F})\, d\mathbf{F}. \tag{17}$$

To prevent cluttering our notation we have not indicated that the approximation $q(\mathbf{F})$ is typically formed in a sequential manner: its parameters $\bar{\mathbf{F}}$ and Σ change as data points are incorporated. This approach to approximating the posterior distribution is known as assumed density filtering (see Maybeck, 1979, Chapter 12 and Minka, 2001, Chapter 3) .

6. Missing Values

In many applications attributes are missing for particular data points. The ability to handle these missing values in a principled way is a desirable characteristic of any algorithm. One motivation behind a probabilistic interpretation of PCA was that the resulting algorithm could handle missing data in a principled manner. This is a characteristic which the Gaussian process latent variable model shares. This should be contrasted with kernel PCA where handling missing values is not so straightforward.

Given the formalism we have described for using different noise models it is straightforward to handle a missing attribute. The corresponding variance from (14) is set to infinity by taking $\beta_{ni} = 0$.

[16] For the special case of Gaussian noise with fixed variance β^{-1} (*i.e.* spherical noise) and shared kernels for each data dimension we find that these blocks are all equal. This leads to computational and memory savings. If the kernels are different or more general noise models are used the blocks will not be equal.

7. Results

In this section we present a range of empirical evaluations with different data sets, each explores a different characteristics of the GP-LVM. Note that for these visualisation algorithms over-fitting is *not* a problem as long as the latent-space is of lower dimensionality than the data-space. This is a consequence of the integration over the mapping between the latent and the data-space.

So far we have briefly considered two different kernel/covariance functions, before proceeding further we will reconsider these and introduce further kernels which will be used in the experiments that follow.

7.1 Kernels to Be Used

A Gaussian process covariance function can be developed from any positive definite kernel, new kernels can also be formed by adding kernels together. In our experiments we principally make use of three different kernel functions.

7.1.1 LINEAR KERNEL

We have already briefly discussed the linear kernel, it is simply the matrix of inner products,

$$k_{\text{lin}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \theta_{\text{lin}} \mathbf{x}_i^{\text{T}} \mathbf{x}_j,$$

where we have introduced θ_{lin}, the process variance, which controls the scale of the output functions.

7.1.2 RBF KERNEL

We also made use of the popular RBF kernel, it leads to smooth functions that fall away to zero in regions where there is no data.

$$k_{\text{rbf}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \theta_{\text{rbf}} \exp\left(-\frac{\gamma}{2}\left(\mathbf{x}_i - \mathbf{x}_j\right)^{\text{T}}\left(\mathbf{x}_i - \mathbf{x}_j\right)\right)$$

where γ is the inverse width parameter.

7.1.3 MLP KERNEL

The MLP kernel (Williams, 1997) is derived by considering a multi-layer perceptron (MLP) with an infinite number of hidden units,

$$k_{\text{mlp}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \theta_{\text{mlp}} \sin^{-1}\left(\frac{w\mathbf{x}_i^{\text{T}}\mathbf{x}_j + b}{\sqrt{\left(w\mathbf{x}_i^{\text{T}}\mathbf{x}_i + b + 1\right)\left(w\mathbf{x}_j^{\text{T}}\mathbf{x}_j + b + 1\right)}}\right)$$

where we call w the weight variance and b the bias variance (they have interpretations as the variances of prior distributions in the neural network model). This covariance function also leads to smooth functions, but they have an important characteristic that differentiates them from the RBF kernel: outside regions where the data lies functions will not fall to zero, but tend to remain at the same value.

7.1.4 THE NOISE TERM

In the experiments in Section 3 we also made use of a 'white noise term'. A white noise process has a kernel of the form

$$k_{\text{white}}\left(\mathbf{x}_i, \mathbf{x}_j\right) = \theta_{\text{white}}\delta_{ij}$$

where δ_{ij} is the Kronecker delta which is zero unless $i = j$ when it takes the value 1. Note that the use of white noise in the kernel is often redundant with some parameters in the noise model, for example with a Gaussian noise model, leaving out the white noise term and setting

$$p\left(y_{in}|f_{in}\right) = N\left(y_{in}|f_{in}, \theta_{\text{white}}\right)$$

is equivalent to including the white noise kernel and setting

$$p\left(y_{in}|f_{in}\right) = \lim_{\sigma^2 \to 0} N\left(y_{in}|f_{in}, \sigma^2\right).$$

In our experiments we preferred to include the noise term with the kernel as the noise level, θ_{white}, can then be jointly optimised with the kernel parameters and the latent point positions.

7.1.5 PARAMETER CONSTRAINTS AND INITIALISATION

All the kernels we have mentioned so far have parameters that need to be constrained to be positive. In our experiments this was implemented by reparameterising:

$$\theta = \ln\left(1 + \exp\left(\theta'\right)\right).$$

Note that as our transformed parameter $\theta' \to -\infty$ the parameter $\theta \to 0$ and as $\theta' \to \infty$ we see that $\theta \to \theta'$.

We used a consistent initialisation of the parameters for all experiments. This was $\theta_{\text{lin}} = 1$, $\theta_{\text{rbf}} = 1$, $\gamma = 1$, $\theta_{\text{mlp}} = 1$, $w = 10$ and $b = 10$.

7.2 Overview of Experiments

For the experiments that follow we used Algorithm 1 with $T = 15$ iterations and an active set of size $d = 100$. The experiments were run on a 'one-shot' basis, *i.e.* each experiment was only run once with one setting of the random seed and the values of T and d given.

The remainder of this section is structured as follows, firstly, in Section 7.2.1 we revisit the oil data first introduced in Section 3.2, but with the revised algorithm which allows us to efficiently visualise all the data points. As well as comparing the sparse algorithm to the GTM and PCA we also include a full GP-LVM model. For each of the different algorithms we explore the quality of the visualisation in terms of the ease with which the different flow regimes can be separated in the embedded space. In Section 7.3.1 we turn to a much higher (256) dimension data set of hand-written digits. Again we compare the GTM and PCA with the sparse GP-LVM algorithm by seeing how well the different digits are separated in the latent-space.

In both of the preceding data sets we made use of the Gaussian noise model, our final experiment with this noise model concerns issues with initialisation. In the data sets presented above we have no simple 'ground truth' which the algorithm hopes to recover. In Section 7.2.3 we consider the Swiss-roll data Tenenbaum et al. (2000). For this data the ground truth is known and it turns out that using PCA to initialise the GP-LVM the ground truth is not recovered, however by initialising using

Model	PCA	Sparse GP-LVM (RBF)	GP-LVM (RBF)	Sparse GP-LVM (MLP)	GTM
Errors	162	24	1	14	11

Table 3: Number of errors for nearest neighbour classification in the latent-space for the full oil data set (1000 points).

Isomap (which is known to give the ground truth) we can recover a probabilistic representation of this data.

In Section 7.3.1 we move on to non-Gaussian data sets. We consider a binary data set of handwritten 2s. We compare a binary model with a Gaussian model and show that the binary model is more effective at reconstructing twos when pixels are obscured from the model.

7.2.1 OIL FLOW DATA

In this section we return to the twelve dimensional oil data set that we first introduced in Section 3.2. We now visualise all 1000 of the data points. For this data set we are interested in evaluating two different things: the effect of using the different non-linear kernels and the effect of the sparse GP-LVM algorithm relative to the full model.

In Figure 3(a) and (b) we present visualisations of the data using sparse GP-LVM algorithm with the RBF and MLP kernels respectively. In Figure 4(a) we show the data visualised with the non-sparse GP-LVM algorithm and in Figure 4(b) we have recreated the visualisation in (Bishop et al., 1998) which uses the GTM algorithm.

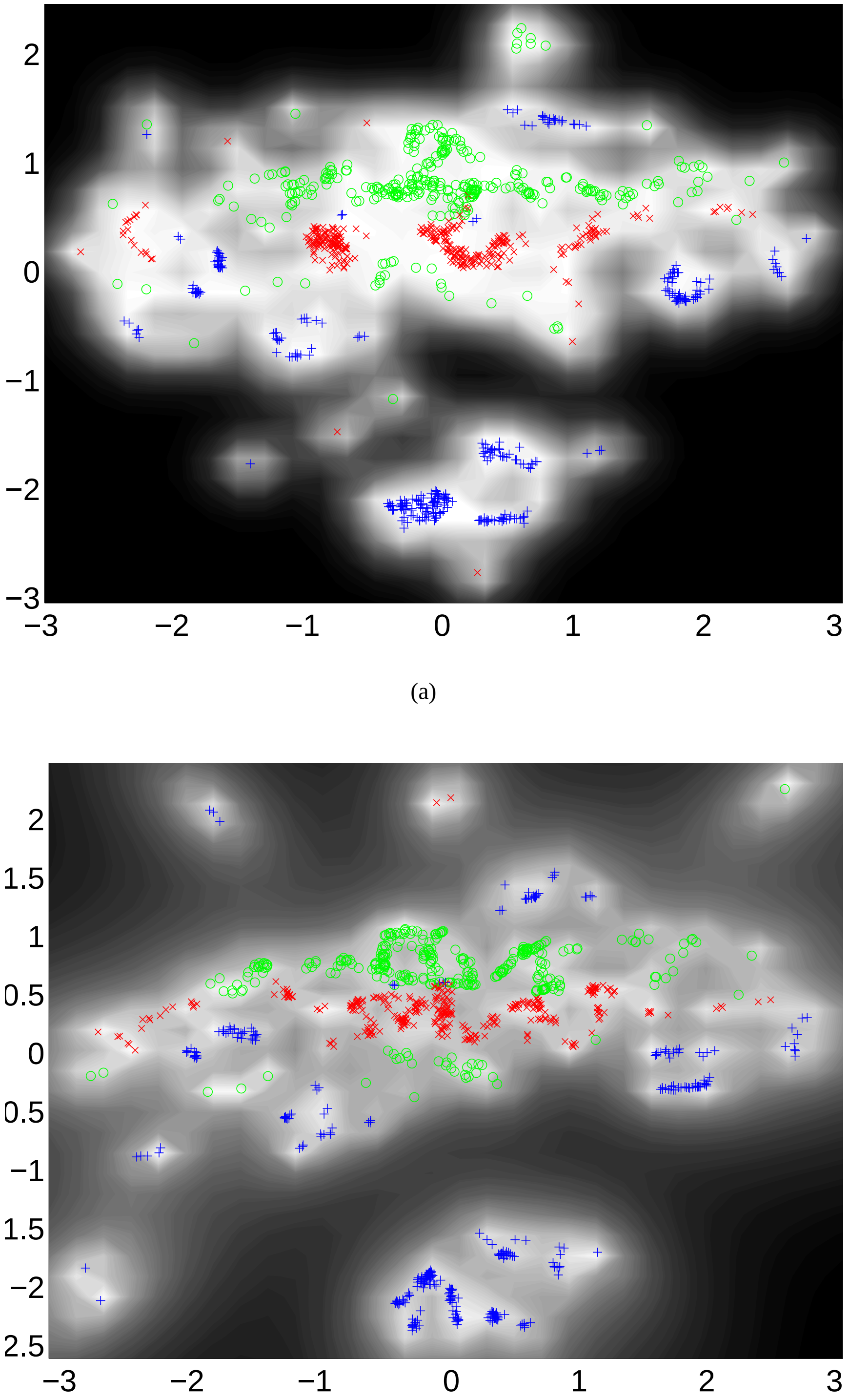
Again we considered a nearest neighbour classifier in the latent-space to quantify the quality of the visualisations.

We note that there appears to be a degradation in the quality of the GP-LVM model associated with the sparsification, in comparision to the full GP-LVM algorithm and the GTM the sparse GP-LVM performs worse.

7.2.2 HANDWRITTEN DIGITS

The oil flow data has twelve attributes, twelve dimensions is too many for the structure of the data set to be visualised without resorting to displaying embedded spaces, but there are many data sets with much greater dimensionality. One popular data set for visualisation algorithms has been handwritten digits. We therefore followed Hinton and Roweis (2003) in our 2-D visualisation of a sub-set of 3000 of the digits 0-4 (600 of each digit) from a 16×16 greyscale version of the USPS digit data set (Figure 5). Again we made use of the RBF and the MLP kernel. As well as visualising with the GP-LVM we present visualisations from a GTM and PCA (Figure 6).

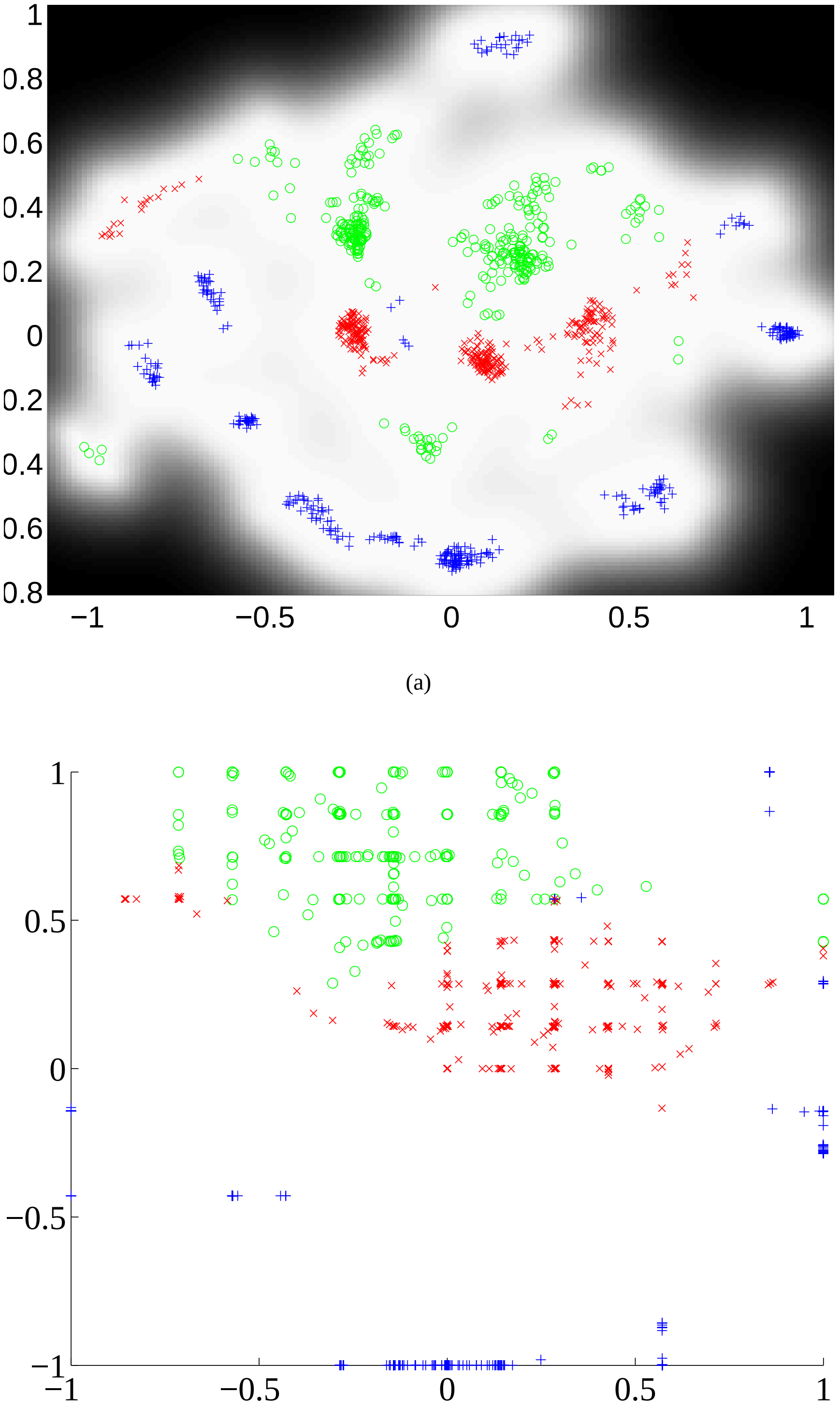
As for the oil data we looked for an objective assessment of the quality of the visualisation by evaluation errors on a nearest neighbour classifier in the latent-space. The performance benefits associated with the non-linear visualisations are more apparent here than they were for the oil data (Table 4). The sparse GP-LVM is once again outperformed by the GTM algorithm under this criterion. Comparision with the full GP-LVM model for this data set is not currently practical.



(a)

(b)

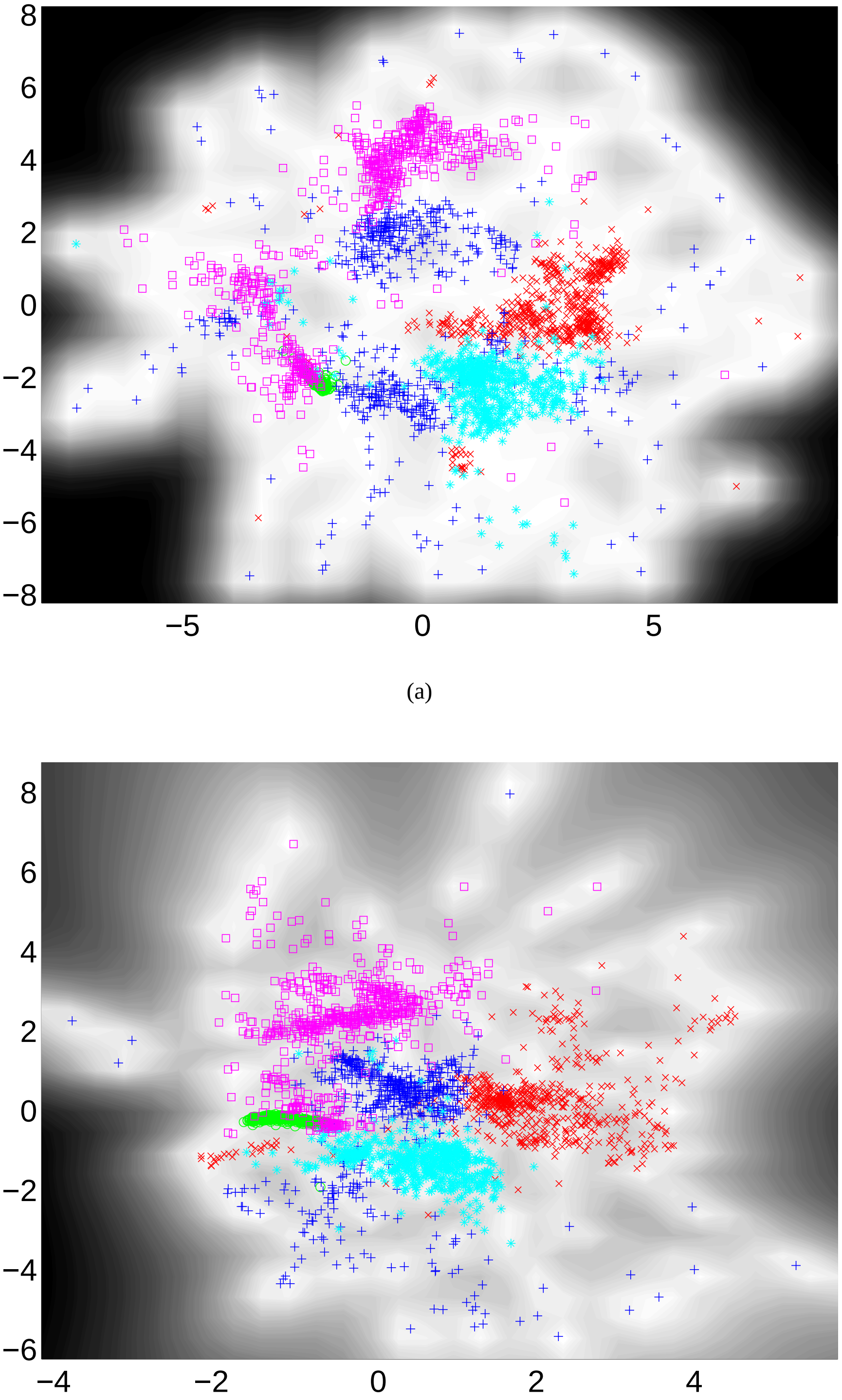
Figure 3: The full oil flow data set visualised with (a) an RBF based sparse GP-LVM, (b) an MLP based sparse GP-LVM.



(a)

(b)

Figure 4: (a) The full GP-LVM algorithm with RBF kernel on the oil flow data. (b) GTM with 225 latent points laid out on a 15×15 grid and with 16 RBF nodes.



(a)

(b)

Figure 5: The digit images visualised in the 2-D latent-space. ‘0’ is represented by red crosses; ‘1’: green circles; ‘2’: blue pluses; ‘3’: cyan stars and ‘4’: magenta squares. (a) Visualisation using an RBF kernel. (b) Visualisation using an MLP kernel.

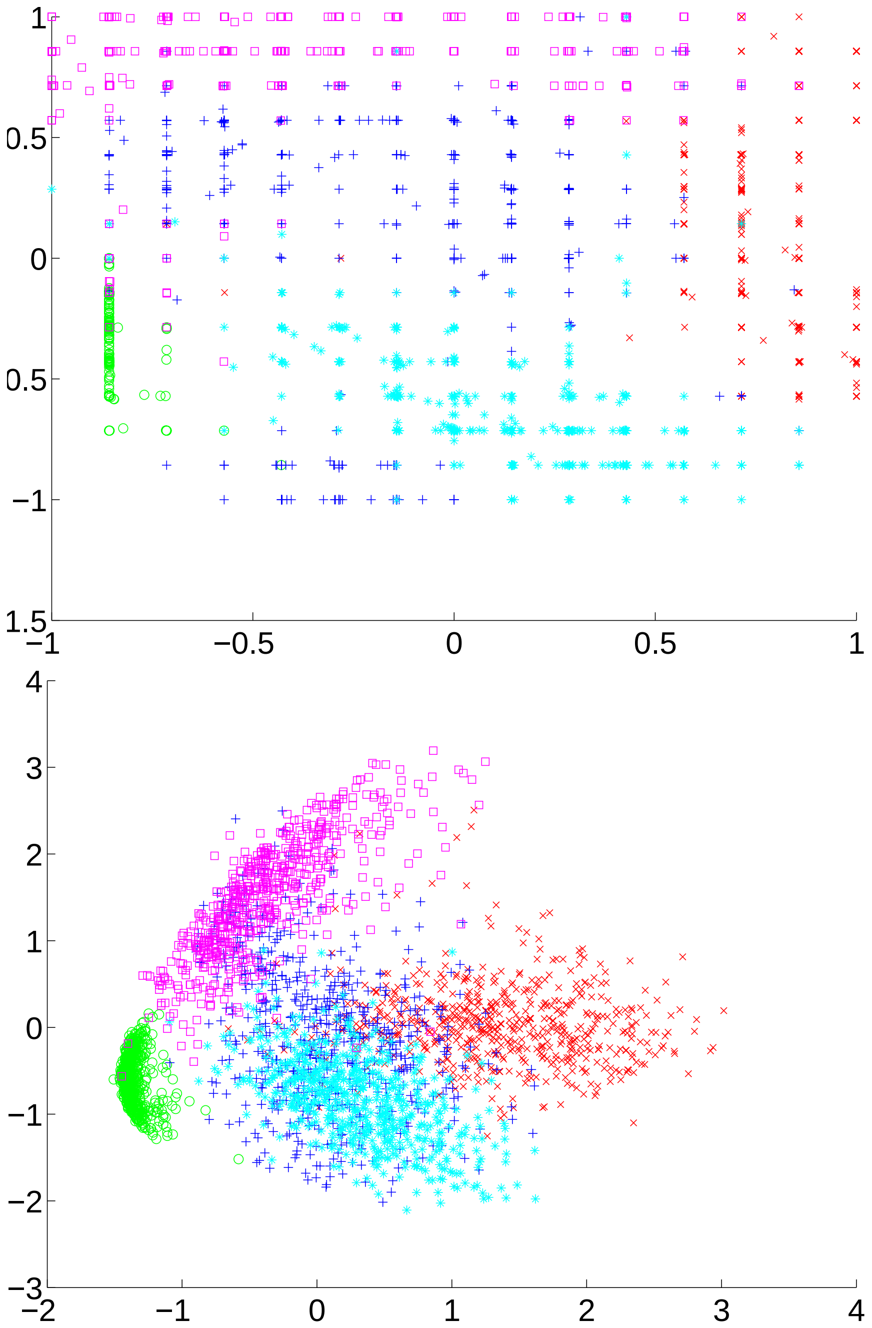


Figure 6: The digit images visualised in the 2-D latent-space. ‘0’ are red crosses, ‘1’ are green circles, ‘2’ are blue pluses, ‘3’ are cyan stars and ‘4’ are magenta squares. (a) Visualisation using the GTM algorithm. (b) Visualisation using PCA.

Model	PCA	Sparse GP-LVM (RBF)	Sparse GP-LVM (MLP)	GTM
Errors	780	208	202	158

Table 4: Errors for nearest neighbour classification in the latent-space for the digit data.

7.2.3 Initialisation of the Model

In the experiments we described above PCA was used to initialise the positions of the points in latent-space, however, there are data sets for which PCA can provide a poor initialisation, causing the GP-LVM to become caught in a local minima. In Figure 7(a) we show a result from modelling the ‘Swiss-roll’ data set (Tenenbaum et al., 2000, data available on line). For this data the true structure is known—the manifold is a two dimensional square twisted into a spiral along one of its dimensions and living in a three dimensional space. We follow Roweis and Saul (2000) in using colour to show the position along the sheet.

When the GP-LVM is initialised with PCA it becomes stuck in an optimum that does not recover the true embedded space. However, by initialising using the Isomap algorithm, we are able to recover the underlying structure and then provide a probabilistic description of the data through the GP-LVM (Figure 7(b)). In this way we can combine the strengths of the two different approaches—Isomap (and related proximity data based algorithms) provide a unique solution which can recover the structure of the manifold on which the data lies, the GP-LVM provides an underlying probabilistic model and an easy way to compute the mapping from the latent to the observed space. Due to the probabilistic nature of the GP-LVM we can also compare the resulting models through their log likelihood. The log likelihood of the Isomap initialised model (-45.19) is over a factor of ten smaller than that of the PCA initialised model (-534.0) further demonstrating the advantage of the Isomap initialisation for this data set.

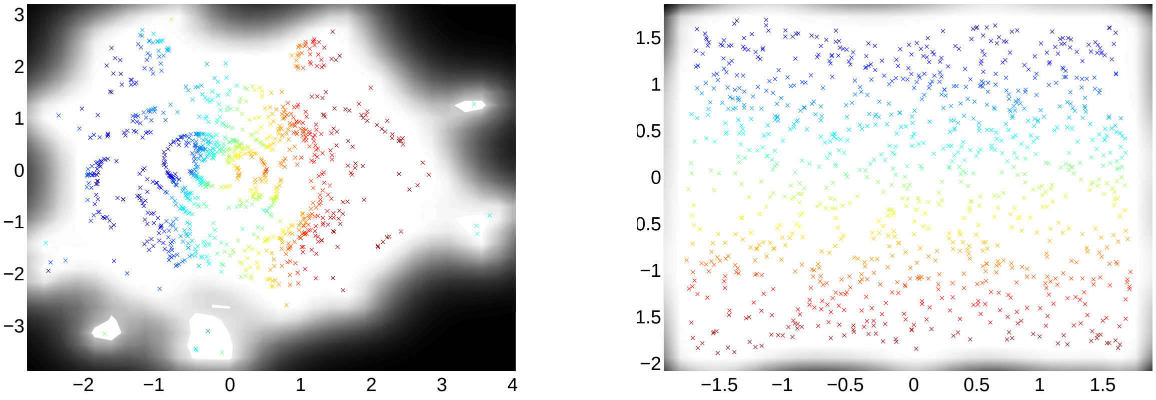
7.3 Missing Data and Non-Gaussian Noise Models

The examples we have presented so far are for Gaussian noise models. In cases where the data is not continuous a Gaussian noise model is no longer appropriate. Non-Gaussian, *linear*, latent trait models have already been proposed (Bartholomew, 1987; Tipping, 1999), in this section we use the ADF approach described in Section 5 to explore two non-Gaussian data sets with GP-LVM models based around non-Gaussian noise models.

7.3.1 Visualisation of Binary Data

In our first example we follow Tipping (1999) in visualising binary handwritten twos. In Figure 8 we show visualisations from an 8×8 data set derived from the USPS Cedar CD-ROM. The data contains 700 examples, these examples were taken from the complete data set of all digits used in Hinton et al. (1995). For both visualisations an RBF kernel was used in combination with a Gaussian prior over the latent-space, however the two visualisations make use of different noise models. In Figure 8(a) a Gaussian noise model was used, in Figure 8(b) a Bernoulli noise model was used.

There are certainly differences between the two visualisations in Figure 8, however we again wish to make an objective assessment of the qualities of the embedded spaces. To this end, we turned to a test set containing 400 hundred digits. For each digit in the test set we removed 20% of the pixel values. The digit was then presented to the model and its position in the embedded space



(a)

(b)

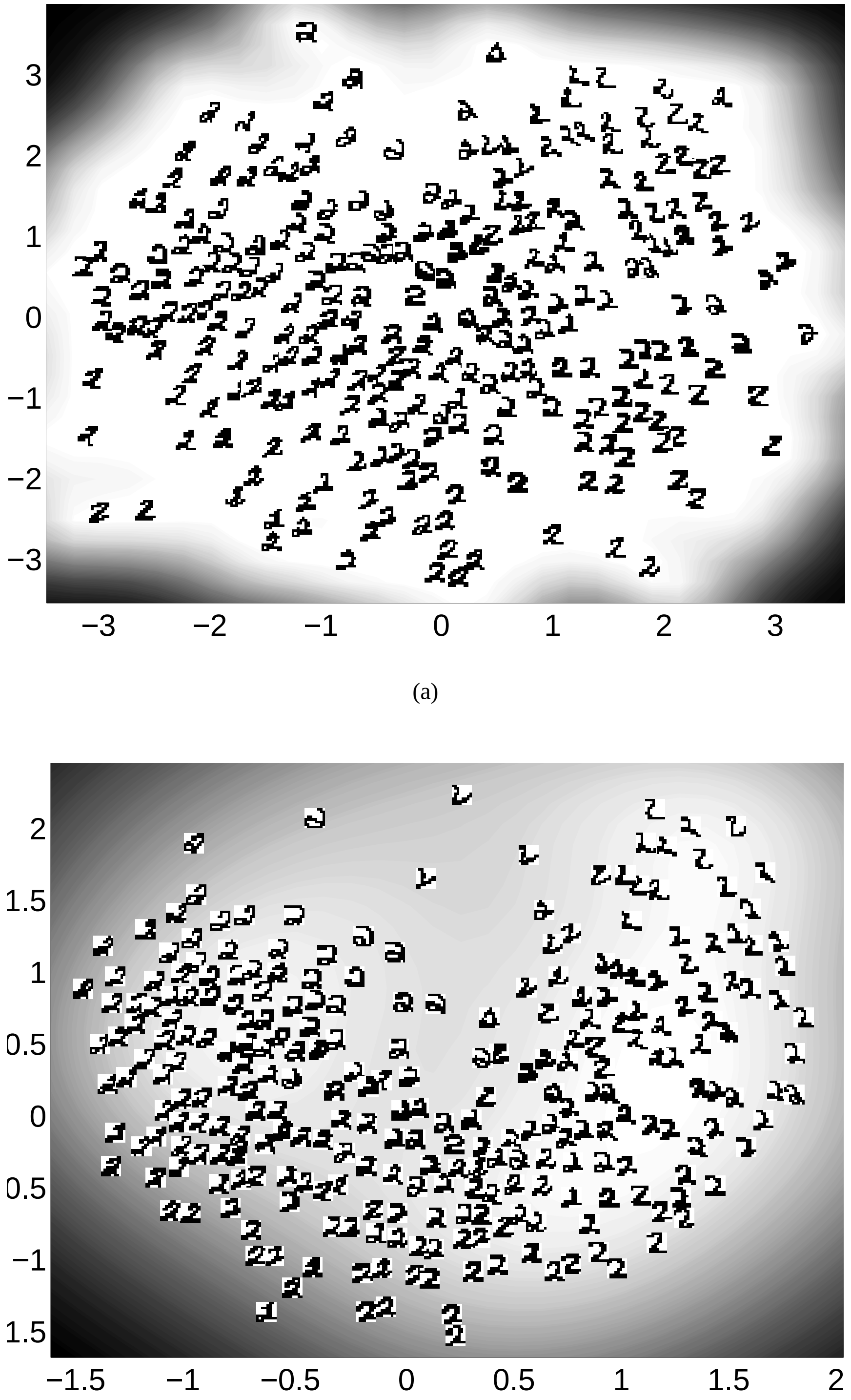
Figure 7: The effect of a poor initialisation. (a) GP-LVM initialised using PCA. The log-likelihood of the resulting model was -534.0 (b) GP-LVM initialised using Isomap. The log likelihood of the resulting model was -45.19.

Reconstruction method	pixel error rate
GP-LVM with Bernoulli noise	23.5%
GP-LVM with Gaussian noise	35.9%
Assume pixels are ‘not ink’	51.5%

Table 5: Pixel reconstruction error rates.

optimised. The missing pixels were then filled in by using the mapping from the embedded to the data-space. Note that there can be local minima in the embedded space, we therefore optimised the embedded space location ten times with different starting positions and selected that with the largest likelihood. Since we know the original pixel values we can compute the pixel reconstruction error rate. These rates are summarised in Table 5. Results are shown for the Bernoulli noise model, the Gaussian noise model and a baseline approach (which is simply to assume that the missing pixels do not contain ink).

As might be hoped, both approaches considerably outperform the baseline approach. We also note that using the Bernoulli noise model leads to far better results than the Gaussian noise model. To illustrate the type of mistakes that are made we show some randomly sampled results in Figure 9. For each test digit we present: the original digit, an image showing which pixels are removed and reconstruction using the three methods outlined above. Note that for the GP-LVM reconstructions, particularly for the Bernoulli noise model, even when mistakes are made the resulting image often still looks like a handwritten 2.



(a)

(b)

Figure 8: The two images visualised in the 2-D latent-space. (a) Visualisation using an Gaussian noise model. (b) Visualisation using a Bernoulli noise model.

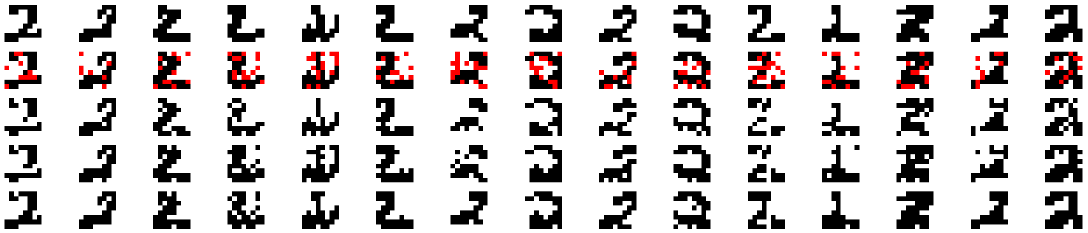


Figure 9: Randomly sampled examples from the test data for the 'twos' problem. *Top row*: test images from the data set of twos, *second row*: pixels removed from the test images are shown in red, *third row*: reconstruction which assumes missing pixels are 'not ink', *fourth row*: reconstruction by the Gaussian GP-LVM, *fifth row*: reconstruction by the binary noise model.

8. Discussion

We have presented the Gaussian process latent variable model, which is a non-linear probabilistic extension of PCA. Our experiments show that the GP-LVM is a viable alternative to other non-linear visualisation approaches for small data sets. We reviewed a practical algorithm for fitting the GP-LVM (Lawrence, 2004) in large data sets, but noted that it is associated with a degradation in performance of the method. The GP-LVM model was extended in a principled manner to take account of missing data and binary data. The advantage of explicitly modelling the data type was shown by a missing data problem in handwritten digits.

8.1 Computing the Likelihood of Test Data

One key advantage of the GP-LVM is that it is probabilistic. There is a likelihood associated with the training data. The model can be viewed as a non-parametric density estimator: the size of $\mathbf{X}$ grows proportionally with the size of $\mathbf{Y}$. However this introduces particular problems when we are interested in computing the likelihood of a previously unseen (test) data point. In the traditional probabilistic PCA model when a new data point, $\mathbf{y}_*$, is presented its likelihood under the marginal distribution,

$$p\left(\mathbf{y}_*|\mathbf{W},\beta\right) = N\left(\mathbf{y}_*|\mathbf{0},\mathbf{W}\mathbf{W}^{\mathrm{T}}+\beta^{-1}\mathbf{I}\right), \tag{18}$$

is easily computed. Therefore the likelihood of a previously unseen test data set is straightforward to compute. In the GP-LVM the likelihood takes a different form. The new datum has an associated latent variable, $\mathbf{x}_*$. The likelihood of $\mathbf{y}_*$, for the special case where variances over each output direction are constant, is given by

$$p\left(\mathbf{y}_*|\mathbf{X},\mathbf{x}_*\right) = N\left(\mathbf{y}_*|\mu,\sigma^2\right), \tag{19}$$

where

$$\mu = \mathbf{Y}^{\mathrm{T}}\mathbf{K}_{I,I}^{-1}\mathbf{k}_{I,*}, \tag{20}$$

$\mathbf{k}_{I,*}$ being a column vector developed from computing the elements of the kernel matrix between the active set and the new point $\mathbf{x}_*$. The variance is then given by

$$\sigma^2 = k\left(\mathbf{x}_*,\mathbf{x}_*\right) - \mathbf{k}_{I,*}^{\mathrm{T}}\mathbf{K}_{I,I}^{-1}\mathbf{k}_{I,*}. \tag{21}$$

To determine the likelihood of the new point, we first find the MAP solution for this new latent point. The likelihood could then approximated by computing the probability of the observed data under the distribution given by projecting the MAP solution for $\mathbf{x}_*$ back into data-space. However, since the posterior over $\mathbf{X}$ can be multi-modal with respect to $\mathbf{x}_*$, this solution will not necessarily be unique. In an ideal world, we would integrate out the latent-space to determine this marginal likelihood, and the problem with multiple modes would not arise. In practice it may be necessary to seek several modes by random restarts within the latent-space, if the likelihood is strongly peaked around each of these modes and there is a large difference between the magnitude of the two largest modes it is enough to approximate the solution with the largest mode. In other cases it may be necessary to turn to sampling methods to evaluate the likelihood.

9. Conclusions

We have presented a new class of models for probabilistic modelling and visualisation of high dimensional data. We provided theoretical groundings for these models by proving that principal component analysis is a special case. We showed there is a general objective function based on the Kullback-Leibler divergence that connects these models with proximity data based methods such as kernel PCA and multidimensional scaling. Further analysis of this objective function is expected to provide deeper insights into the behaviour of these algorithms. On real world data sets we showed that visualisations provided by the model placed related data points close to each other. We demonstrated empirically that the model performed well in traditionally difficult domains that involve missing and discrete data in high dimensions.

Our approach is related to density networks and the generative topographic mapping in that these models all provide a non-linear mapping from the embedded space to the observed space. In all these cases the embedded space is treated as a latent variable and problems of propagating distributions through the non-linear mapping are avoided by using point representations of the data within the latent space. A novel characteristic of the GP-LVM is that we can visualise the uncertainty with which the manifold is defined in the data-space.

Acknowledgments

We thank Aaron Hertzmann and his collaborators for ongoing access to their work on style based inverse kinematics, Amos Storkey for pointing out that the GP-LVM fails on the Swiss-roll data with a PCA initialisation and Michael Tipping for discussions on visualisation techniques.

Appendix A. Probabilistic Interpretations of PCA

The standard probabilistic interpretation of PCA (Tipping and Bishop, 1999) involves a likelihood,

$$p\left(\mathbf{Y}|\mathbf{W},\mathbf{X},\beta\right) = \prod_{n=1}^{N} p\left(\mathbf{y}_n|\mathbf{W},\mathbf{x}_n,\beta\right)$$

which is taken to be Gaussian,

$$p\left(\mathbf{y}_n|\mathbf{W},\mathbf{x}_n,\beta\right) = N\left(\mathbf{y}_n|\mathbf{W}\mathbf{x}_n,\beta^{-1}\mathbf{I}\right),$$

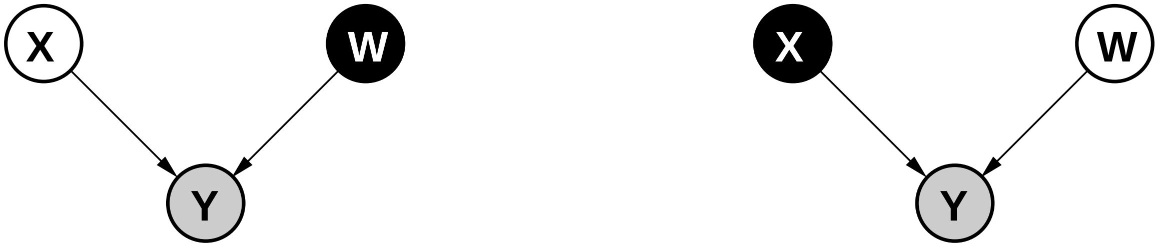


Figure 10: Graphical representation of (a) the standard probabilistic PCA model and (b) its dual representation which also leads to a probabilistic interpretation of PCA. The nodes are shaded to represent different treatments. *Black* shaded nodes are optimised, *white* shaded nodes are marginalised and *grey* shaded nodes are observed variables.

the prior distribution for the latent variables is then taken to be Gaussian,

$$p\left(\mathbf{x}_n\right) = N\left(\mathbf{x}_n|\mathbf{0},\mathbf{I}\right),$$

and is duly marginalised to recover the marginal likelihood for the data,

$$p\left(\mathbf{Y}|\mathbf{W},\beta\right) = \prod_{n=1}^{N} p\left(\mathbf{y}_n|\mathbf{W},\beta\right), \tag{22}$$

where

$$p\left(\mathbf{y}_n|\mathbf{W},\beta\right) = N\left(\mathbf{y}_n|\mathbf{0},\mathbf{W}\mathbf{W}^{\mathrm{T}}+\beta^{-1}\mathbf{I}\right). \tag{23}$$

The structure of this model is shown graphically in Figure 10(a).

The dual representation of probabilistic PCA involves integrating out **W** and maximising with respect to $\mathbf{x}_n$

$$p\left(\mathbf{Y}|\mathbf{X},\beta\right) = \int \prod_{n=1}^{N} p\left(\mathbf{y}_n|\mathbf{x}_n,\mathbf{W},\beta\right) p\left(\mathbf{W}\right) d\mathbf{W}.$$

By first specifying a prior distribution,

$$p\left(\mathbf{W}\right) = \prod_{i} N\left(\mathbf{w}_i|\mathbf{0},\mathbf{I}\right)$$

where $\mathbf{w}_i$ is the ith row of the matrix **W**, and then integrating over **W** we obtain a marginalised likelihood for **Y**,

$$p\left(\mathbf{Y}|\mathbf{X},\beta\right) = \frac{1}{(2\pi)^{\frac{DN}{2}}|\mathbf{K}|^{\frac{D}{2}}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right)\right), \tag{24}$$

where $\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}}+\beta^{-1}\mathbf{I}$ and $\mathbf{X} = \left[\mathbf{x}_1^{\mathrm{T}} \ldots \mathbf{x}_N^{\mathrm{T}}\right]^{\mathrm{T}}$. The structure of this model is shown in 10(b). Note that by taking $\mathbf{C} = \mathbf{W}\mathbf{W}^{\mathrm{T}}+\beta^{-1}\mathbf{I}$ we and substituting (23) into (22) as

$$p\left(\mathbf{Y}|\mathbf{X},\beta\right) = \frac{1}{(2\pi)^{\frac{DN}{2}}|\mathbf{C}|^{\frac{N}{2}}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{C}^{-1}\mathbf{Y}^{\mathrm{T}}\mathbf{Y}\right)\right),$$

which highlights to a greater extent the duality between (24) and (22). Optimisation of (24) is clearly highly related to optimisation of (22). Tipping and Bishop (1999) showed how to optimise (22), in the next section we review this optimisation for DPPCA, but generalise it slightly so that it applies for any symmetric matrix $\mathbf{S}$, rather than only the inner product matrix $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$. Thereby the derivation also covers the kernel PCA and multidimensional scaling cases outlined in Section 2.6.

Appendix B. Optimisation of Dual PCA, KPCA and MDS Objective functions

Maximising (24) is equivalent to minimising

$$L = \frac{N}{2}\ln 2\pi + \frac{1}{2}\ln|\mathbf{K}| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{S}\right), \tag{25}$$

where $\mathbf{S} = D^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$. The derivation that follows holds regardless of the form of $\mathbf{S}$ and therefore also applies to the objective function outlined in Section 2.6. However, $\mathbf{S}$ needn't be constrained to this form, we outlined an objective function (for kernel PCA) in where $\mathbf{S}$ was any positive definite kernel.

The gradient of the likelihood with respect to $\mathbf{X}$ can be found as

$$\frac{\partial L}{\partial \mathbf{X}} = -\mathbf{K}^{-1}\mathbf{S}\mathbf{K}^{-1}\mathbf{X} + \mathbf{K}^{-1}\mathbf{X},$$

setting the equation to zero and pre-multiplying by $\mathbf{K}$ gives

$$\mathbf{S}\left[\beta^{-1}\mathbf{I} + \mathbf{X}\mathbf{X}^{\mathrm{T}}\right]^{-1}\mathbf{X} = \mathbf{X}.$$

We substitute $\mathbf{X}$ with its singular value decomposition, $\mathbf{X} = \mathbf{U}\mathbf{L}\mathbf{V}^{\mathrm{T}}$, giving

$$\mathbf{S}\mathbf{U}\left[\mathbf{L} + \beta^{-1}\mathbf{L}^{-1}\right]^{-1}\mathbf{V}^{\mathrm{T}} = \mathbf{U}\mathbf{L}\mathbf{V}^{\mathrm{T}}$$

Right multiplying both sides by $\mathbf{V}$ (note that the solution is invariant to $\mathbf{V}$) we have, after some rearrangement,

$$\mathbf{S}\mathbf{U} = \mathbf{U}\left(\beta^{-1}\mathbf{I} + \mathbf{L}^{2}\right),$$

which, since $\left(\beta^{-1}\mathbf{I} + \mathbf{L}^{2}\right)$ is diagonal can be solved by an eigenvalue problem where $\mathbf{U}$ are eigenvectors of $\mathbf{S}$ and $\Lambda = \left(\beta^{-1}\mathbf{I} + \mathbf{L}^{2}\right)$ are the eigenvalues. This implies that the elements from the diagonal of $\mathbf{L}$ are given by

$$l_i = \left(\lambda_i - \beta^{-1}\right)^{\frac{1}{2}}. \tag{26}$$

B.1 The Retained Eigenvalues

The natural follow up question is which of the N possible eigenvalues/vector pairs should be retained? For convenience let us ignore our previously defined ordering of the eigenvalues in terms of their magnitude and assume that we keep the first q eigenvalues.

First note that

$$\mathbf{K} = \mathbf{U}\left[\mathbf{L}^{\mathbf{2}} + \beta^{-1}\mathbf{I}\right]\mathbf{U}^{\mathrm{T}}$$

where $\mathbf{U}$ is all the eigenvectors of $\mathbf{S}$. The full KL divergence is

$$\begin{aligned}
\mathrm{KL}\left(\mathbf{S}||\mathbf{K}\right) &= \frac{1}{2}\ln|\mathbf{K}| - \frac{1}{2}\ln|\mathbf{S}| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{S}\right) - \frac{N}{2} \\
&= \frac{1}{2}\sum_{i=1}^{q}\ln\lambda_i - \frac{N-q}{2}\ln\beta - \frac{1}{2}\sum_{i=1}^{N}\ln\lambda_i + \frac{1}{2}\mathrm{tr}\left(\left[\mathbf{L}^2+\beta^{-1}\mathbf{I}\right]^{-1}\Lambda\right) \\
&= -\frac{1}{2}\sum_{i=q+1}^{N}\ln\lambda_i - \frac{N-q}{2}\ln\beta - \frac{N-q}{2} + \frac{\beta}{2}\sum_{i=q+1}^{N}\lambda_i
\end{aligned}$$

where we have used the fact that $\mathbf{S} = \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}$. Differentiating with respect to β and setting the result to zero to obtain a fixed point equation then gives

$$\beta = \frac{N-q}{\sum_{i=q+1}^{N}\lambda_i}$$

which when substituted back leads to

$$\mathrm{KL}\left(\mathbf{S}||\mathbf{K}\right) = \frac{N-q}{2}\left(\ln\frac{\sum_{i=q+1}^{N}\lambda_i}{N-q} - \frac{1}{N-q}\sum_{i=q+1}^{N}\ln\lambda_i\right), \tag{27}$$

which is recognised as the difference between the log ratio of the arithmetic and geometric means of the discarded eigenvalues. This difference will be zero if and only if the discarded eigenvalues are constant (when the arithmetic and geometric means become equal) otherwise it is positive. The difference is minimised by ensuring that the eigenvalues we discard are adjacent to each other in terms of magnitude.

Which eigenvalues should we then discard? From (26) we note that the retained eigenvalues must be larger than β, otherwise l_i will be complex. The only way this can be true is if we discard the smallest $N-q$ eigenvalues, as retaining any others would force at least one eigenvalue of $\mathbf{X}$ to be negative.

Appendix C. Equivalence of Eigenvalue Problems

In this section we review the equivalence of the eigenvalue problems associated with DPPCA and PPCA. For DPPCA the eigenvalue problem is of the form

$$\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{U} = \mathbf{U}\Lambda.$$

Premultiplying by $\mathbf{Y}^{\mathrm{T}}$ then gives

$$\mathbf{Y}^{\mathrm{T}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{U} = \mathbf{Y}^{\mathrm{T}}\mathbf{U}\Lambda \tag{28}$$

Since the $\mathbf{U}$ are the eigenvectors of $\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$ (see the previous section) the matrix $\mathbf{U}^{\mathrm{T}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{U} = \Lambda$, therefore matrix $\mathbf{U}' = \mathbf{Y}^{\mathrm{T}}\mathbf{U}\Lambda^{-\frac{1}{2}}$ is orthonormal. Post multiplying both sides of (28) by $\Lambda^{-\frac{1}{2}}$ gives

$$\mathbf{Y}^{\mathrm{T}}\mathbf{Y}\mathbf{U}' = \mathbf{U}'\Lambda$$

which is recognised as the form of the eigenvalue problem associated with PPCA, where the eigenvectors of $\mathbf{Y}^{\mathrm{T}}\mathbf{Y}$ are given by $\mathbf{U}' = \mathbf{Y}^{\mathrm{T}}\mathbf{U}\Lambda^{-\frac{1}{2}}$ and the eigenvalues are given by Λ (as they were for DPPCA).

References

David J. Bartholomew. *Latent Variable Models and Factor Analysis*. Charles Griffin & Co. Ltd, London, 1987.

Alexander Basilevsky. *Statistical Factor Analysis and Related Methods*. Wiley, New York, 1994.

Christopher M. Bishop. Bayesian PCA. In Michael J. Kearns, Sara A. Solla, and David A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 482–388, Cambridge, MA, 1999. MIT Press.

Christopher M. Bishop and Gwilym D. James. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research*, A327:580–593, 1993.

Christopher M. Bishop, Marcus Svensén, and Christopher K. I. Williams. A fast EM algorithm for latent variable density models. In D. S. Touretzky, Michael C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 465–471. MIT Press, 1996.

Christopher M. Bishop, Marcus Svensén, and Christopher K. I. Williams. GTM: a principled alternative to the Self-Organizing Map. In *Advances in Neural Information Processing Systems*, volume 9, pages 354–360. MIT Press, 1997.

Christopher M. Bishop, Marcus Svensén, and Christopher K. I. Williams. GTM: the Generative Topographic Mapping. *Neural Computation*, 10(1):215–234, 1998.

Lehel Csató. *Gaussian Processes — Iterative Sparse Approximations*. PhD thesis, Aston University, 2002.

Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popovic. Style-based inverse kinematics. In *ACM Transactions on Graphics (SIGGRAPH 2004)*, 2004.

Geoffrey E. Hinton, Peter Dayan, Brendan J. Frey, and Radford M. Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161, 1995.

Geoffrey E. Hinton and Sam T. Roweis. Stochastic neighbor embedding. In Sue Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 857–864, Cambridge, MA, 2003. MIT Press.

Antti Honkela and Harri Valpola. Unsupervised variational Bayesian learning of nonlinear models. In Lawrence Saul, Yair Weiss, and Léon Bouttou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 593–600, Cambridge, MA, 2005. MIT Press.

Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

Joseph B. Kruskal. Multidimensional scaling by optimizing goodness-of-fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–28, 1964.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

Neil D. Lawrence. Gaussian process models for visualisation of high dimensional data. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 329–336, Cambridge, MA, 2004. MIT Press.

Neil D. Lawrence, Matthias Seeger, and Ralf Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In Sue Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 625–632, Cambridge, MA, 2003. MIT Press.

David Lowe and Michael E. Tipping. Feed-forward neural networks and topographic mappings for exploratory data analysis. *Neural Computing and Applications*, 4(83), 1996.

David B. MacKay and J. L. Zinnes. A probabilistic model for the multidimensional scaling of proximity and preference data. *Marketing Sciences*, 5:325–334, 1986.

David J. C. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, A*, 354(1):73–80, 1995.

Jan R. Magnus and Heinz Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley and Sons, Chichester, West Sussex, 2nd edition, 1999.

Kantilal V. Mardia, John T. Kent, and John M. Bibby. *Multivariate analysis*. Academic Press, London, 1979.

Peter S. Maybeck. *Stochastic Models, Estimation and Control, Volume 1*, volume 141 of *Mathematics in Science and Engineering*. Academic Press, New York, NY, 1979. ISBN 0-12-4807011.

Thomas P. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

Martin F. Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4):525–533, 1993.

Man-Suk Oh and Adrian E. Raftery. Bayesian multidimensional scaling and choice of dimension. *Journal of the American Statistical Association*, 96:1031–1044, 2001.

Anthony O'Hagan. Some Bayesian numerical analysis. In José M. Bernardo, James O. Berger, A. Phillip Dawid, and Adrian F. M. Smith, editors, *Bayesian Statistics 4*, pages 345–363, Valencia, 1992. Oxford University Press.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

John W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, C-18(5):401–409, 1969.

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

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2001.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

Joshua B. Tenenbaum, Virginia de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Michael E. Tipping. *Topographic Mappings and Feed-Forward Neural Networks*. PhD thesis, Aston University, Aston Street, Birmingham B4 7ET, U.K., 1996.

Michael E. Tipping. Probabilistic visualisation of high-dimensional binary data. In Michael J. Kearns, Sara A. Solla, and David A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 592–598, Cambridge, MA, 1999. MIT Press.

Michael E. Tipping. Sparse kernel principal component analysis. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 633–639, Cambridge, MA, 2001. MIT Press.

Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 6(3):611–622, 1999.

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

Christopher K. I. Williams. Computing with infinite networks. In Michael C. Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, Cambridge, MA, 1997. MIT Press.

Christopher K. I. Williams. Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In Michael I. Jordan, editor, *Learning in Graphical Models*, volume 89 of *Series D: Behavioural and Social Sciences*, Dordrecht, The Netherlands, 1998. Kluwer.

Christopher K. I. Williams. On a connection between kernel PCA and metric multidimensional scaling. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 675–681, Cambridge, MA, 2001. MIT Press.