Dimensionality Reduction in the Emulator Setting

D. M. Maniyar, D. Cornford, and A. Boukouvalas

Neural Computing Research Group
Aston University
Aston Street, Birmingham B4 7EG, UK
Tel. +44-121-204-3652, Fax. +44-121-204-3685
Email: maniyard, d.cornford, boukouva @aston.ac.uk

Abstract

Dimensionality reduction is utilised in a range of research domains. Put simply, dimensionality reduction aims to reduce the dimension (number of variables) in a given data set, while retaining some properties of the original complete data set. Dimensionality reduction techniques have not been widely applied in an emulator setting, although some recent work does consider the application of principal components analysis to the output variables of emulator models. Traditionally, dimensionality reduction is used for visualisation, with the aim often being to determine a 2D representation of the complete high dimensional data set, and as a preprocessing step to inference in a range of models. Inference methods often work better, for a fixed number of samples, in terms of both computational efficiency and their ability to consistently estimate parameters in the models, with smaller numbers of variables, and thus typically a smaller number of parameters that need to be estimated. The aim of this report is to provide an overview of dimension reduction methods that might be applicable in the emulator setting. This is assisted by the creation of a decision tree which a modeller can employ to assess the suitability of dimensionality reduction techniques for a given problem. We present some results on a series of toy model settings aimed at illustrating the ability of the various methods to capture the intrinsic dimensionality of the model, and then assess their use in an emulator setting.

Technical Report NCRG/2007/005

October 2007

1 Introduction

One of the central aims of the MUCM project is to assess, and improve on, our ability to deal with complex, *high-dimensional* models in an emulation setting. In this context, the *high dimensional* means that these models might have many inputs and / or produce many outputs. One of the basic building blocks of the MUCM project is the use of *emulators*, which provide a statistical approximation of a *simulator*. The *simulator* is typically, but not exclusively, a deterministic piece of computer code, that represents at some level of abstraction a real system being modelled. The simulator can in this context be considered an input-output mapping, which the emulator approximates probabilistically. Typically, a Gaussian process methodology is used to represent such an *emulator*. The Gaussian process methodology becomes more computer-intensive as the dimensionality of the input and output spaces increase, in particular the methodology does not scale well in the number of input-output pairs (we call these training points). The major goal of Theme 1 of the MUCM project, which is undertaken at Aston University and Durham University, is to deal with high-dimensionality issues in the input and output spaces, as well as addressing the scaling with the number of training points. The aim is to develop solutions together with a decision tree which will help modellers who wish to employ emulator methods, to deal with some of the largest and most challenging models in areas such as environmental science, protein modelling and particle physics. Within this theme, the work at Aston is divided into two work-packages; WP 1.1 - tackles screening issues and WP 1.2 - deals with sparsity and the use projection methods.

This report considers dimension reduction in the input space. Screening methods are widely applied in simulator / emulator setting under the umbrella of sensitivity analysis (Saltelli et al., 2000). In screening the interpretation of the results is comparatively straightforward as one typically screens out redundant variables. This is a form of dimension reduction in the input variables, but in this report we consider projective based dimension reduction. Interpretation becomes difficult in the projective dimension reduction case, since the resulting reduced dimensions are linear / non-linear combinations of the input variables and in some cases this mapping might not be invertible.

This technical report provides a discussion of dimensionality reduction approaches, and assess how these might be applied in the emulator setting. Our aim is to develop a decision tree which can be utilised by a modeller to take effective decisions about which dimensionality reduction techniques to apply to the problem at hand. The focus of this report is discussion on dimension reduction in general, excluding the screening methods. Readers interested in screening methods in emulator setting are directed to a separate technical report on the work carried out at Aston on the screening part by Boukouvalas et al. (2007).

In the next section we state the motivation of applying dimension reduction prior to emulation. Section 3 presents the decision tree which provides a discussion on which dimension reduction techniques to use in a given situation. We review some of the popular unsupervised dimension reduction methods and supervised dimension reduction methods in Section 4 and 5 respectively. Results on two toy datasets are reported and discussed in Section 6. In Section 7 some open questions are listed before the summary of the report in Section 8.

2 Why attempt dimension reduction prior to emulation?

In existing applications emulation methods have been applied to a range of relatively low dimensional problems, with input dimensions up to $O(100)$[1] and outputs typically of dimension 1, but up to similar orders of magnitude. Allowing the emulation methodology to be applied in higher dimensions requires very careful treatment. Several problems arise:

- If the simulator has a very large number of seemingly relevant inputs (assuming the screening has already been applied to remove totally irrelevant inputs), then the number of training points required to cover the input space might be very large indeed, growing typically as n^D, where n is the number of points required along each dimension to fill that space, and D is the dimension of the input space. Other design schemes have somewhat better sampling properties, but at some expense.

[1] We use the ‘Big Oh’ notation, to denote the order of magnitude.

- If there are a very large number of input dimensions, then estimating parameters in the corresponding covariance functions can be very time consuming (and quite ill posed with a small number of training points) even when assuming independence of length scales. If a completely isotropic distance metric is considered then the number of parameters to estimate will scale as $\frac{1}{2}(D^2)$.

These are in essence computational problems in that, at least theoretically speaking, they are able to be solved by extensive use of computational resources. However a more elegant approach might be to assess whether we can apply dimension reduction before the application of the emulator, in order to address some of these issues. Of course in doing so, one has to take care that the dimension reduction itself is not more computationally demanding than the emulation.

A key idea we wish to explore and test is that by defining a projection of the input space to a lower dimensional manifold we will be able to exploit that lower dimensional manifold in our experimental design, designing the simulator runs in the lower dimensional manifold, and then mapping back to the original input space in order to evaluate the simulator output for the design points. In this manner we hope to address the potentially devastating scaling of the number of design points with input dimension. Another advantage that emerges is that our emulator can be learnt from the lower dimensional manifold to the output space, simplifying the Gaussian process model, and hopefully thus permitting tighter credibility intervals on the emulator. In the discussion we address whether this is really worth all the effort given that an emulator could be applied in the high dimensional space directly.

We expect the dimension reduction methods to have most relevant to models with spatial or temporal inputs, since location and time typically have strong separation distance dependent correlations, induced by real (mixing) processes. Most simulators of fields over location or time treat these continuous, thus infinite dimensional, spaces by discretisation. The discretisation must be sufficiently fine to resolve the features of interest, but the correlation induces redundancy in the representation (which is required for the internal computations in the simulators). It is quite common in many applications to attempt to reduce the redundancy using dimension reduction methods such as principal components analysis, and other methods.

3 The dimension reduction decision tree

Figure 1 presents a decision tree which gives an overview of which methods / approaches should be considered in given situation. In future, this tree will be further expanded.

At the top level we need to check whether or not we have any observations[2]; if we have a sufficient number of observations to characterise the probability density of the inputs $p(\mathbf{X})$ then we can consider *unsupervised dimensionality reduction* techniques discussed in Section 4. Also, in cases where prior elicitation on the inputs has provided a very informative estimate of the joint distribution of the inputs, $p(\mathbf{X})$, we could consider some form of *unsupervised dimensionality reduction.* In general this seems unlikely as most priors are given over the *marginal distributions* of each parameter, and thus the only assumption one could make is to assume a factorising joint distribution which is not able to capture any interesting structure in the inputs. However if there is good prior information, then the most straightforward approach would be to sample from $p(\mathbf{X})$ and treat these samples as being pseudo observations and then applying the *unsupervised dimensionality reduction* approaches mentioned in Section 4 before attempting emulation.

One of the weaknesses of the unsupervised dimensionality reduction methods is that no account is taken of the simulator output values. It might be the case that the projection[3] of the inputs needed to explain the output distribution, rather than capture the input distribution, is either simpler, or allows projection onto a lower dimensional manifold[4]. This is very related to screening, which can be seen as the extreme of removing (projecting to a null space, classical linear algebra projection) all variables that are not important in determining the value of the output. *Supervised dimension reduction* methods,

[2]by observations we mean, possibly indirect, measurements of the inputs to the simulator of the physical system.

[3]We define projection to mean the mapping from a D dimensional space to a M dimensional space, and, in our dimension reduction context, we generally think of this as going from the high dimension space to the lower dimension space, although this need not always be the case, unlike the application of the term in linear algebra.

[4]A manifold is a lower dimensional object embedded into a higher dimensional space, e.g. a (possibly curved) 2D surface embedded into a 3D space.

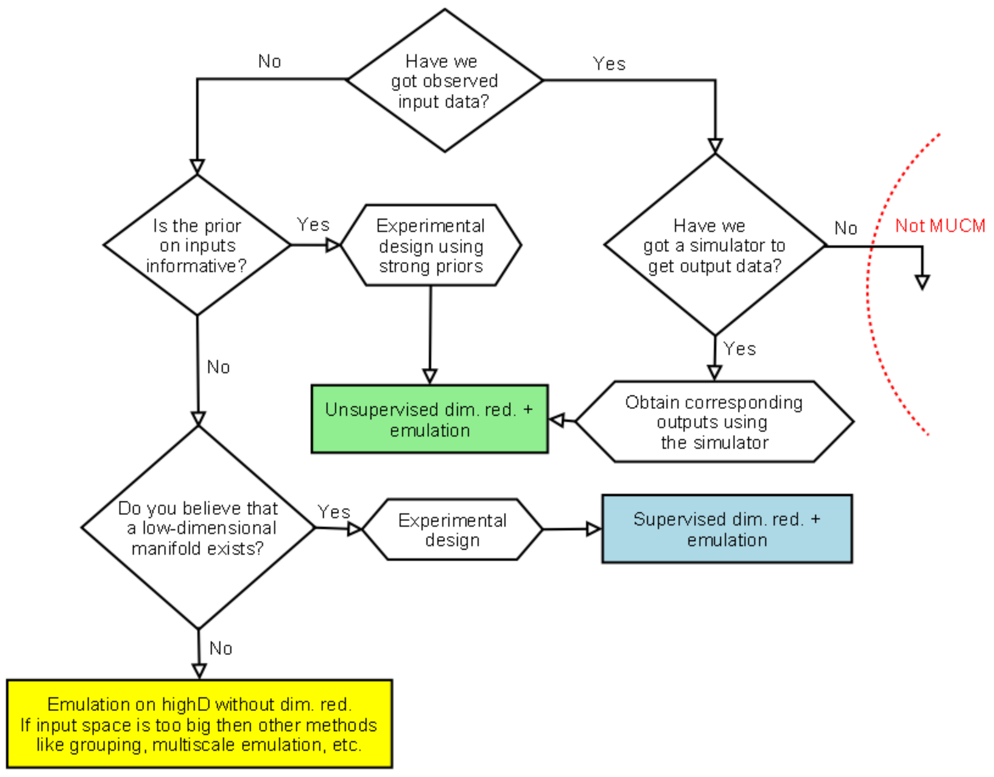

Have we got observed input data?
No
Yes
Is the prior on inputs informative?
Yes
Experimental design using strong priors
Have we got a simulator to get output data?
No
Not MUCM
No
Yes
Unsupervised dim. red. + emulation
Obtain corresponding outputs using the simulator
Do you believe that a low-dimensional manifold exists?
Yes
Experimental design
Supervised dim. red. + emulation
No
Emulation on highD without dim. red.
If input space is too big then other methods like grouping, multiscale emulation, etc.

Figure 1: The decision tree for dealing with high dimensional inputs. In future this will be expanded.

discussed in Section 5, can be useful in this setting, since they consider both the inputs and the outputs jointly in determining the dimension reduction.

Thus far the emphasis has been on the input space, which is probably the most tricky problem to tackle, since the simulator can be used to generate a distribution over the output space, given the inputs, which might be expected to naturally lie on some manifold even when the inputs are sampled from some naive, possibly uniform distribution. The most obvious approach for dimensionality reduction in the output space is to separate the outputs so that they become independent - probably using Independent Component Analysis (ICA) (Comon, 1994); then we can construct separate emulators for each output, and recombine them to produce the joint output distribution (since standard ICA is a linear projection, we can construct a joint covariance matrix quite easily). In future work we will consider the output space.

Focus of this report is on dimension reduction in the input space. The following two sections present the unsupervised and supervised dimensionality reduction domains and the methods we considered from each of them.

4 Unsupervised dimensionality reduction

Unsupervised dimensionality reduction approaches are most relevant to simulators of a real system, for which there are large numbers of observations available, or were there is strong prior information on the simulator input space which defines the joint distribution over the simulator inputs. The most likely scenario is that the simulator does not make physical sense for certain joint ranges of inputs. If this restriction to certain parts and combinations of state space defines a manifold, then being able to determine this manifold will be very useful, since the simulator runs can be designed on this manifold. This is very much related to *prior elicitation* and the concept of implausibility in emulation. A particular case where we expect this might be very useful is where the inputs have spatial or spatio-temporal support - that is where they represent spatial or spatio-temporal fields of inputs to a simulator, which will typically be discretised over a grid or basis function representation.

We do not present an exhaustive list of dimension reduction techniques, rather focusses on those we believe will be most relevant, or with which we have most experience.

4.1 Principal component analysis

Principal Component Analysis (PCA) is a multivariate procedure which rotates the data such that maximum variance is projected onto the principal axes. Essentially, a set of correlated variables are transformed into a set of uncorrelated variables which can be ordered by reducing variance. The uncorrelated variables, often called principal components, are linear combinations of the original variables, and in many cases the last few of these principal components can be removed with minimum loss of information, since these are often, but not always essentially noise variables. If the data covariance matrix has full rank, there can be as many principal components as there are variables.

Given a set of observations $\tilde{\mathbf{x}}_n \in \mathbb{R}^Q, n = 1, \ldots, N$, which are centred, $\sum_{n=1}^N \tilde{\mathbf{x}}_n = 0$, in PCA we find the principal components by diagonalising the covariance matrix given by:

$$\boldsymbol{C} = \frac{1}{N} \sum_{n=1}^{N} \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^\top, \tag{1}$$

which is achieved in practice using an eigen-decomposition:

$$\mathbf{CU} = \mathbf{U\Lambda}, \tag{2}$$

where $\mathbf{U}$ is a $Q \times Q$ matrix which has the unit length eigenvectors with columns $\mathbf{u}_1, \ldots, \mathbf{u}_Q$ and Λ is a diagonal matrix with the corresponding eigenvalues, $\lambda_1, \ldots, \lambda_Q$. The eigenvectors are the principal components and the eigenvalues are the corresponding variances.

PCA is the most commonly applied projection method. The fact that PCA defines a linear, orthogonal sub-space gives it favourable computational and interpretability properties, but it is also its main limitation since any non-linear relationship between variables will not be captured.

4.2 Neuroscale

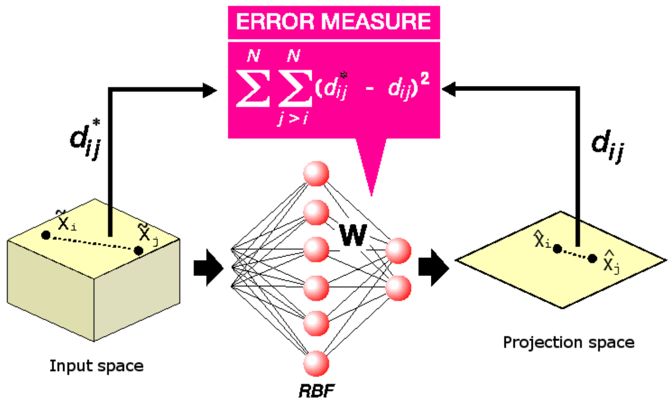


Figure 2: Schematic representation of the Neuroscale model.

Neuroscale (Lowe and Tipping, 1997) is a neural network based implementation of Sammon's mapping. It is a projection method which attempts to preserve relative 'similarity' (in this case characterised by *Euclidean* distance) between points in the higher-dimensional input space ($\tilde{\mathbf{X}}$) and the lower-dimensional projection space ($\hat{\mathbf{X}}$), as shown in Figure 2 (adapted from (Tipping and Lowe, 1998)).

The distance-preserving nature of the transformation is imposed by the 'stress' term which accounts for the preservation of inter-point similarity

$$E = \sum_{i}^{N} \sum_{j>i}^{N} (d^*_{ij} - d_{ij})^2 \ , \tag{3}$$

where

$$d^*_{ij} = \parallel \tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j \parallel , \tag{4}$$

and

$$d_{ij} = \parallel \hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j \parallel , \tag{5}$$

where the metric is in general the Euclidean one, although others might be conceived.

Points are projected from the data space onto the latent (projection) space by means of a radial basis function (RBF) neural network. The RBF parameters are defined by minimising the 'stress' term, rather than the traditional residual error minimisation that might be employed in a regression setting. The high dimensional targets are projected onto the low dimensional representation using the RBF:

$$\hat{\mathbf{X}} = \mathbf{W}\boldsymbol{\phi}(\tilde{\mathbf{X}}), \tag{6}$$

where $\mathbf{W}$ is a weight matrix and $\boldsymbol{\phi}(\cdot)$ are the basis functions.

In the experiments presented in this report we initialise Neuroscale using PCA. Initialisation of Neuroscale is an important issue and finding the best way to initialise it is an open question.

4.3 Generative topographic mapping

The Generative Topographic Mapping (GTM) models a constrained probability distribution in the high-dimensional data space, $\mathcal{Q} \subset \mathbb{R}^Q$, by means of low-dimensional latent, or hidden, variables (Bishop et al., 1998). The data is projected / visualised in the latent space, $\mathcal{H} \subset \mathbb{R}^L$.

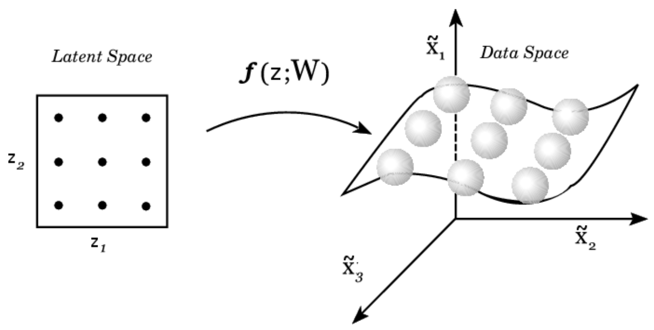


Figure 3: Schematic representation of the GTM model.

As demonstrated in Figure 3 (adapted from (Bishop et al., 1998)), we cover the latent space, $\mathcal{H}$, with an array of M latent space centres, $\mathbf{z}_i \in \mathcal{H}, i = 1, 2, ..., M$. The non-linear GTM transformation, $f : \mathcal{H} \Rightarrow \mathcal{Q}$, from the latent space to the data space is defined using the RBF network. We define in the latent space with a set of K fixed non-linear basis functions (we use Gaussian functions of the same width σ), $\boldsymbol{\phi} : \mathcal{H} \Rightarrow \mathbb{R}$, on a regular grid in the latent space. Given a point $\mathbf{z} \in \mathcal{H}$ in the latent space, its image under the map f is

$$f(\mathbf{z}_i, \mathbf{W}) = \boldsymbol{\phi}(\mathbf{z}_i)\mathbf{W}, \tag{7}$$

where $\boldsymbol{\phi}(\mathbf{z}_i) = (\phi_1(\mathbf{z}_i), ..., \phi_K(\mathbf{z}_i))^T$ and $\mathbf{W}$ is a $K \times Q$ matrix of weight parameters.

GTM defines a generative probabilistic model in the data space by placing a radially symmetric multi-variate Gaussian with zero mean and inverse variance β around images, under f, of the latent space centres $\mathbf{z}_i \in \mathcal{H}, i = 1, 2, ..., M$. We refer to the Gaussian density associated with the centre $\mathbf{z}_i$ by $p(\tilde{\mathbf{x}} \mid \mathbf{z}_i, \mathbf{W}, \beta)$. Defining a uniform prior over $\mathbf{z}_i$, the density model in the data space provided by the GTM is

$$p(\tilde{\mathbf{x}} \mid \mathbf{W}, \beta) = \frac{1}{M} \sum_{i=1}^{M} p(\tilde{\mathbf{x}} \mid \mathbf{z}_i, \mathbf{W}, \beta). \tag{8}$$

For the purpose of data projection, we use Bayes' theorem to invert the transformation f from the latent space $\mathcal{H}$ to the data space $\mathcal{Q}$. The posterior distribution on $\mathcal{H}$, given a data point $\tilde{\mathbf{x}}_n \in \mathcal{Q}$, is a sum of delta functions centred at centres $\mathbf{z}_i$, with coefficients equal to the posterior probability $R_{i,n}$ that the i-th Gaussian (corresponding to the latent space centre $\mathbf{z}_i$) generated $\tilde{\mathbf{x}}_n$:

$$R_{i,n} = \frac{p(\tilde{\mathbf{x}}_n \mid \mathbf{z}_i, \mathbf{W}, \beta)}{\sum_{j=1}^{M} p(\tilde{\mathbf{x}}_n \mid \mathbf{z}_j, \mathbf{W}, \beta)}. \tag{9}$$

The latent space representation of the point $\tilde{\mathbf{x}}_n$, i.e. *the projection of* $\tilde{\mathbf{x}}_n$, is taken to be the mean, $\sum_{i=1}^{M} R_{in}\mathbf{z}_i$ of the posterior distribution on $\mathcal{H}$. The parameters of the GTM (weights $\mathbf{W}$ and inverse variance β) are inferred from data using a variant of the Expectation Maximisation (EM) algorithm. The f–image of the latent space $\mathcal{H}, \Omega = f(\mathcal{H}) = \{f(\mathbf{z}) \in \mathbb{R}^Q \mid \mathbf{z} \in \mathcal{H}\}$, forms a smooth L-dimensional manifold in the data space. We refer to the manifold Ω as the *projection manifold* of the GTM.

4.4 Gaussian process latent variable model

Recently introduced, the Gaussian process latent variable model (GP-LVM) is a fully probabilistic, non-linear, latent variable model that is gaining in popularity and is applied mainly in visualisation contexts. An important characteristic of the GP-LVM algorithm is that it provides a smooth mapping from the latent space to the data space. This property of ease and accuracy with which probabilistic reconstructions of the data can be made, given a (possibly new) point in the latent space can be very useful in the emulator setting where once we find a good latent space, we might want to design new points in the latent space and obtain their corresponding input points on which the simulator can be evaluated.

The GP-LVM relates a high-dimensional data set, $\mathbf{X}$, and a low dimensional latent space, $\mathbf{Z}$, using a Gaussian process mapping from the latent space to the data space. Given a covariance function for the Gaussian process, $k_X(\mathbf{z}, \mathbf{z}')$, the likelihood of the data given the latent positions is,

$$p(\mathbf{X}|\mathbf{Z}, \bar{\beta}) = \frac{1}{\sqrt{(2\pi)^{ND}|\mathbf{K}_X|^D}} \exp\left(-\frac{1}{2}\mathtt{tr}(\mathbf{K}_X^{-1}\mathbf{X}\mathbf{X}^{\mathrm{T}})\right), \tag{10}$$

where elements of the kernel matrix $\mathbf{K}_X$ are defined by the covariance function, $(\mathbf{K}_X)_{i,j} = k_X(\mathbf{z}_i, \mathbf{z}_j)$. For the results presented in this report we use a kernel that is the sum of an RBF, a bias or constant term, and a noise term.

$$k_X(\mathbf{z}, \mathbf{z}') = \theta_1 \exp\left(-\frac{\theta_2}{2} \parallel \mathbf{z} - \mathbf{z}' \parallel^2\right) + \theta_3 + \frac{\delta_{\mathbf{z},\mathbf{z}'}}{\theta_4}, \tag{11}$$

where $\theta = \{\theta_1, \theta_2, ...\}$ comprises the kernel hyper-parameters that govern the output variance, the RBF support width, the bias, and the variance of the additive noise, respectively. The posterior can be written as

$$p(\mathbf{Z}, \bar{\beta}|\mathbf{X}) \propto p(\mathbf{X}|\mathbf{Z}, \bar{\beta})p(\mathbf{Z})p(\theta). \tag{12}$$

Estimation in the GP-LVM consists of minimizing the log posterior with respect to the latent space configuration, $\mathbf{Z}$, and the hyper-parameters, θ,

$$\mathcal{L} = \mathcal{L}_r + \sum_i \ln \theta_i + \sum_i \frac{1}{2} \parallel \mathbf{z}_i \parallel^2, \tag{13}$$

where uninformative priors are placed over the kernel hyper-parameters, and simple priors over the latent positions. The priors over the latent positions prevent the GP-LVM from placing latent points infinitely far apart, i.e. latent positions close to the origin are preferred. The log likelihood associated with (10) is,

$$\mathcal{L}_r = \frac{D}{2} \ln |\mathbf{K}_X| + \frac{1}{2}\mathtt{tr}(\mathbf{K}_X^{-1}\mathbf{X}\mathbf{X}^{\mathrm{T}}) \tag{14}$$

A key benefit of the GP-LVM model is its use of (closed form) Bayesian model averaging (Lawrence, 2005), both to mitigate problems due to over-fitting with small data sets, and to remove the need to select parameters of the function approximators.

4.5 Other unsupervised dimensionality reduction methods

Other dimension reduction techniques, widely popular in machine learning for visualisation purposes, include methods for locally linear embedding (Roweis and Saul, 2000), Hessian LLE (Donoho and Grimes, 2003), Laplacian Eigenmaps (Belkin and Niyogi, 2003), and Isomap (J. B. Tenenbaum and Langford, 2000). Broadly, these techniques construct a low-dimensional data representation using a cost function that retains local properties of the data. In the emulation setting our goal is to obtain a good predictive manifold rather than visualisation so we have not considered these approaches further. However early results do suggest that local methods might have a role in the future since the global methods based on a functional mapping demonstrate a tendency to many local minima. Local methods might be more robust in determining manifolds, particularly when these are well defined and well separated, but potentially very complex in shape.

5 Supervised dimensionality reduction

In the previous section we discussed dimension reduction methods which only use the input data, $\mathbf{X}$ to determine a lower dimensional manifold. There are some dimension reduction methods which also, or only, use output information, $\mathbf{Y}$, during dimension reduction. One of the major domains in this regard is known as Sufficient Dimensionality Reduction (SDR). The main idea of SDR for regression is that of finding a subspace, $\mathcal{S}$, such that the projection of the covariate vector, $\mathbf{X}$, onto, $\mathcal{S}$, captures the statistical dependence of the response $\mathbf{Y}$ and $\mathbf{X}$. This desideratum is captured formally as a conditional independence assertion:

$$\mathbf{Y} \perp\!\!\!\perp \mathbf{X} | \mathbf{B}^{\mathrm{T}}\mathbf{X}, \tag{15}$$

where $\mathbf{B}$ denotes the orthogonal projection of $\mathcal{X}$ onto $\mathcal{S}$. The subspace $\mathcal{S}$ is referred to as a dimension reduction subspace. Dimension reduction subspaces are not unique. We can derive a unique "minimal" subspace, defined as the intersections of all reduction subspaces $\mathcal{S}$. This minimal subspace does not necessarily satisfy the conditional independence assertion; when it does, the subspace is referred to as the central subspace.

The seminal paper of Li (1991) on Sliced Inverse Regression (SIR) started much interest in SDR for regression. There has been a lot of interest in SDR since then and many methods based on this idea have been developed to effectively identify the central subspace and applications of SDR (Li, 1992; Cook and Yin, 2001; Chiaromonte and Cook, 2002; Li et al., 2005). Most of these methods make some sort of modelling assumptions. They are effective if the modelling assumptions that they embody are met, but if these assumptions do not hold there is no guarantee of finding the central subspace. Recently, contributions from Xia et al. (2002) and Fukumizu et al. (2004) have provided ways to reduce the number of assumptions. In this section we will introduce these supervised methods.

5.1 Sliced inverse regression

SIR is an effective technique for dimension reduction in non-parametric regression problems where the response variable y depends on K unknown linear combinations of the input (explanatory) variables $(x_1, \ldots, x_p) = \mathbf{x}^{\mathrm{T}}$, but the exact form of dependence is unknown. These regression problems can be represented by the model

$$y = f(\beta_1^{\mathrm{T}}\mathbf{x}, \beta_2^{\mathrm{T}}\mathbf{x}, \ldots, \beta_K^{\mathrm{T}}\mathbf{x}, \epsilon), \tag{16}$$

where f is an unknown function defined on $\mathbb{R}^{K+1}; K < p; \beta_1, \ldots, \beta_K$ are unknown $p \times 1$ vectors, and ϵ and $\mathbf{x}$ are independent random variables. The essential feature of the model is that instead of the p-dimensional $\mathbf{x}$ we need only the K-dimensional variables $(\beta_1^{\mathrm{T}}\mathbf{x}, \ldots, \beta_K^{\mathrm{T}}\mathbf{x})$ for predicting y. As Li (1991) pointed out, since the function f is unknown, the parameters $\beta_1, \ldots, \beta_K$ are not identifiable; however, the linear space, $\mathcal{B}$, spanned by the β's is identifiable. The space $\mathcal{B}$ is called the effective dimension-reduction (e.d.r.) space and any basis of $\mathcal{B}$ is called a set of e.d.r. directions. Naturally, the first step of solving the regression problem (16) is to estimate the e.d.r. space $\mathcal{B}$. Then the form of the function f can be explored based on the estimated e.d.r. space. Of course, since f and the β's are irretrievably confounded, the form of f usually depends on the particular choice of the representation of the e.d.r. space.

Under model (16) and the condition that the conditional expectation of any linear function of $x_1, \ldots, x_p$ given $\beta_1^{\mathrm{T}}\mathbf{x}, \ldots, \beta_K^{\mathrm{T}}\mathbf{x}$ is also a linear function $\beta_1^{\mathrm{T}}\mathbf{x}, \ldots, \beta_K^{\mathrm{T}}\mathbf{x}$, Li (1991) shows that the centred "inverse regression" curve $E(\mathbf{x}|y) - E(\mathbf{x})$ is confined to a K-dimensional linear subspace spanned by $\Sigma_x\beta_1, \Sigma_x\beta_2, \ldots, \Sigma_x\beta_K$, where Σ_x denotes the covariance matrix of $\mathbf{X}$. This relates the inverse regression $E(\mathbf{x}|y)$ to the e.d.r. space $\mathcal{B}$. Let $z = \Sigma_x^{-1/2}(\mathbf{x} - E(\mathbf{x}))$ be the standardization of $\mathbf{x}$; then the e.d.r. space $\mathcal{B}$ is spanned by $\Sigma_x^{-1/2}\eta_1, \ldots, \Sigma_x^{-1/2}\eta_K$, where $\eta_1, \ldots, \eta_K$ are the column eigenvectors associated with the K largest eigenvalues of the covariance matrix $\mathtt{Cov}(E(\mathbf{z}|y))$. Thus it involves eigen analysis. The algorithm is straightforward to implement. Further details are available in (Li, 1991).

5.2 Minimum average variance estimation

The Minimum Average Variance Estimation (MAVE) algorithm improves in several aspects over SIR method (Xia et al., 2002). First, MAVE does not undersmoothing when estimating the link function f (see eq. (16)) to achieve a faster rate of convergence. Secondly, MAVE can be applied to many models including dynamical models and has been extended to other related problems such as classification (Antoniadis et al., 2003).

To describe MAVE, let K represent the working dimension, $1 \leq K \leq p$. For a given number K of directions B in model (16), Xia et al. (2002) proposed to estimate B by minimising the unexplained variance $E\{y - E(y|\mathbf{x})\}^2 = E\{y - f(B^{\mathrm{T}}\mathbf{x})\}^2$, where the unknown function f is locally approximated by a linear function; that is, $f(B^{\mathrm{T}}\mathbf{x}_0) \approx a_0 + b_0^{\mathrm{T}} B^{\mathrm{T}}(\mathbf{x} - \mathbf{x}_0)$ around some $\mathbf{x}_0$. The novel feature of MAVE is that one minimises simultaneously with respect to directions B and coefficients a_0 and b_0 of the local linear approximation. Hence, given a sample $(\mathbf{x}_i, y_i)_{i=1}^n$ from $(\mathbf{X}, \mathbf{y})$, MAVE estimates B by minimising

$$\min_{\substack{B: B^{\mathrm{T}}B = I_p \\ a_j, b_j, j=1,\ldots,n}} \sum_{i=1}^n \sum_{j=1}^n [y_i - \{a_j + b_j^{\mathrm{T}} B^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)\}]^2 w_{ij}, \tag{17}$$

where w_{ij} are weights describing the local character of linear approximation. Initially, weights at any point $\mathbf{x}_0$ are given by a multidimensional kernel function K_h, where h refers to a bandwidth: $w_{i0} = K_h(\mathbf{x}_i - \mathbf{x}_0)\{\sum_{i=1}^n K_h(\mathbf{x}_i - \mathbf{x}_0)\}^{-1}; i = 1, \ldots, n$. Additionally, once we have an estimate $\hat{B}$ of the reduced dimension space, it is possible to iterate using weights based on distances in the reduced space: $w_{i0} = K_h\{\hat{B}^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_0)\}[\sum_{i=1}^n K_h\{\hat{B}^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_0)\}]^{-1}$. Iterating until convergence results in a refined MAVE.

5.3 Kernel dimension reduction

Kernel Dimension Reduction (KDR) is a recently-developed approach to SDR that uses cross-covariance operators on Reproducing Kernel Hilbert Spaces (RKHS) to measure quantities related to conditional independence, with the restrictive assumptions of linearity that constrain the applicability of SIR.

Let $\mathcal{H}_{\mathcal{X}}$ be a RKHS of functions on $\mathcal{X}$ induced by the kernel function $\mathbf{K}_{\mathcal{X}}(,\mathbf{X})$ for $\mathbf{X} \in \mathcal{X}$. We define the space $\mathcal{H}_{\mathcal{Y}}$ and the kernel function $\mathbf{K}_{\mathcal{Y}}$ similarly. Define the cross-covariance between a pair of functions $f \in \mathcal{H}_{\mathcal{X}}$ and $g \in \mathcal{H}_{\mathcal{Y}}$ as follows:

$$\mathbf{C}_{fg} = \mathbf{E}_{\mathbf{XY}}\left[(f(\mathbf{X}) - \mathbf{E}_{\mathbf{X}}[f(\mathbf{X})])\,(g(\mathbf{Y}) - \mathbf{E}_{\mathbf{Y}}[g(\mathbf{Y})])\right]. \tag{18}$$

It turns out that there exists a bilinear operator $\Sigma_{\mathbf{YX}}$ from $\mathcal{H}_{\mathcal{X}}$ and $\mathcal{H}_{\mathcal{Y}}$ such that $\mathbf{C}_{fg} =< g, \Sigma_{\mathbf{YX}} f >_{\mathcal{H}_{\mathcal{Y}}}$ for all functions f and g. Similarly we can define covariance operators $\Sigma_{\mathbf{XX}}$ and $\Sigma_{\mathbf{YY}}$. Finally, we can use these operators to define a class of conditional cross-covariance operators in the following way:

$$\Sigma_{\mathbf{YY}|\mathbf{X}} = \Sigma_{\mathbf{YY}} - \Sigma_{\mathbf{YX}}\Sigma_{\mathbf{XX}}^{-1}\Sigma_{\mathbf{XY}}. \tag{19}$$

This definition assumes that $\Sigma_{\mathbf{XX}}$ is invertible; more general cases are discussed in (Fukumizu et al., 2006).

Note that the conditional covariance operator of $\Sigma_{\mathbf{YY}|\mathbf{X}}$ eq. (19) is "less" than the covariance operator $\Sigma_{\mathbf{YY}}$, as the difference $\Sigma_{\mathbf{YX}}\Sigma_{\mathbf{XX}}^{-1}\Sigma_{\mathbf{XY}}$ is positive semidefinite. This agrees with the intuition that conditioning reduces uncertainty.

Now we are ready to link these cross-covariance operators to the central subspace. Consider any subspace $\mathcal{S}$ in $\mathcal{X}$. Let us map this subspace to a RKHS $\mathcal{H}_{\mathcal{S}}$ with a kernel function $\mathbf{K}_{\mathcal{S}}$. Furthermore, let us define the conditional covariance operator $\Sigma_{\mathbf{YY}|\mathcal{S}}$ as if we would regress $\mathbf{Y}$ on $\mathcal{S}$. Fukumizu et al. (2006) proves that $\Sigma_{\mathbf{YY}|\mathbf{X}} \leq \Sigma_{\mathbf{YY}|\mathbf{Z}}$, where $\mathbf{Z} = \mathbf{B}\mathbf{B}^{\mathrm{T}}\mathbf{X} \in \mathcal{S}$ and $\mathbf{B}$ is a projection matrix such that $\mathbf{B}\mathbf{B}^{\mathrm{T}}$ is an identity matrix, and $\Sigma_{\mathbf{YY}|\mathbf{X}} = \Sigma_{\mathbf{YY}|\mathbf{Z}}$ if and only if $\mathbf{Y} \perp\!\!\!\perp \mathbf{X}|\mathbf{B}^{\mathrm{T}}\mathbf{X}$, that is, $\mathcal{S}$ is the central subspace (Fukumizu et al., 2006, Theorem 1).

Note that this theorem does not impose assumptions on either the marginal distributions of $\mathbf{X}$ and $\mathbf{Y}$ or the conditional distribution $P(\mathbf{Y}|\mathbf{X})$. This results lead to an algorithm for estimating the central subspace, characterised by $\mathbf{B}$, for empirical samples. Further information on this algorithm is given in (Fukumizu et al., 2006). The essence of the method is to jointly estimate a projection and regression as in SDR, but use the 'kernel trick' to work in a feature space that allows for non-linear relationships.

6 Experimental results and discussion

The toy dataset generation framework presented in Boukouvalas et al. (2007) is employed for the experiments presented here. The details of the experimental set-up is presented in Section 6.1. It is followed by two sections presenting results on two different datasets, one with weakly non-linear embedding of a lower-dimensional manifold into higher-dimensions and the second one with a more non-linear embedding, qhich raises several challenging issues. Results of the effect of the degree of non-linearity in the embedded manifold on the use of dimension reduction methods is discussed in Section 6.4.

6.1 The data generation set-up

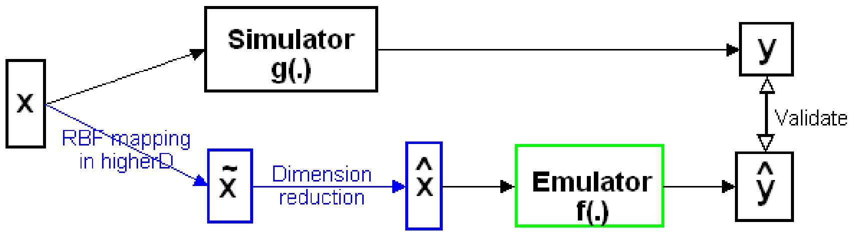


Figure 4: Experimental set-up. The problem that is faced in the real world is that there is an input set $\tilde{\mathbf{X}}$ and outputs from the simulator $\mathbf{y}$, that should be emulated. The toy data is set-up so that the real inputs of the simulator, $\mathbf{X}$ are some projection of $\tilde{\mathbf{X}}$, and in essence the aim is to identify these, as $\hat{\mathbf{X}}$.

Figure 4 summarises the toy-dataset generation framework used for the experiments presented here. To generate the toy-datasets:

1. sample N base vectors, $\mathbf{X}$, of dimensionality $N \times d$ using a Latin hypercube sampling from a uniform distribution (Stein, 1987). All dimensions have the same domain $[l, u]$;

2. obtain noiseless model outputs, $\mathbf{y}$, using a chosen simulator function. For the datasets generated for the experiments presented in this report, the simple $y = f(\mathbf{x}) = \sum_{i=1}^{D} \sin(x_i)$ was used as the simulator function;

3. to evaluate the dimensionality reduction methods, we artificially map the base input data ($\mathbf{X}$), living in lower dimension, d, into a higher dimensional space, of dimension q. A radial basis function (RBF) network is used for this mapping as it can provide a range of complexities for the non-linear mapping, from approximately linear, to highly non-linear. The resulting data matrix, mapped into higher dimension (q), $\tilde{\mathbf{X}}$, is a $N \times q$ matrix.

Once the toy-data,$(\mathbf{X}, \tilde{\mathbf{X}}, \mathbf{y})$,is generated, 50% of the the high-dimensional input points, $\tilde{\mathbf{X}}$, were used to estimate the parameters of the dimension reduction methods which were then used to map the high-dimensional input points onto a lower-dimension space, generating $\hat{\mathbf{X}}$ for each method. A Gaussian process emulator for $\mathbf{y}$, with Matern kernel, was then trained on the lower-dimensional projection ($\hat{\mathbf{X}}$) using the same data used to estimate the parameters of the dimension reduction methods.

6.2 Weakly non-linear embedding dataset

Using the above set-up a dataset with 1 relevant input was generated, which was then embedded into a 2-dimensional space using a *weakly non-linear* RBF mapping (so here $d = 1$ and $q = 2$). In this context, *weakly non-linear* means that a very small number of basis functions was chosen and that the parameters in these were sampled such that the embedding of the 1D line into the 2D plane was very smooth. This simple example is used to assist in understanding the behaviour of the dimension reduction techniques since it is possible to visualise the high-dimensional ($q = 2$) data and corresponding mapping into $d = 1$ dimensions, obtained using the various dimension reduction techniques. $N = 300$ data points were generated in the range on $[-3, 3]$. Our goal is to demonstrate the projection obtained using the dimension reduction methods discussed in Section 4 and Section 5.

Figure 5 presents projection obtained using different dimensionality reduction methods we considered. Note that mapping of the lower-dimensional base input data into higher dimension is almost linear so as it can be seen in Figure 5(b), we have a slightly curved *sin* function. Not surprisingly, all dimension reduction methods perform reasonably well. MAVE fails to map the more distorted part of the *sin* curve and thus the emulator developed on the projection obtained using MAVE gives higher root mean squared error (RMSE) compared to the emulators developed using the projection obtained using other dimension reduction methods. RMSE values are presented in Table 1.

6.3 Non-linear mapping dataset

To address how the algorithms are able to cope with increasingly non-linear embeddings, a more non-linear embedding was constructed. This dataset has an identical set-up as in the previous section but here the base data is embedded into a 2-dimensional space using a *non-linear* RBF mapping, obtained by sampling from a more complex mapping with larger parameter values.

	$\tilde{\mathbf{X}}$	$\hat{\mathbf{X}}_{\texttt{PCA}}$	$\hat{\mathbf{X}}_{\texttt{NS}}$	$\hat{\mathbf{X}}_{\texttt{GTM}}$	$\hat{\mathbf{X}}_{\texttt{GPLVM}}$	$\hat{\mathbf{X}}_{\texttt{SIR}}$	$\hat{\mathbf{X}}_{\texttt{MAVE}}$	$\hat{\mathbf{X}}_{\texttt{KDR}}$
Almost linear mapping dataset	0.0000	0.0000	0.0000	0.0081	0.00143	0.0000	0.5450	0.0000
Non-linear mapping dataset	0.0000	0.1660	0.0000	0.1465	0.1025	0.7633	0.7588	0.3075

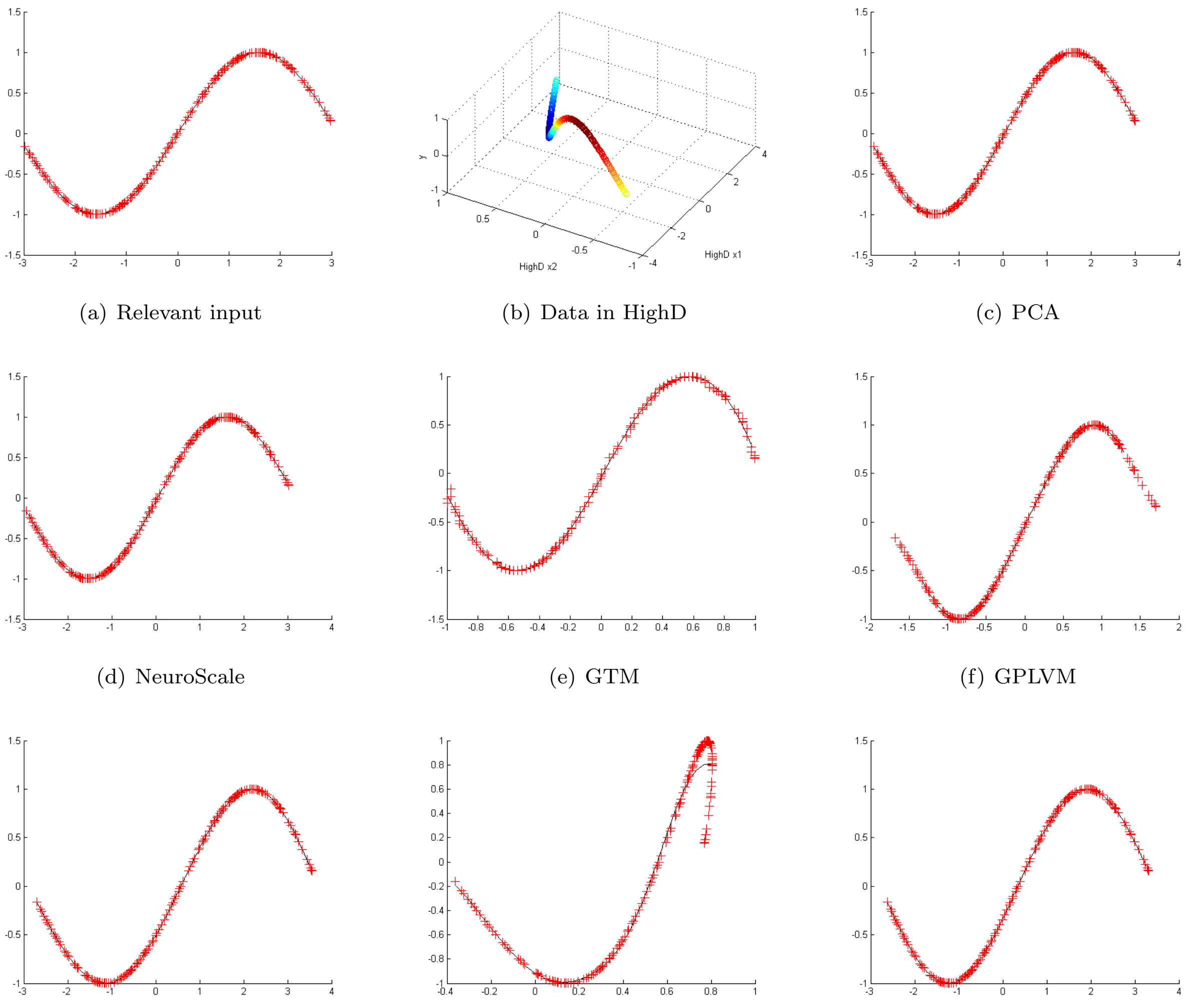
Table 1: Root mean square prediction errors in emulation on the test set (up to 4 decimal places) using the high dimensional data $\tilde{\mathbf{X}}$ and the projected data $\hat{\mathbf{X}}$ using a range of dimension reduction methods.

Figure 6 presents the results obtained using different dimension reduction methods on the more strongly non-linear dataset. Figure 6(b) shows the non-linear embedding of the base data into high-dimension produces a more distorted embedding with an almost circular structure. This causes all of the linear dimension reduction methods (PCA, SIR, and MAVE) to fail. It is surprising that KDR, inspite of being a non-linear extension of SIR, did not perform well in this scenario. We suspect that initialisation and subsequent convergence / optimisation issues have caused the failure of KDR but this requires further investigation. NS, GTM, and GP-LVM show promising results, and are able to capture the latent space manifold accurately in this example.

Table 1 shows that errors in an emulator trained on the projection of the high dimensional inputs $\tilde{\mathbf{X}}$ does rather well in both cases, having a very low RMSE, but this is not overly surprising since high dimensional simply means 2D in this case. The only dimension reduction method that is able to maintain this level of error on this data set in the Neuroscale projection and we believe this might be related to the distance preserving nature of the Neuroscale projection.

6.4 The effect of non-linearity of the manifold

As can be observed from the results presented in the previous section, it is important to know complexity (non-linearity, shape) of the embedded manifold to decide which dimension reduction technique to apply and whether they will be able to determine the manifold correctly. In our experiments, all of the methods discussed here fail when the manifold is very complex. We also observed that an emulator developed with all input dimensions (the high-dimensional data, $\tilde{\mathbf{X}}$) almost always gives lower RMSE than an emulator developed with projected data, $\hat{\mathbf{X}}$, using any dimension reduction technique when the simulator function

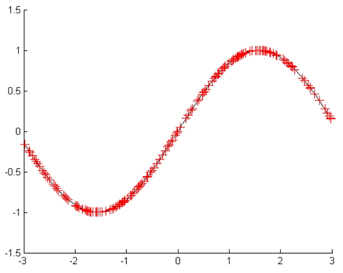


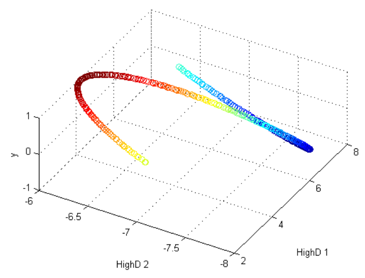
(a) Relevant input (b) Data in HighD (c) PCA

(d) NeuroScale (e) GTM (f) GPLVM

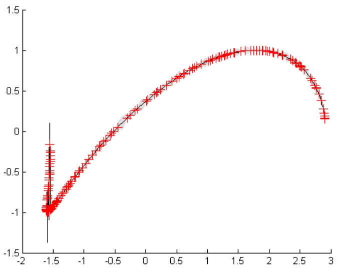
(g) SIR (h) Using MAVE (i) KDR

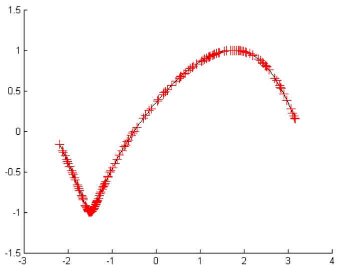
Figure 5: Output mapping with the base data and the projections obtained using different dimension reduction methods for the almost linear (weakly non-linear) mapping dataset. Training data (plus), mean prediction using a trained GP on the test set (solid line) with error bars at 2 standard deviations (gray area, typically very difficult to see due to the density of the training data).

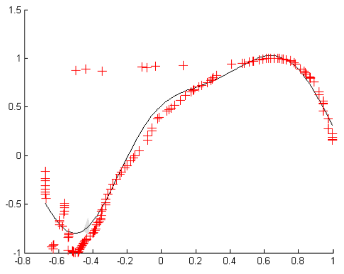

(a) Relevant input

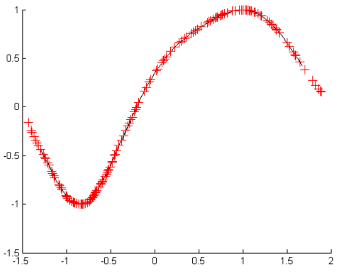


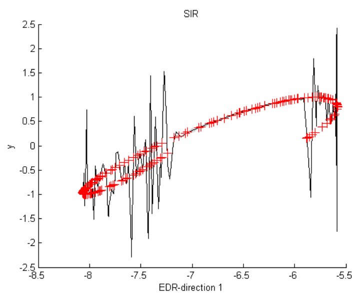
(b) Data in HighD


(c) PCA

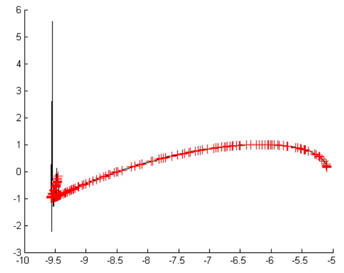

(d) NeuroScale


(e) GTM


(f) GPLVM



(g) SIR


(h) Using MAVE

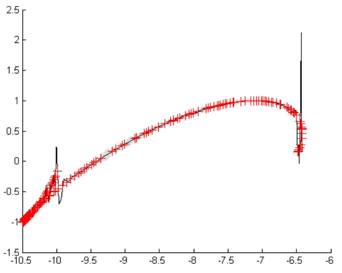

(i) KDR

Figure 6: Output mapping with the base data and the projections obtained using different dimension reduction methods for the non-linear (strongly non-linear) mapping dataset. Training data (plus), mean prediction using a trained GP on the test set (solid line) with error bars at 2 standard deviations (gray area).

is periodic, although in all cases the optimal emulator is obtained by emulating the true input, $\mathbf{X}$, to $\mathbf{y}$ simulator, as would be expected. Were the simulator function is not periodic, it seems that where the dimension reduction methods are able to correctly identify the manifold, the emulator in the reduced dimension input space, $\hat{\mathbf{X}}$, performs better than using the high dimensional inputs $\tilde{\mathbf{X}}$. These experiments have thus far only considered the high dimensional space to have up to 100 dimensions, which is still relatively low, and further work is required to assess where these methods might have more practical relevance in very high dimensions, although in these cases serious identifiability issues exist with respect to the projective mapping.

In this section the relative curvature measures of non-linearity presented by Bates and Watts (1980) are used to quantify the non-linearity of the manifold embedded in higher dimension. To obtain this non-linearity measure, the normal relative curvatures of the embedded manifold are calculated at each latent space point, $\mathbf{x}$, for different directions. Here, for each latent space point, the embedding curvature is calculated along 16 different directions distributed uniformly in the solid angle $360°$. The sum of the maximum of these directional curvature measures at each latent space point gives the total curvature of the manifold which is used as a measure to quantify the non-linearity of the entire embedded manifold. Note that the total curvature value does not give information on the shape of the manifold or the folding which might be critical for a dimension reduction method to work. Our aim in this analysis is to see how different dimension reduction techniques perform as the curvature measure (non-linearity) increases.

For the experiments presented in this section, again, we use the data generation setting presented in 6.1 to generate a dataset with 1 relevant input which was then embedded into a 3-dimensional space using an *almost linear* RBF mapping (so here $d = 1$ and $q = 3$). This simple example is useful to understand the behaviour of the dimension reduction techniques as we will be able to visualise the high-dimensional ($q = 3$) data and corresponding mapping ($d = 1$) obtained using the techniques we apply. 500 data points (N) were generated in the range on $[-6, 6]$ ($[l, u]$) so there is periodic behaviour as this simulator is a *sin* function. The goal is to demonstrate the behaviour of the projection obtained using the dimension reduction methods discussed in Section 4 and Section 5 as the amount of curvature in the embedded manifold changes. We chose one linear and one non-linear methods from Section 4 and 5. PCA and NeuroScale from Section 4 and MAVE and KDR from Section 5.

As discussed in Section 3 in some situations we may have observational data on the inputs, without corresponding simulator output. We can use this observed data to train unsupervised learning methods, which might give us a better projection. To simulate such a scenario, in this experiment the entire high-dimension training-set, $\tilde{\mathbf{X}}^{\mathtt{tra}}$, is used to train unsupervised dimension reduction methods (PCA and NS) but only 10% of the training-set, $\tilde{\mathbf{X}}^{\mathtt{tra}}$, is employed to train all the emulators and supervised dimension reduction methods (MAVE and KDR).

Figure 7 displays a scatter plot showing the relation between the total curvature of the embedded manifold in high dimensions and the emulator RMSE on the test set using the different dimension reduction methods for 30 different realisations of the non-linear embedding. It seems odd that the plot does not show any striking pattern, which suggests that the simple non-linearity measure of the manifold defined in Bates and Watts (1980) is not a reliable measure of how well non-linear projection methods are able to recover the true underlying data generating manifold. It is likely this is related to the observation that it is the shape of the manifold in higher dimension that is important, since a manifold that curves back on itself is very difficult to determine using the global mappings explored in this work. It is thus clear that further work is required to provide a guideline as to whether, or which, dimension reduction technique would work in a given setting. It is also interesting to note that in all but one case, using all inputs does not significantly worsen the prediction of the Gaussian process. This can be expected for two reasons; first, so long as the embedding into the high dimensional space is sufficiently smooth, so that the manifold does not cross or nearly touch, a Gaussian process can readily capture this high dimensional signal. Secondly, in this example period functions are employed for the simulator, and if the curvature is in phase with the oscillation then mapping this into the high dimensional space might even simplify the Gaussian process, making a stationary covariance function more appropriate, which we do observe in some cases.

Figure 8 and Figure 9 show plots of the high dimension data and projections obtained using different dimension reduction techniques for two different realisations, one with the lowest RMSE using Neuroscale and the other with the highest RMSE using Neuroscale respectively. It is easier to understand these results by studying the rotation of the embedded manifold in Figure 8(b) and Figure 9(b) as one can easily

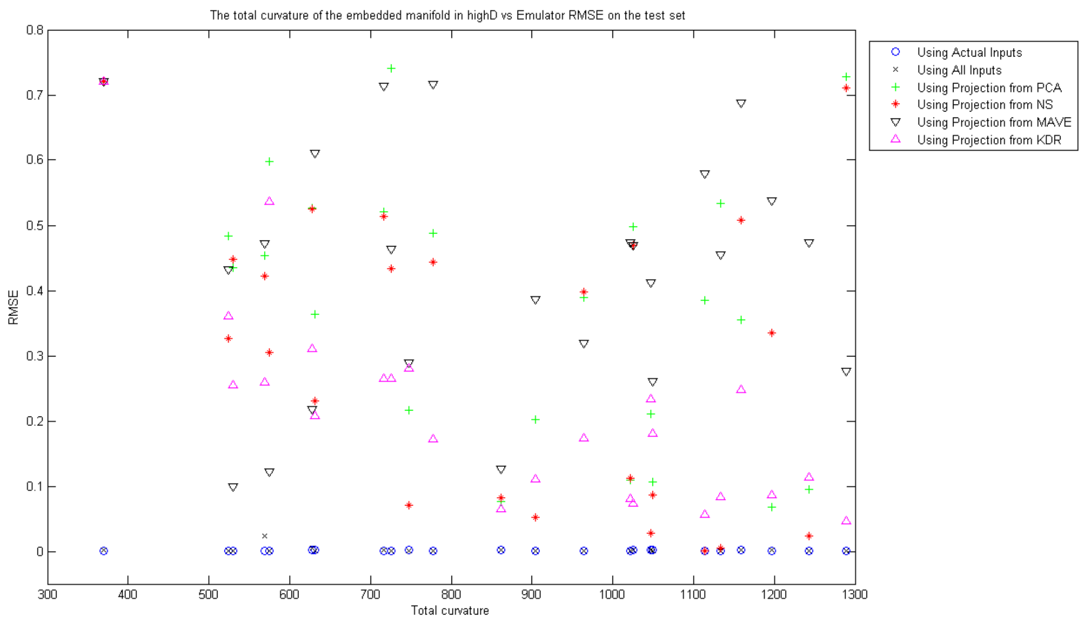


Figure 7: The total curvature of the embedded manifold in high dimensions vs emulator RMSE on the test set, for the set-up described in the text.

observe that even though the embedded manifold in Figure 8(b) has higher total curvature, dimension reduction methods can obtain a good lower dimension projection of the embedded manifold in Figure 8(b) than the embedded manifold in Figure 9(b). The embedded manifold in Figure 9(b) has a sudden turn and is twisted at one end which makes it difficult for the dimension reduction methods to find a good manifold. The corresponding curvature values are presented in the caption of the respective figures.

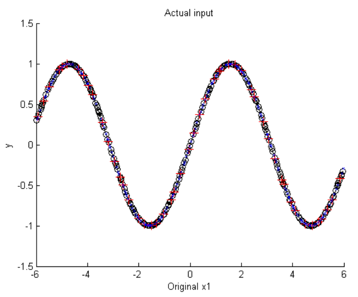
The results in Figure 8 illustrate that where it is possible to capture the manifold, then this knowledge can be used to make more accurate predictions than using the high dimensional data directly in the emulator inputs. Together with the significant advantage in being able to design experiments far more robustly in lower dimensional spaces, this result seem to have some promise, however in the majority of the cases studied in writing this report, the dimension reduction methods failed to capture the manifold. When this occurs, as can be seen, for example, in Figure 8(c) where PCA fails to capture the mapping, the resulting emulator has very inappropriate estimates of the covariance function parameters, largely due to the apparently multi-valued emulator output due to the poor approximation of the manifold. The most obvious method to address this would be to improve the fitting of the manifold, but this is a very complex problem, and there are no general solutions.

7 Discussion on open questions

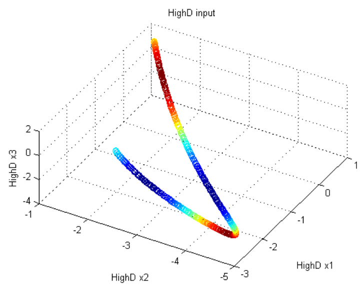
In this section we discuss some open questions which require further work:

Intrinsic dimensionality : The biggest assumption in the experiments presented here was knowledge of the correct intrinsic dimensionality. Determining intrinsic dimensionality is a fundamental and difficult problem. Cross validation is not a very appealing option as most of the dimension reduction methods are computationally expensive. Methods such as Minka (2000) can provide a mechanism to estimate bounds on the intrinsic dimensionality but for all methods we are aware of these work only for linear mappings.

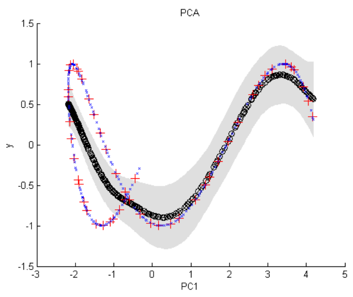
Inverting the mapping : As discussed in Section 2, one of the biggest advantages of dimension reduction prior to emulation could be in sequential experimental design. For this it is necessary to invert the estimated dimension reduction mapping so that design can be carried out in the reduced



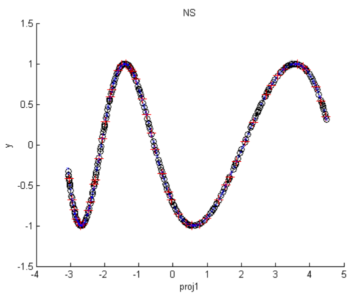
(a) Relevant input vs output output



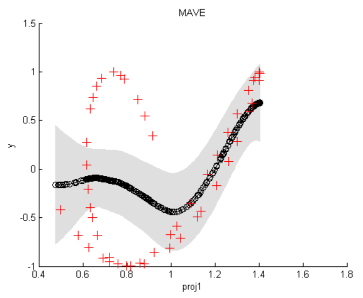
(b) Data in HighD



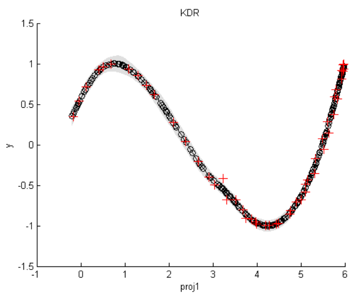
(c) PCA



(d) Neuroscale

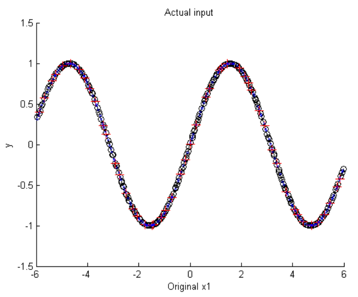


(e) Using MAVE

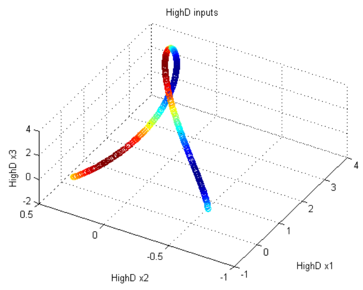


(f) KDR

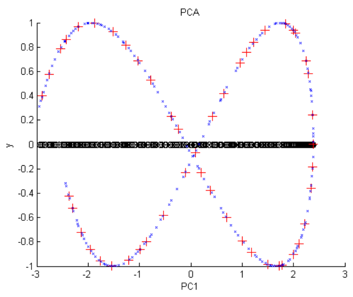
Figure 8: Output vs the base data and the projections using dimension reduction methods for the realisation with lowest RMSE using NS (total curvature of this realisation is 1113.6). GP training points (plus), extra training points used to train the unsupervised dimension reduction methods (crosses), mean prediction using a trained GP (○ and solid line) with error bars at 2 standard deviations (gray area).



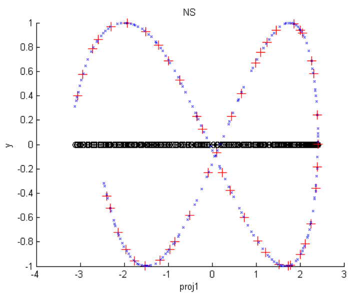
(a) Relevant input vs output output



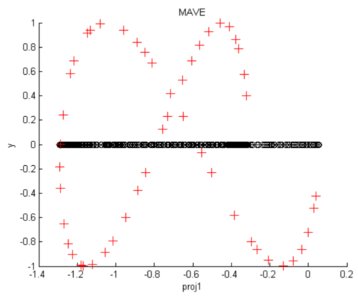
(b) Data in HighD



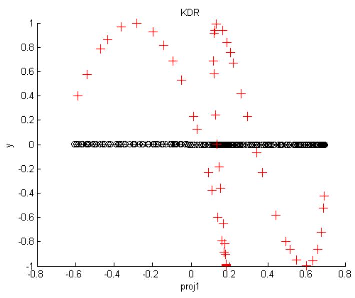
(c) PCA



(d) Neuroscale



(e) Using MAVE



(f) KDR

Figure 9: Output vs the base data and the projections using dimension reduction methods for the realisation with highest RMSE using NS (total curvature for this realisation is 369.6). GP training points (plus), extra training points used to train the unsupervised dimension reduction methods (crosses), mean prediction using a trained GP (○ and solid line) with error bars at 2 standard deviations (gray area).

dimension space and then mapped to the high dimensional simulator input space. Not all methods can provide such a one to one invertible mapping. Having a mapping both ways might be useful for other tasks too, so this avenue is one we intend to explore further.

Initialisation and optimisation : It is interesting to note that initialisation of algorithms which use non-linear models, such as Neuroscale, GTM, GPLVM, KDR, is an important issue. Models such as these, which are not initialised correctly can readily become stuck in local minima of their cost function (either likelihood, or posterior measure surfaces). In fact, the reason why the results of PCA and Neuroscale on complex embeddings, such as Figure 9, are sometimes identical is because Neuroscale is initialised using PCA and thus becomes stuck in a local minima very close to that point. We need to explore initialisation strategies which take account of the problem at hand, to find a more robust method for finding a global minima in the cost function. Alternatively we could employ non-deterministic optimisation methods, such as simulated annealing (Kirkpatrick et al., 1983) to explore the parameter space of the models more thoroughly.

Realistic experimental set-up : It is important that we apply these dimension reduction methods in a more realistic emulation setting and also understand the results thoroughly. We are seeking for a real-life model for which we can test our ideas. At present an attractive compromise might be to create a more realistic toy model based on the 2D advection / dispersion of a contaminant based on a simple, yet extensible flow model. The aim would be to write the dynamical model such that it could provide a test bed for methods across MUCM including data assimilation and in particular dimension reduction.

Scaling / limitations of dimension reduction techniques : Since the goal of MUCM is to deal with complex models, we need to test these methods for their scalability. It is important to know how far we can go with each method. One of the biggest challenges in testing these methods for for very high-dimensional data is understanding the results. It is not easy to understand why a method fails if we do not know the exact shape and folding of the embedded manifold in high dimension. Additionally, methods such as GTM have computational restrictions that mean they are only applicable where the intrinsic dimensionality is less than 3. Moreover, frequently in the emulation setting, one has to work with $O(100)$ data points, although Bob Parish from Energy SciTech suggested this is really no longer the case, and increasingly more runs are possible for some models. It can be difficult for a density modelling based dimension reduction method such as GTM to learn an effective manifold from such a small number of data points. We foresee a set of tools being developed to address those cases that can be treated, but there will remain problems which are not able to be reduced in dimension.

Validation method : In these experiments, RMSE values on test set obtained using an emulator trained in high-dimension performed almost as well as the emulator developed using the real inputs and almost always outperformed emulators developed using the projection obtained dimension reduced inputs. It seems that Gaussian processes generalise well into high dimensions, occasionally performing better in high dimensions. Indeed this almost suggests that Gaussian processes might be made more accurate by projecting the data into higher dimensions, not lower, with the projection defined so that the covariance is most simple; i.e. stationary and isotropic, very much in the spirit of Sampson and Guttorp (1992), although this clearly has some danger of over-fitting. It does emphasise that the main benefit of being able to define a lower dimensional manifold is the more efficient designs this might facilitate and the more sharp estimates of the hyper-parameters one might expect, thus making the fixing of these less problematic in the Bayesian treatment.

Estimating the mapping and covariance jointly : At present the mapping to lower dimensions is unconstrained; it would be interesting to add a constraint that the covariance of the emulator process in the lower dimensional space was stationary, in a manner similar to the treatment in Sampson and Guttorp (1992). Much work remains to be done, but such a method might have several attractions. It would also be interesting to relate the methods to the *outer product emulator* formulation proposed by Jonty Rougier (personal communication).

8 Summary

This reports explores the applicability of dimension reduction treated as a preprocessing step applied before emulation. We considered popular unsupervised and supervised dimensionality reduction methods, providing a top-level decision tree to help a modeller to choose appropriate dimension reduction approach for the problem at hand. Though non-linear dimension reduction methods such as Neuroscale and KDR gave good results for the simple models presented here, their applicability is subject to many caveats. In particular, as the complexity of the embedding grows, the non-linear methods become increasingly likely to fail, however emulation in the high dimensional space remains possible, if slightly sub-optimal. Apart from the knowledge of the intrinsic dimensionality, one of the important factors which decides whether dimension reduction prior to emulation will be useful for a particular problem at hand is the shape and the form of the embedded manifold in higher dimension space. In further work we will address the issues raised above. It is not yet clear whether the methods for dimension reduction explored in this report are applicable to real high dimensional models. We are currently working with the advisory panel members and other MUCM participants to define which models might be relevant and will shortly start to address dimension reduction in inputs and outputs for these models.

References

Antoniadis, A., S. Lambert-Lacroix, and F. Leblanc (2003). Effective dimension reduction methods for tumor classification using gene expression data. *Bioinformatics 19*(5), 563–570.

Bates, D. M. and D. G. Watts (1980). Relative curvature measures of nonlinearity. *Journal of the Royal Statistical Society. Series B (Methodological) 42*(1), 1–25.

Belkin, M. and P. Niyogi (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation 15*(6), 1373–1396.

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

Boukouvalas, A., , and D. Cornford (2007). Screening strategies for high dimensional input spaces. Technical report, Neural Computing Research Group, Aston University.

Boukouvalas, A., D. M. Maniyar, and D. Cornford (2007). Dimensionality reduction in complex models. Technical report, Neural Computing Research Group, Aston University.

Chiaromonte, F. and R. D. Cook (2002). Sufficient dimension reduction and graphics in regression. *Annals of the Institute of Statistical Mathematics 54*, 768–795.

Comon, P. (1994). Independent component analysis, a new concept? *Signal Processing 36*(3), 287–314.

Cook, R. D. and X. Yin (2001). Dimension reduction and visualization in discriminant analysis (with discussion). *Australian & New Zealand Journal of Statistics 43*, 147–199.

Donoho, D. and C. Grimes (2003). Hessian eigenmaps: locally linear embedding techniques for high dimensional data. *Proc. of National Academy of Sciences 100*(10), 5591–5596.

Fukumizu, K., F. R. Bach, and M. I. Jordan (2004). Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research 5*, 73–99.

Fukumizu, K., F. R. Bach, and M. I. Jordan (2006). Kernel dimension reduction in regression. Technical report, Department of Statistics, University of California, Berkeley.

J. B. Tenenbaum, V. d. S. and J. C. Langford (2000). A global geometric framework for nonlinear dimensionality reduction. *Science 290*(5500), 2319–2323.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi (1983). Optimization by simulated annealing. *Science 220*(4598), 671–680.

Lawrence, N. D. (2005). Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research 6*, 1783–1816.

Li, B., H. Zha, and F. Chiaromonte (2005). Contour regression: A general approach to dimension reduction. *The Annals of Statistics 33*, 1580–1616.

Li, K. C. (1991). Sliced inverse regression for dimension reduction (with discussion). *Journal of the American Statistical Association 86*(414), 316–342.

Li, K. C. (1992). On principal hessian directions for data visualization and dimension reduction: Another application of steins lemma. *Journal of the American Statistical Association 86*, 326–342.

Lowe, D. and M. E. Tipping (1997). Neuroscale: Novel topographic feature extraction with radial basis function networks. *Advances in Neural Information Processing Systems 9*, 543–549.

Minka, T. (2000). Automatic choice of dimensionality for pca. In *Advances in Neural Information Processing Systems*.

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

Saltelli, A., K. Chan, and E. M. Scott (2000). *Sensitivity Analysis*. Wiley.

Sampson, P. D. and P. Guttorp (1992). Nonparameteric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association 87*, 108–119.

Stein, M. L. (1987). Large sample properties of simulations using latin hypercube sampling. *Technometrics 29*, 143–151.

Tipping, M. E. and D. Lowe (1998). Shadow targets: A novel algorithm for topographic projections by radial basis functions. *Neurocomputing 19*, 211–222.

Xia, Y., H. Tong, W. Li, and L. Zhu (2002). An adaptive estimation of dimension reduction space. *Journal of the Royal Statistical Society B 64*(3), 363–410.