Dimension Reduction for Multivariate Emulation

Alexis Boukouvalas, Dan Cornford

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

Abstract
This article discusses the dimensionality reduction methods investigated during the Predicting Uncertainty in Complex Models (PUCM) Research Playground. We provide an overview of the theory of the methods examined and apply them to the output of a climate model. Our goal is to examine dimensionality reduction methods within the wider context of multivariate emulation as a way of handling model complexity. The authors wish to thank the other PUCM researchers for the stimulating discussions around multivariate emulation and climate models.

Technical Report NCRG/2008/006

July 2008
List of Figures

1. The time series data used. The black coloured points are actual observations. 
2. Percentage of variance explained. Plot help decide how many principal components to keep for ALL time series? In this example we see that we three principal components we can explain 95% of the variance. 
3. Factor Loadings for first 5 principal components. We can see temperature data ignored until the last principal component. 
4. PCA on SCM data using 5 PCs. Data has been standardised to the unit hypercube $[0, 1]^d$. Left column: Reconstructed data in blue plotted with the original data set in red. Right column: Reconstruction Error. Note the different scale on the MOC residuals. 
5. PCA on SCM data using 20 PCs. Data has been standardised to the unit hypercube $[0, 1]^d$. Left column: Reconstructed data in blue plotted with the original data set in red. Right column: Reconstruction Error. Note the different scale on the MOC residuals. 
6. Reconstruction of MOC time series data using 2 source ICA. In (c) we show the original set vs the reconstructed set with the straight line denoting a perfect fit. 
7. Visualisation of the mixing and source matrices for a 2 source ICA applied to the MOC time series. 
8. Mean and maximum residual error using different number of sources for ICA on the MOC data. 
9. Reconstruction of MOC time series data using 15 source ICA. In (c) we show the original set vs the reconstructed set with the straight line denoting a perfect fit. 
10. Visualisation of the mixing and source matrices for a 15 source ICA applied to the MOC time series. 
11. The MOC data for D1 (red crosses) plotted alongside the 8 centres (black solid line) for an 8 component mixture of PCA model. 
12. The reconstructed data and centre for each of the 8 components on the MOC dataset. Each component is 8 dimensional. 
13. Likelihood of 8 Mixture PCA model on MOC data (left) and reconstruction error (right). 
14. Drawing 20 samples from a 2 Mixture 2 Dimension PCA model fitted on the MOC data. Solid lines represent the mean of each component. 
15. Two dimensional latent space using GPLVM on the temperature time series from the SCM dataset D1. Half the data set was used for training (red) and the other for validation (blue). We can clearly see that both sets lie on the same region of the latent space. The greyscale denotes the uncertainty of the mapping, with darker regions have higher uncertainty. Note that in standard GPLVM the variance is taken to be identical for all output dimension allowing visualisation of the uncertainty in the latent space. 
16. GPLVM on temperature data. The residuals from the training set (red) and test set (blue) are shown. 
17. MOC time series data from dataset D2 used for the experiment in Section 10. 
18. Fit of 4 Dimensional PCA (a), 4 Mixture Model 1 dimensional PCA (b), 2 Mixture Model 2 dimension PCA (c), 3rd order Polynomial (d) and 4 source ICA (e) on MOC time series data. The black lines in (b) and (c) are the centers of the mixture components. The red lines are the original dataset and the other colours denote reconstructed time series. 
19. Residuals from fit of 4 Dimensional PCA (a), 4 Mixture Model 1 dimensional PCA (b), 2 Mixture Model 2 dimension PCA (c), 3rd order Polynomial (d) and 4 source ICA (e) on MOC time series data. The residuals are $Y - Y_{reconstructed}$. 
20. Residual error across 20 iterations using random initialisation. Output has been rescaled to $[0,1]$ so interpretation of the residuals is facilitated. Plotted are PCA using four principal components, mixture of PCAs with 2 centres each 2 dimensional, mixture of PCAs with 4 centres each 1 dimensional, third order polynomial regression (resulting in a 4 dimensional latent space since the bias term is included) and a 4 source ICA.
1 Introduction and Motivation

This report focuses on multivariate emulation and explores whether it might be possible to simplify emulation in this context using dimension reduction to reduce the number of ‘coupled emulators’ that need to be developed. Coupled emulation is essentially multivariate emulation; the main difference is one of approach. Typically in multivariate emulation one seeks to jointly model the distribution of a vector of outputs. When applying ‘coupled emulation’ one seeks to modify the outputs such that the emulation can be done independently in the modified outputs, which can then be recombined to form the joint distribution over the outputs in a post processing step. In this report we focus on the dimension reduction methods that will facilitate coupled emulation and screening. In the work it has become clear that coupled emulation remains an unsolved problem. Further work is required to explore better how to pre-process the outputs such that the emulators can be estimated independently and subsequently combined.

In particular, we present work done in the PUCM Research Playground with regards to dimension reduction for emulation of climate models. Some of the methods we present here have also been examined in (Maniyar et al., 2007) where the focus was on the dimensionality reduction of input spaces. Looking at output spaces allows for more flexibility in our methods.

Our goal is to reduce the dimensionality of output spaces in order to build effective emulators in the reduced spaces. We use a simple climate model provided by Nathan Urban (Pennsylvania State University) for our examples. The analysis is focused on time series data but we expect these methods will naturally extend to spatio-temporal data.

The issue at hand is that we can rarely directly emulate high-dimensional data of multiple types. A review of multivariate emulation can be found in (Fricker, 2008). In summary, current multivariate emulation approaches can be categorised as:

- **Independent.** In this approach we build \( k \) independent emulators with no correlation between output types being captured.
- **Separable.** In this approach we assume all output types have the same length scales. The variance matrix decomposes into a Kronecker product resulting into substantial computational savings. We expect that capturing the correlation among outputs allows prediction using fewer training examples compared to treating the outputs as independent.
- **Linear Model of Coregionalisation (LMC).** Each output \( k \) is expressed as a linear combination of \( p \) independent kernels. Typically the same number of kernels as outputs is used, i.e. \( p = k \). The problem is the size of the variance matrix to invert: \((n_x \times n_t \times k)\).

where \( n_x \) the number of input points, \( n_t \) the number of discrete time steps and \( k \) output types.

The LMC approach is the most flexible but quickly becomes unmanageable due to the size of the variance matrix to be inverted. Separability on the output types assumes the same length scales across outputs which in our opinion is too restrictive for most realistic examples.

For climate output data, it is known that the outputs are not independent. Not only are spatial fields and time series data correlated but functional dependencies are known to exist across output types. For example, in the \( C - GOLDSTEIN \) model both humidity and temperature are output with the latter having a very strong effect on the absolute value of the former.

Our approach is to use dimension reduction methods on individual output types and subsequently apply a flexible multivariate emulation method such as the LMC on multiple output types, each represented by a lower dimensional manifold.

2 Notation for Dimension Reduction

Prior to the discussion of the methods and results obtained, we introduce the notation used in this report.

The goal of dimension reduction is to discover a low dimensional manifold \( x \) that explains data sampled from a high dimensional space \( y \) well-enough given a criterion. In this report we consider only spaces of real numbers, i.e. \( y \in \mathbb{R}^D \) and \( x \in \mathbb{R}^M \) and for the transformation to be worthwhile \( M \ll D \).

The data we examine in this report are always time series which can be expressed as : \( y^i = [y_1, y_2, \ldots, y_{n_t}] \), where \( n_t \) the number of time steps for model run \( i \). Given \( n_x \) evaluations of the model,
the corresponding design matrix is then denoted by \( Y = [y^1 \ldots y^n]^T \) which is an \( n_x \times n_t \) matrix. The design matrix for the latent space is \( X = [x^1 \ldots x^n]^T \) and is an \( n_x \times M \) matrix where \( M \) is the latent space dimensionality.

In general, any dimension reduction method examined in this report can be written as:

\[
y^i = f(x^i, w) + \epsilon
\]

where \( w \) the parameters for the reduction model and \( \epsilon \) a model discrepancy.

For example, in PCA (Section 5) \( f(x^i, w) \) is a linear function in both \( x \) and \( w \). Further, the model discrepancy for all methods examined is assumed to be isotropic Gaussian, i.e., \( \epsilon = N(0, \sigma^2 I) \).

If no noise is assumed, \( \epsilon = 0 \), and the data truly lies in a lower dimensional manifold then it is possible to construct a perfect reduction method, \( y^i = f(x^i, w) \) where no information is lost in the transformation. However we typically always assume the presence of noise in the transformation because we either truly believe the data is noisy or that the reduction technique is not flexible enough to model the true manifold and the noise \( \epsilon \) captures unresolved signal.

3 The Simple Climate Model (SCM)

The model used in our examples is the simple climate model provided by Nathan Urban. This model was chosen to use in our investigations because it is fairly simple and fast to run but exhibits sufficiently realistic behaviour similar to more complex model that it makes a useful test bed for emulation and dimension reduction methods.

The configuration used for the experiments herein is shown in Table 1. The model was run in two configurations: in the D1 setup all 8 of the input model parameters were varied while in the second configuration D2 3 of the model input were kept fixed. This was done after a determination was made by the model expert that modifying the extra 3 parameters was scientifically not interesting.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Design points</th>
<th>Input Parameters</th>
<th>Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>986</td>
<td>8</td>
<td>Single Latin Hypercube</td>
</tr>
<tr>
<td>D2</td>
<td>343</td>
<td>5</td>
<td>2 Latin Hypercubes (243 and 100)</td>
</tr>
</tbody>
</table>

Table 1: Datasets obtained from the SCM and used for the experiments in this report.

In addition to the model input parameters, a forced time series input is available for each model run. The forced input is concentration of \( CO_2 \) in the atmosphere. The forcing input time series can be generated stochastically if the effect of volcanoes is simulated thus making the model stochastic in nature. In our experiments however we have chosen to use deterministic forcings since emulation of deterministic models is a more established research area. However stochastic model dimension reduction offers interesting challenges for further research.

Four time series outputs are provided: temperature, \( CO_2 \), ocean heat and meridional overturning circulation (MOC) strength. None of the outputs are spatially resolved and are global averages. The time series is discrete at yearly intervals starting at year 1850 and ending in 2300 giving a total of 451 time steps for each output type.

4 Overview of Dimension Reduction

Dimension Reduction methods can be broadly classified as:

- **Globally linear.** This class of methods, assume the data lie on a hyperplane. Examples include Principal Component Analysis (PCA) (Section 5), Independent Component Analysis (ICA) (Section 6) and Factor analysis. Expressing them in a generative setting they can be described as:

\[
y = Wx + \mu + \epsilon
\]

where \( x \) the latent space representation, \( W \) the projection matrix, \( \mu \) is the mean of the data in the \( Y \) space and \( \epsilon \) the noise model. The difference between PCA and Factor analysis is on the assumed
noise model. PCA assumes isotropic noise $\epsilon = N(0, \sigma^2 I)$ while in Factor analysis an independent noise model is assumed $\epsilon = N(0, \text{diag}(\sigma^2_1, \ldots, \sigma^2_M))$. In ICA the $W$ is non orthogonal and the projection attempts to make latent dimensions statistically independent.

- **Locally Linear**: Mixtures of linear models. Examples are mixtures of PCA (Section 7) or ICA.
- **Parametric Latent Variable Models**, e.g. polynomial regression (Section 9). This class of methods are still linear in the parameters but non-linear in inputs, i.e. can be written as:

$$y = \sum_{i=1}^{M} w_i f_i(x_i) + \mu + \epsilon$$

- **Non-Linear**: Gaussian Process Latent Variable Model (GPLVM). A Gaussian Process is used to define the mapping from latent space $X$ to feature space $Y$. Using Back Constraint (e.g. a radial basis function) an inverse mapping is guaranteed to exist (Section 8).

## 5 Principal Component Analysis

Principal Component Analysis (PCA) is a popular method of dimension reduction and is commonly defined in two equivalent ways (Bishop, 2007):

- The orthogonal projection of the data onto a lower dimensional linear space, known as the principal subspace, such that the variance of the projected data is maximised.
- The linear projection that minimises the average projection cost, defined as the mean squared distance between the data points and their projection, i.e. the reconstruction error.

Standard PCA involves the following steps:

1. Compute the sample mean $\mu$ and covariance $S$.
2. Decide on the dimensionality $M$ of the principal subspace. See Section 5.2 for a discussion on how to select $M$.
3. Compute the $M$ eigenvectors of $S$ corresponding to the $M$ largest eigenvalues. A full eigendecomposition of the covariance matrix $S$ is not needed and should be avoided since it is an $O(D^3)$ operations where $D$ the dimensionality of $y$. (Bishop, 2007) suggests using the power method that scales $O(MD^2)$ or using an EM algorithm. Alternatively a Singular Value Decomposition of the input matrix $X$ can be performed directly:

$$Y - \mu = U \times P \times V^T,$$

where $P = \text{diag}(s_1, \ldots, s_n)$ the singular values and $V$ the eigenvectors. Note that $\Lambda = \frac{1}{n-1} \times S^2$ where $\Lambda$ the eigenvalues of $S$ and $n$ the number of data points.

4. Project the data

$$y_n = U_M^T(x_n - \mu_x),$$

where $U_M$ is the $D \times M$ matrix of the first $M$ eigenvectors.

### 5.1 Definitions

The term **factor loadings** refers to the elements of $U_M$ (see equation (1)). For a single principal component, each element can be directly interpreted as the weighing of that input dimension in the linear latent space.

**Factor score** is defined as the projected data onto the principal subspace, i.e. $y_n$ in equation (1).
5.2 Dimensionality of subspace

Determining the intrinsic dimensionality for PCA is a key choice in the usage of the algorithm in any realistic situation. Traditionally, the percentage of variance explained has been used as a heuristic measure. This is computed by ranking the eigenvalues in magnitude and performing a running average for each eigenvalues by summing all the eigenvalues of greater magnitude and dividing by the sum of all eigenvalues. An example of the application of this heuristic is shown in Figure 2.

Recently more elaborate approaches have been suggested and a review is given in (Minka, 2000).

5.3 Uncorrelatedness vs Independence

PCA rotates the data in such a way as to make the factor scores uncorrelated. This can easily be demonstrated (see (Bishop, 2007)) by considering the covariance of the rotated space:

\[
S_y = \frac{1}{N} \sum_{n=1}^{N} y_n y_n^T = U^T S U = \Lambda
\]

since \(SU = UA\). Hence \(S_y\) is diagonal proving all latent dimension are uncorrelated.

Uncorrelatedness does not imply however independence.

As is shown in (Hyvarinen, 1999), uncorrelatedness for any two variables \(y_i, y_j\) can be expressed as:

\[
E[y_i y_j] - E[y_i]E[y_j] = 0
\]

whereas statistical independence is equivalent to

\[
E[g_1(y_i)g_2(y_j)] - E[g_1(y_i)]E[g_2(y_j)] = 0
\]

for any measurable functions \(g_1\) and \(g_2\) which is an obviously much stronger assumption.

However when \(y_i, y_j\) have a joint Gaussian distribution uncorrelatedness is equivalent to independence.

5.4 Experiment: PCA on time series data

We initially implemented PCA on the simple climate model dataset D1 shown as four time series in Figure 1. In this experiment we apply PCA on the entire data set. Let \(n_x\) the number of model runs and \(n_t\) the number of time steps. Then the design matrix for this experiment can be written as the concatenation of the four individual output time series, i.e. \(X = [X_{moc}, X_{co2}, X_{oceanHeat}, X_{temperature}]\) where \(X_{moc}, X_{co2}, X_{oceanHeat}, X_{temperature}\) are \(n_x \times n_t\) matrices. Thus \(X\) is a \((4 \cdot n_x) \times n_t\) matrix. In our experiment \(n_x = 986\) and \(n_t = 451\).

In order to get meaningful results using PCA on multiple output types which are in different units of measurement, all output types were rescaled to lie on the unit hypercube. Each output type (e.g. MOC) was rescaled using:

\[
X_{\text{rescaled}} = \frac{X - \text{min}(X)}{\text{max}(X) - \text{min}(X)}
\]

If this is not done, PCA will place greater emphasis on the dimensions with the largest unit magnitudes which may be appropriate in some scenarios. Other transformations are also possible. For example each output could be rescaled using the minimum and maximum of all outputs and thus retain relative variances between outputs.

The cumulative eigenspectrum shown in Figure 2, shows that using the first 3 principal components explains 95% of the total variance. In our experiment however we have used 5 principal components in order to capture the temperature data in the loadings. As is clear by looking at the factor loading in Figure 3 the principal components place greater emphasis on the moc time series data since it’s the most complex to describe (has the highest variance). As a result all other time series are essentially ignored. In particular the temperature data is not used until the 5th principal component.

In Figures 4 and 5 the reconstruction error is plotted for all four time series using 5 and 20 principal components respectively. The reconstruction error is computed by first projecting the entire data set using only the first \(M\) eigenvectors:

\[
\hat{Y} = ((Y - \mu) \times U_M) \times U_M^T + \mu
\]
Figure 1: The time series data used. The black coloured points are actual observations.

Figure 2: Percentage of variance explained. Plot help decide how many principal components to keep for ALL time series? In this example we see that we three principal components we can explain 95% of the variance.
Figure 3: Factor Loadings for first 5 principal components. We can see temperature data ignored until the last principal component.

The reconstruction error is the difference between the reconstructed data $\hat{Y}$ and the original data set $Y$. There is clear structure in the residuals for all time series. The most significant errors appear for the MOC time series data. The MOC data has some features that make it particularly difficult for a linear projection method such as PCA:

- There is a discontinuity at the point of collapse for each model run where the moc strength goes to zero and does not recover.
- MOC strength is a strictly non-negative quantity. This constraint is violated by the reconstructed time series as can be seen in Figure 4.

6 Independent Component Analysis (ICA)

Independent Component Analysis is a popular technique to separate a signal into multiple statistically independent sources. For a review of ICA methods see (Hyvarinen, 1999).

Unlike PCA, the sources are not orthogonal, rather the sources are chosen to be statistically independent (see Section 5.3). ICA can be described generatively as:

$$Y = A \times X + \epsilon$$

where $A$ is known as the mixing matrix, $X$ the sources matrix and $\epsilon$ a zero mean gaussian noise model. Unlike PCA where the eigenvectors are constrained to have norm 1, in ICA both $A$ and $X$ are unconstrained making the interpretation of the loadings more challenging. In PCA, constraining the norm of the eigenvectors ensures that we can directly interpret the factor loadings (see Section 5.1) as relevance values (see for example Figure 3). In ICA however both the mixing and source matrices have to be examined to determine relevance.

In our experiments we have used Mean Field ICA (Pedro A.d.F.R. Hjen-Srensen and Hansen, 2002) which is a noisy version of ICA and allows constraining of both the mixing matrix and signals to be positive. This ensures that the reconstruction will be non-negative which is required for the MOC time series.
Figure 4: PCA on SCM data using 5 PCs. Data has been standardised to the unit hypercube $[0, 1]^d$. Left column: Reconstructed data in blue plotted with the original data set in red. Right column: Reconstruction Error. Note the different scale on the MOC residuals.

Figure 5: PCA on SCM data using 20 PCs. Data has been standardised to the unit hypercube $[0, 1]^d$. Left column: Reconstructed data in blue plotted with the original data set in red. Right column: Reconstruction Error. Note the different scale on the MOC residuals.
We conclude this section with an example of applying Mean Field ICA to the MOC time series dataset D2. In Figure 6 we see the reconstruction using 2 Independent Components (ICs). The reconstructed MOC data is non-negative but two ICs are clearly not enough to capture this dataset. The reconstruction shows significant differences to the original dataset.

To further understand the workings of ICA we can look at the mixing and source matrices shown in Figure 7. We see the components of the mixing matrix $A$ in subfigure 7(a) that capture two distinct aspects of the time series. The image plots of the matrices show the magnitudes of the coefficients. The second component of the $A$ matrix is clearly dominant and places most significance in the earlier year of the MOC time series where most variability occurs. The magnitudes in the source matrix are smaller than the mixing matrix coefficients thus not invalidating our conclusion.

A method that would normalise one of the matrices would allow considerably easier interpretation of the ICA loadings similar to PCA and this remains an open question for future research.

Lastly, in order to determine how many sources to use in ICA the simplest approach is to measure the residual error for different number of components. The mean and maximum residual error for different number of sources is shown in Figure 8 where we can see that the minimum is attained when using close to 15 ICs. The corresponding reconstruction and mixing matrix when using 15 independent sources are shown in Figures 9 and 10 where we see significant improvement in the fit.

One suggestion that has been made is to apply PCA prior to ICA and utilise ICA only on the retained PCs. However prior to further experimentation, we should further examine the appropriateness of ICA in an emulation context. In particular, we do not see how statistical independence offers any benefits when using Gaussian Process emulators. Further, statistical independence of the data, does not imply independence conditional on model inputs.

Figure 6: Reconstruction of MOC time series data using 2 source ICA. In (c) we show the original set vs the reconstructed set with the straight line denoting a perfect fit.

Figure 7: Visualisation of the mixing and source matrices for a 2 source ICA applied to the MOC time series.
Figure 8: Mean and maximum residual error using different number of sources for ICA on the MOC data.

Figure 9: Reconstruction of MOC time series data using 15 source ICA. In (c) we show the original set vs the reconstructed set with the straight line denoting a perfect fit.

Figure 10: Visualisation of the mixing and source matrices for a 15 source ICA applied to the MOC time series.
7 Mixture of PCAs

An intermediate approach between linear and non-linear methods are mixture models. In (Tipping and Bishop, 1999), mixtures of PCAs were introduced building upon a probabilistic description of PCA.

Two steps are involved in the definition of the mixture model of PCAs:

- Partition the data into regions.
- Fit Principal Components in each region.

In the Probabilistic PCA framework a maximum likelihood framework using an EM algorithm estimates both.

Log Likelihood is:

$$L = \sum_{n=1}^{N} \ln \left( \sum_{j=1}^{M} \pi_j p(x_n | j) \right),$$

- where $p(x_n | j)$ is a PPCA model and $\pi_j$ is the mixing coefficient. Each component is described by a mean vector $\mu_j$, a projection matrix $W_j$ and an isotropic noise model $\sigma_j^2$.

In the E step the responsibility $p(j | x_n)$ is computed and in the M step the parameters for each component are reestimated.

In our experiments, all mixture components are constrained to have the same latent dimension although there is no inherent constraint in the algorithm to do so.

7.1 Experiment on MOC data

To demonstrate the basic algorithm, we applied it to the MOC data of dataset D1. An 8 dimension 8 component mixture model of PCAs was estimated on the MOC time series data. The centre of each component is shown in Figure 11. The reconstructed data and the centre of each component are shown in Figure 12. We see each mixture component can be related to similarly shaped time series.

![Figure 11: The MOC data for D1 (red crosses) plotted alongside the 8 centres (black solid line) for an 8 component mixture of PCA model.](image)

One issue in the implementation of the algorithm was numerical stability. To workaround these issues the time series was subsampled every three time steps, thus reducing the time series from 451 to 151 time steps. Also in the experiment the k-means algorithm as used to initialise the centres and initialisation proved quite important in the performance of the algorithm. Random initialisation gave on average
quite poorly behaved mixture models while k-means with significant optimisation (10 steps in a scaled conjugate gradient algorithm) gave more consistent results.

In Figure 12 the reconstructed data and centre for each of the 8 components on the MOC dataset. Each component is 8 dimensional.

Figure 12: The reconstructed data and centre for each of the 8 components on the MOC dataset. Each component is 8 dimensional.

In Figure 13 the likelihood and reconstruction error are given for different number of mixture and latent dimensionalities. Encouragingly, the likelihood surface corresponds well to the actual reconstruction error.

Finally we note that since mixture models use the probabilistic PCA formalism and are defined in probabilistic density terms it is possible to draw samples from a fitted model. An example is shown in Figure 14 for a 2 dimension 2 mixture PCA model fitted on the MOC data. Such sampling allows visualisation and better understanding of the behaviour of the fitted model. In this case the resulting samples are noisier than the underlying signals because we constrain the model to allow only 2 dimensions in each component and thus the remaining structure is modelled by noise.

In addition to having to estimate the intrinsic dimensionality for each component, a key question for mixture models is on how to decide on the number of mixtures to utilise. A fully Bayesian treatment is proposed in (Bishop and Winn, 2000), based on variational inference, that allows the number of components as well as the effective number of dimensionalities of the individual components, to be inferred from the data.

8 Gaussian Process Latent Variable Model (GPLVM)

The Gaussian Process Latent Variable Model (GPLVM) is an extension of the Probabilistic PCA framework allowing an arbitrary non-linear mapping defined by a kernel function from latent to data space.

For a detailed discussion of GPLVM see (Lawrence, 2004).

In GPLVM, a Gaussian Process is used to map latent space to data space. The GP has multiple outputs which are assumed independent conditional on the latent space:

$$p(Y|X) = \prod_{i=1}^{D} \frac{1}{(2\pi)^{D/2}|K|^{1/2}} e^{xp(-\frac{1}{2}y_i^tK^{-1}y_i)}$$

i.e. the GP can be expressed as a product of $D$ independent Gaussian Processes, where $D$ is the data space dimensionality.
Figure 13: Likelihood of 8 Mixture PCA model on MOC data (left) and reconstruction error (right).

Figure 14: Drawing 20 samples from a 2 Mixture 2 Dimension PCA model fitted on the MOC data. Solid lines represent the mean of each component.
Using a back constraint allows for an inverse mapping from feature to latent space (Lawrence, 2006). The back constraint is defined as a non-linear function (e.g. Multi-Layer Perception, Radial Basis Function of Kernel regression) that specifies a unique latent space position conditioned on a feature space data point.

Gaussian Process Dynamic Models (GPDM) extends GPLVM by introducing a GP dynamical model in latent space (Wang et al., 2008).

To test this method, we applied to the temperature time series of the D1 SCM data set. We initially examined only the simplest formulation of GPLVM without dynamics or back constraints.

In our experiment we split the data set in equally sized training and test sets. As we see in Figure 15, both training and test sets were mapped to the same region of the latent space. The residuals shown in Figure 16 further confirm that the algorithm has learned this data set well and has generalised.

However this data set is quite simple and can be adequately described using PCA with only a few PCs. Therefore further testing is needed on more challenging data like the MOC time series.

Interpretation of the mapping provided by GPLVM is quite difficult but quite flexible through the choice of kernel. Furthermore the GP mapping automatically offers an estimate of the reconstruction error and the work by (Wang et al., 2008) demonstrates how to incorporate a GP in the latent space and jointly optimise both the emulator GP and the latent-to-data space mapping GP.

However this can result in a quite complicated optimisation task with hard to interpret results and convergence issues. The method is quite powerful and warrants further investigation should simpler methods prove unsatisfactory on the desired model outputs.

![Figure 15: Two dimensional latent space using GPLVM on the temperature time series from the SCM dataset D1. Half the data set was used for training (red) and the other for validation (blue). We can clearly see that both sets lie on the same region of the latent space. The greyscale denotes the uncertainty of the mapping, with darker regions have higher uncertainty. Note that in standard GPLVM the variance is taken to be identical for all output dimension allowing visualisation of the uncertainty in the latent space.](image)

9 Parametric Latent Variable Models: Polynomial Regression

Visual inspection of the MOC time series data, suggests polynomial regression using a fairly low order polynomial should capture most of the features of the data.
The idea is to fit a parametric model to the time series and the emulator predicts the model parameters.

When fitting an $M^{th}$ order Polynomial for model run $i$:

$$P_i(t) = \sum_{n=1}^{M+1} a_n t^{n-1}$$

the emulator will predict the $a_n$ coefficients which form the latent space representation for each model run. Thus the effective dimensionality of the latent space is determined by the degree of the polynomial.

A polynomial fit allows us to model the collapse of the MOC explicitly:

- Fit polynomial on non-collapsed region.
- Root of polynomial in given domain, gives point of collapse.

Thus no discontinuity need be modelled.

In polynomial regression the model parameters are the polynomial coefficients and the dimension reduction function is linear in the coefficients but non-linear in inputs $t$. The emulator output will be the coefficients $a_n$.

10 Experimental Results on the MOC time series data

In this final experiment we have compared the behaviour or PCA, mixture of PCAs, the Polynomial Latent Variable Model and ICA on the MOC time series from dataset D2 shown in Figure 17. In particular for the experiment to be meaningful the same number of free parameters were used for each reduction technique, i.e.:

- PCA using 4 principal components,
- Mixture of PCAs with 2 centres each 2 dimensional
- Mixture of PCAs with 4 centres each 1 dimensional
- Third order Polynomial regression (4 coefficients including bias).
ICA with 4 sources.

The fit on a sample of model runs is shown in Figure 18. Note that the reconstructed runs for all PCA methods violate the non-negative constraint of the MOC time series data while the polynomial fit and ICA do not.

In Figure 19 the residual errors are plotted. We see structure in the residuals of all five reduction methods. The PCA methods show similar structure in the residuals while the polynomial model shows consistently large errors towards the tail of the time series, i.e. for model runs where the MOC collapses near the end of the time series or does not collapse at all. We believe this is due to the faster than polynomial decay of the MOC time series at these times. For ICA we also see larger error towards the end of the time series but also significant structure in the entirety of the time series. We thus conclude that for ICA significant signal has not been resolved.

In Figure 20 the mean and maximum residual error are shown for all methods across multiple iterations. We see that the polynomial method gives significantly and consistently smaller maximum error. The mean error is quite similar for all methods (the scale on the y-axis is $10^{-3}$).

Thus in our experiment, the polynomial model has provided the best fit for MOC data. We suggest that a different set of bases that can more accurately capture the trend of the time series after the 300th time step would perform significantly better than the polynomial model since the bulk of its errors occur in this region.

11 Uncertainty in Reduced Spaces

11.1 Propagating Uncertainty

Thus far our investigations have focused on the reconstruction error of different methods. However if the emulator is operating in a latent space, a specification on how to propagate the uncertainty of the emulator from the latent space to the data space is needed.

Let the input to the emulator be $z_i$, the model output time series $t_i = [t_{i1} \ldots t_{in}]$, the predicted time series $\hat{t}_i$ and the latent space representation $p_i$ and $\hat{p}_i$ respectively.

The predictive distribution of the emulator is $p(\hat{p}_i | z_i) = MVN(\mu_{\hat{p}_i}, \Sigma_{\hat{p}_i})$ where $MVN$ denotes a Multivariate Normal Distribution.
Figure 18: Fit of 4 Dimensional PCA (a), 4 Mixture Model 1 dimensional PCA (b), 2 Mixture Model 2 dimension PCA (c), 3rd order Polynomial (d) and 4 source ICA (e) on MOC time series data. The black lines in (b) and (c) are the centers of the mixture components. The red lines are the original dataset and the other colours denote reconstructed time series.

Figure 19: Residuals from fit of 4 Dimensional PCA (a), 4 Mixture Model 1 dimensional PCA (b), 2 Mixture Model 2 dimension PCA (c), 3rd order Polynomial (d) and 4 source ICA (e) on MOC time series data. The residuals are $Y - Y_{reconstructed}$. 

Figure 20: Residual error across 20 iterations using random initialisation. Output has been rescaled to [0,1] so interpretation of the residuals is facilitated. Plotted are PCA using four principal components, mixture of PCAs with 2 centres each 2 dimensional, mixture of PCAs with 4 centres each 1 dimensional, third order polynomial regression (resulting in a 4 dimensional latent space since the bias term is included) and a 4 source ICA.

In PCA since the mapping is linear, the data space distribution is
\[ p(t_i|z_i) = MVN(\mu_B\hat{t}_i, B\Sigma_i B^T) \]
where \( B = U_M \), the matrix of eigenvectors.

In the case of polynomial regression since the mapping is still linear in the parameters (the polynomial coefficients) it turns out the same mechanism can be used with the appropriate \( B \) specified.

\[
\begin{pmatrix}
  t_1 \\
  t_2 \\
  \vdots \\
  t_n
\end{pmatrix}
_T =
\begin{pmatrix}
  t_1^M \\
  t_2^M \\
  \vdots \\
  t_n^M
\end{pmatrix}
_T \times
\begin{pmatrix}
  a_{M+1} \\
  \vdots \\
  a_1
\end{pmatrix}
_B
\]

The dimensions of the matrices are \((n_t \times 1) = (n_t \times (M + 1)) \times ((M + 1) \times 1)\). \( M \) is the order of polynomial. In vector notation:

\[ t_i = a_i^T B \]

Equation (2) is a particular example of a linear in the parameters model. We can extend the formulation to allow for arbitrary basis functions instead of just polynomials. Then we can see that algorithms such as Neuroscale (Lowe and Tipping, 1997) can be applied with propagation of uncertainty of the multivariate emulator still tractable without sampling.

However estimating the reconstruction error depends on matrix the form of matrix \( B \) - see section 11.2.

11.2 Reconstruction Error

In addition to propagating the uncertainty of the emulator, a more realistic uncertainty estimate in the data space will include an estimate of the reconstruction error incurred by the dimension reduction itself.

For PCA this turns out to be straightforward. The reconstruction error can be estimated as the average of the discarded eigenvalues which represent the discarded variance in the training data (Tipping and Bishop, 1999). Thus the total uncertainty in the data space is:

\[ N(A\mu_Y, A\Sigma_Y A^T + \Sigma_R) \]

where \( A \) is the mixing matrix and \( \Sigma_R \) is our estimate of the reconstruction error. For PCA \( \Sigma_R = \frac{1}{d-q} \Sigma_{i=q+1}^{q+d} \lambda_i \times I_d \)
However estimating the reconstruction error for other linear in parameters models depends on the form of matrix $B$ (equation (2)). If $B$ is linear in the inputs, then we recover PCA. For other algorithms such as Neuroscale presumably a numeric estimate using the training set could be used.

As was discussed in Section 8, when using the GPLVM algorithm, the uncertainty of the mapping GP could be used to estimate the reconstruction error.

### 12 Emulating in a Reduced Space

The training of the Gaussian Process emulator is done as follows:

1. The dimension reduction model is fitted to the training data so $y$ is reduced to $r$, i.e. $f(r, w) = y$.
2. With the same training data, fit on the emulator so $GP(x, \theta) \rightarrow r$.

Prediction is done as follows:

- For new input $x_*$, apply emulator $GP(x_*) \rightarrow r_*$
- Apply mapping $f(r_*, w) \rightarrow y_*$

For the polynomial latent variable model, the only parameter is the order of the polynomial and the time series steps. The coefficients $a_i$ are the reduced dimension representation of an output $y_i$.

Calibration should be performed on the data space rather than the latent space so reconstruction error is accounted for in the model parameters.

### 13 Summary and Open questions

In this report we have presented work done in the PUCM Research Playground using dimension reduction methods on climate model time series outputs. We have focused on simple methods such as PCA, ICA and Polynomial regression to get a good methodological grounding before looking into more advanced methods such as GPLVM. Further, these simpler methods offer easier interpretability of the resulting lower dimensional representation as well as allowing propagation of the emulator uncertainty and estimation of the reconstruction error.

In our experiments, ICA did not perform as well as the other methods. However other flavours of the ICA algorithm such as Fast ICA (Hyvarinen, 1999) may offer better performance. The mixture of PCAs method is promising as well as the polynomial regression method. Both types of method do better than standard PCA on the MOC time series data. However both require selection of model parameters such as the number of mixing components and the latent space dimensionality. For mixtures of PCAs (Bishop and Winn, 2000) present a variational framework to estimate both the number of mixtures and the dimensionality of each component. The robustness of this method is a potential issue and we plan to investigate further. Also this method as well as other automatic selection of dimensionality work that uses Bayesian model selection (e.g. (Minka, 2000)) to set the latent space dimensionality for PCA may be extendable to arbitrary fixed basis reductions such as the polynomial method presented in this report.

Several open questions remain:

1. **How should the forcing inputs be used in the latent space emulator or in the dimension reduction?** These forcing inputs are typically time series and significantly affect model behaviour. Furthermore the forcing time series may be generated stochastically (see Section 3).
2. **Do we need the inverse mapping in dimension reduction?** Linear methods such as PCA or even mixture of linear models where the data is clustered and assigned to different centers, allow an exact inverse mapping. However it is not clear such a mapping is needed when looking at output spaces.

   (a) For dimension reduction of input spaces or dynamic emulation where the goal is sequential design such a mapping appears necessary (see (Maniyar et al., 2007) for more details).
(b) Double counting since same training data is to fix reduction parameters \( w \) and emulator hyperparameters \( \theta \). If we wish to avoid the double counting and use separate data sets then an inverse mapping is required for the training set of the emulator.

(c) When doing validation, with no inverse map the validation has to performed in the data space \( Y \) thus not allowing the disambiguation of the error from the reduction mapping and the emulator error.

3. Propagating uncertainty. A linear map such as PCA allows the propagation of variance \( (\Delta \Sigma \Delta') \) whereas this has to occur through sampling for a general non-linear map. However if the non-linear map is probabilistic (e.g. GPLVM) then one option is to use the uncertainty associated with a mapping as an estimate of the reconstruction error.

4. Calculating reconstruction error. Methods to estimate the reconstruction error assume a specific noise model. For example in PCA, spherical noise is assumed for all dimensions. Incorporating more complex noise models, similar to methods used in calibration for representing model inadequacy, may be an extension to better capture the reconstruction error. Good estimation of the reconstruction error is necessary for realistic uncertainty estimation in the data space.

5. In the latent space, even for a deterministic computer model the projected observations must be considered noisy due to the information loss caused by the dimension reduction method. Therefore the emulator should not interpolate exactly.

6. Smoothness of mapping. Typically we assume smooth output surface for our emulator and even though the data space surface maybe smooth, the dimension reduction method may result in a latent space that is considerable less so, especially in the case of non-linear reduction methods.

Future work:

1. Apply reduction methods to spatio-temporal fields.

2. In collaboration with Tom Fricker and Richard Wilkinson:
   - Compare multivariate emulation on a reduced space vs independent emulators vs separable.
   - Calibration of a multivariate emulator using a more complex reduction than PCA.

References


