Smooth noisy PCA Using a First Order Roughness Penalty

Jakob Sigurðsson

A Thesis Presented in Partial Fulfillment of the
Requirements for the Degree
Master of Science in Electrical and Computer Engineering at the
University of Iceland
2011

Thesis Committee:
Assistant Professor Magnús Örn Úlfarsson, advisor
Professor Jóhannes R. Sveinsson, co-advisor
Professor Jón Atli Benediktsson

Háskóli Íslands / University of Iceland
Verkfræði- og náttúruvísindasvið / School of Engineering and Natural Sciences
VR-II, Hjardarhaga 2-6, IS-107 Reykjavik, Iceland
Phone + 354 525 4000
verkognatt@hi.is
www.hi.is

Abstract

Principal component analysis (PCA) and other multivariate methods have proven to be useful in a variety of engineering and science fields. PCA is commonly used for dimensionality reduction. PCA has also proven to be useful in functional magnetic resonance imaging (fMRI) research where it is used to decompose the fMRI data into components which can be associated with biological processes. In this thesis, a smooth version of PCA, derived from a maximum likelihood framework, is developed. A first order roughness penalty term is added to the log-likelihood function, which is then maximized for the parameters of interest with an expectation maximization (EM) algorithm. This new method is applied both to simulated data and real fMRI data.

Imposing smoothness is often justifiable. Natural signals are often known to be smooth and all recording devices are susceptible to noise. The proposed method imposes smoothness on the solution within the maximum likelihood framework.

My main contributions to this work has been to derive the proposed method and to develop a EM algorithm that finds the solution. Also, I developed a cross-validation method for determining the smoothness of the result and did a detailed comparison of the proposed method to other PCA methods.

Contents

List of Figures

Abbreviations

AIC	An Information Criterion
AU	Arbitrary Units
BIC	Bayesian Information Criterion
BLUE	Best Linear Unbiased Estimator
BOLD	Blood Oxygen Level Dependency
CRLB	Cramer-Rao Lower Bound
ECG	Electrocardiography
EM	Expectation Maximization
EMF	EM Functional
fMRI	Functional Magnetic Resonance Imaging
LS	Least Squares
ML	Maximum Likelihood
MLE	Maximum Likelihood Estimation
MSE	Mean Square Error
MVU	Minimum Variance Unbiased
nPCA	Noisy Principal Component Analysis
PCA	Principal Component Analysis
PCs	Principal Components
PDF	Probability Density Function
SVD	Singular Value Decomposition
WGN	White Gaussian Noise

Notations

$\mathbf{A}$	Matrices are denoted upper case bold
$\mathbf{a}$	Vectors are denoted lower case bold
a	Scalars are denoted lower case
$\mathbf{A}^\mathsf{T}$	The transpose of a matrix $\mathbf{A}$
$\lvert\mathbf{A}\rvert$	The determinant of a matrix $\mathbf{A}$
$\mathrm{tr}(\mathbf{A})$	The trace of a matrix $\mathbf{A}$
$d\mathbf{A}$	The first differential of a matrix $\mathbf{A}$
$\hat{\theta}$	An estimate of parameter θ
$\dim(\boldsymbol{\theta})$	The degrees of freedom for parameter $\boldsymbol{\theta}$
$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	$\mathbf{x}$ is a sample from a Gaussian dist. of mean $\boldsymbol{\mu}$, and covariance $\boldsymbol{\Sigma}$
$\mathbf{E}[\mathbf{A}]$	The expected value of $\mathbf{A}$
$\lVert\mathbf{A}\rVert_F$	The Frobenius norm of a matrix $\mathbf{A}$, $\lVert\mathbf{A}\rVert_F = \sqrt{\mathrm{tr}(\mathbf{A}\mathbf{A}^\mathsf{T})}$

Chapter 1

Introduction

Principal Component Analysis (PCA) [1], [2] involves a procedure that linearly transforms data into an orthogonal coordinate system such that the first component has greatest variance, the second component has next greatest variance and so on.

PCA is well established in signal and image processing. PCA is often used in postprocessing of data to condense information of a large set of correlated variables into few variables, while maintaining the variability in the data. PCA is for example used in this manner to reduce the dimensionality of multi- and hyperspectral data in remote sensing. PCA is also used medical imaging such as Electrocardiography (ECG) and fMRI analysis. PCA techniques have been used on ECG data for signal detection, compression and other analysis [3]. In fMRI analysis, PCA is, e.g., used as a feature extraction tool [4].

PCA can be expressed as the maximum likelihood (ML) solution of a signal model [5], which we call *noisy PCA* (nPCA), often also called probabilistic PCA (pPCA). A closed form solution is available, but it may also be solved with an expectation maximization (EM) algorithm [6]. Having a solution to PCA within the ML-framework gives the possibility of using statistical methods such as model selection methods, inference and gives a framework to build upon [7].

The reasoning behind adding a penalty term to the log-likelihood function is that data is often known to be smooth. By constricting the solution to be smooth we are able to incorporate this prior knowledge about the data into the solution. One example of a natural smooth signal is Blood Oxygen Level Dependency (BOLD) in fMRI.

The EM algorithm [8] is often an efficient way to compute ML estimates in the presence of missing or hidden (latent) data. The EM algorithm is an iterative process and consists of two steps, the expectation step and the maximization step (E and M). In the E-step, the latent data are estimated given the observed data and current estimate of the model parameters. In the M-step, the EM functional is maximized using the estimates of the data and parameters from the E-step. The EM algorithm is used here to maximize a penalized log-likelihood function.

The EM algorithm has been applied in a variety of fields. A large interest in the EM algorithm within the signal processing area is in maximum likelihood tomographic image reconstruction [9]. In [10], the EM algorithm is used as an inverse algorithm to estimate the non-stationary region boundaries using electrical impedance tomography. The EM algorithm is used to overcome uncertainties caused by Kalman-type filters due to inaccurate model selection.

Many optimization problems have constraints and the solutions which are obtained must satisfy these constraints. Penalty functions are put forth to replace the constrained optimization with less constrained conditions. This allows the use of unconstrained optimization techniques to solve constrained problems. Penalties are also used when optimization problems are ill-conditioned. The penalized solution should ideally converge to the original solution of the constrained problem. However, when using a penalty function a bias is introduced into the solution. A large penalty will result in a large bias and a small penalty a small bias. The penalty function grows when the constrains are violated and forces the merit function (e.g., log-likelihood function) to increase. When the constraints are not violated the penalty function does not grow.

In signal processing, penalty functions are commonly used. In compressed sensing, ℓ_1 penalties are known to promote sparseness [11]. Other penalties can also be used to encourage sparseness, such as the ℓ_0 penalty [12].

In this thesis a smooth version of nPCA is presented. A first order roughness penalty term is added to the log-likelihood function. By adding the penalty term, differences in neighboring values will be assigned a high cost, and roughness in the PCA solution will be discouraged. This is an addition to the basis expansion presented in [13] which is itself an extension to the maximum likelihood framework to PCA proposed by [5]. A regularization parameter affects the

amount of smoothing. Here, the parameter is chosen using a cross-validation (CV) method. Using simulated data, the results obtained using CV are compared to the results obtained using the parameter that minimizes the mean square error. The proposed method is compared in detail, both to normal PCA and nPCA using basis expansion. Simulated and real functional magnetic resonance imaging (fMRI) data are used in the comparison.

A closed form solution is not available for nPCA with a first order roughness penalty term, therefore the use of the EM algorithm is necessary.

The outline of this thesis is as follows. In Chapter 2, a short introduction to parameter estimation is given. Some examples are also given and estimators are examined, such as the maximum likelihood estimator. In Chapter 3, the EM algorithm is reviewed and a derivation of the algorithm is given. Chapter 4 is dedicated to PCA, and noisy PCA is detailed in Chapter 5. The basis expansion extension to nPCA is explained in Chapter 6. Finally, nPCA with a first order penalty term is detailed in Chapter 7. Results and comparison of the methods are given in Chapter 8. Conclusions are drawn and a discussion of further work are given in Chapter 9.

My main contributions have been to derive the proposed method and develop a EM algorithm that finds the solution. Also, I have developed a cross-validation method for determining the smoothness of the result and compared the proposed method to other PCA methods.

Chapter 2

Parameter Estimation

2.1 Introduction

Parameter estimation deals with estimating values of parameters based on measured data. The parameters of interest describe physical attributes of underlying systems.

The first step in parameter estimation is to mathematically model the data. Since data is inherently random it is described by its probability density function (PDF). In actual problems we are not given a PDF, we must choose one that is consistent with the data and is mathematically tractable. The Gaussian distribution is often used to model real world data. A simple mathematical model for data with noise is illustrated in Example 2.1.

Example 2.1 Simple data model
Let us consider data that consists of a straight line with inherent noise,

$$y_t = A + Bt + \boldsymbol{\epsilon}_t \qquad t = 0, 1, \ldots, T-1.$$

We assume that the noise is white Gaussian noise (WGN) and that each sample of $\boldsymbol{\epsilon}_t$ has PDF $\mathcal{N}(0, \sigma^2)$ and is uncorrelated with all other samples. The parameters to be determined are A and B, written as $\boldsymbol{\theta} = [A \ B]^{\mathsf{T}}$. If we let

$\mathbf{y} = [y_0\ y_1 \dots y_{T-1}]^\mathsf{T}$, then the PDF is

$$p(\mathbf{y};\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{\frac{T}{2}}} \exp\left[-\frac{1}{2\sigma^2}\sum_{t=0}^{T-1}(y_t - A - Bt)^2\right]. \tag{2.1}$$

We justify the assumption of WGN because the Gaussian distribution is mathematically tractable so that closed form estimators can be found. The Gaussian distribution is not only a convenient mathematical model, it is a good model of many natural processes. Also it is a reasonable assumption unless strong evidence suggests otherwise, such as highly correlated noise [14].

◇

Estimations based on PDFs such as (2.1) are termed *classical* estimations, because the parameters of interest are assumed to be *deterministic* but unknown.

If we incorporate prior knowledge into the estimation the approach is termed *Bayesian* estimation. In Example 2.1 we could perhaps, based on prior knowledge, constrain the estimator to produce values of A on a specified interval. The estimator would no longer be considered classical, but rather Bayesian.

If we have an estimator we need some way to asses its quality. We may need to consider whether other estimators that may be better in some sense are available. Perhaps, there are estimators available that have less variance or have a lower bias. An unbiased estimator is however not always better than a biased one. The mean square error (MSE) can be written as

$$\text{MSE} = \text{bias}^2 + \sigma^2. \tag{2.2}$$

In some cases, a biased estimator with low variance can outperform an unbiased estimator with a high variance. An unbiased estimator means that on average the estimator will yield the true value of the unknown parameter.

2.2 Minimum Variance Unbiased Estimation

We will now look at estimators that are unbiased and yield the least variability. These estimators are called minimum variance unbiased (MVU) estimators. An estimator is termed *unbiased* if the expected value of $\hat{\theta}$ equals θ,

$$\text{E}[\hat{\theta}] = \theta, \quad a < \theta < b. \tag{2.3}$$

The MSE is defined as

$$\mathrm{MSE}(\hat{\theta}) = \mathrm{E}[(\hat{\theta} - \theta)^2]. \tag{2.4}$$

The MSE measures the average deviation of the estimator from the true value. Unfortunately, estimators based on the MSE are unrealizable, they cannot be written as functions of the data [14]. Finding minimum variance estimators is not an easy task, even if they exist there is no automatic procedure that will always produce a minimum variance estimator. There are however several approaches to finding minimum variance estimators, such as the following:

1. Determine the Cramer-Rao lower bound (CRLB) and check if some estimators satisfy it.
2. Apply the Rao-Blackwell-Lehmann-Scheffe (RBLS) theorem.
3. Find a minimum variance estimator with the restriction that the estimator be linear.

2.3 Cramer-Rao Lower Bound

The CRLB is very important in parameter estimation, it allows us to place a lower bound on the variance of an estimator. This may allow us to determine if an estimator is the minimum variance unbiased (MVU) estimator. At the least it will give us a value which we can compare against any other unbiased estimator.

If we assume that the PDF $p(\mathbf{y};\theta)$ satisfies the "regularity condition", i.e.,

$$\mathrm{E}\left[\frac{\partial \ln p(\mathbf{y};\theta)}{\partial \theta}\right] = 0 \qquad \text{for all } \theta,$$

where the expectation is taken with respect to $p(\mathbf{y};\theta)$, then the variance of any unbiased estimator $\hat{\theta}$ must satisfy

$$\mathrm{var}(\hat{\theta}) \geq \frac{1}{-\mathrm{E}\left[\dfrac{\partial^2 \ln p(\mathbf{y};\theta)}{\partial \theta^2}\right]} \tag{2.5}$$

where the derivative is evaluated at the true value of θ and the expectation is taken with respect to $p(\mathbf{y};\theta)$. The denominator in (2.5) is called the *Fisher information*, $I(\theta)$, that $\mathbf{y}$ contains about θ or

$$I(\theta) = -\mathrm{E}\left[\frac{\partial^2 \ln p(\mathbf{y};\theta)}{\partial \theta^2}\right]. \tag{2.6}$$

2.4 General Minimum Variance Unbiased Estimation

To find an MVU estimator we need the concept of sufficient statistics and the Rao-Blackwell-Lehmann-Scheffe theorem.

A statistic is sufficient with respect to a model and its unknown parameter if "no other statistic which can be calculated from the same sample provides any additional information as to the value of the parameter" [15]. If a sufficient statistic is known we no longer need the individual data values since all information about the data is represented by the statistic. A statistic is called complete if *there is only one function of the statistic that is unbiased* [14].

To find a sufficient statistic, the Neyman-Fisher factorization can be used, that is if we can factor the PDF $p(\mathbf{y};\theta)$ as

$$p(\mathbf{y};\theta) = g(T(\mathbf{y}),\theta)h(\mathbf{y}), \tag{2.7}$$

where g is a function depending on $\mathbf{y}$ only through $T(\mathbf{y})$ and h is a function depending only on $\mathbf{y}$.

A procedure for finding the MVU is as follows:

1. Find a single sufficient statistic for θ, that is, $T(\mathbf{y})$, by using the Neyman-Fisher factorization theorem.
2. Determine if the sufficient statistic is complete and, if so, proceed; if not the approach cannot be used.
3. Find a function g of the sufficient statistic that yields an unbiased estimator $\hat{\theta} = g(T(\mathbf{y}))$. The MVU estimator is then $\hat{\theta}$.

An alternative third step is:

3.' Evaluate $\hat{\theta} = \mathrm{E}[\check{\theta}|T(\mathbf{y})]$, where $\check{\theta}$ is any unbiased estimator.

2.5 Best Linear Unbiased Estimation

It is often the case that an MVU estimator can not be found, even though it exits. We may not know the PDF of the data or are unwilling to assume a

model for it. In cases where the MVU can not be found we have to look at other sub optimal estimators. A common approach is to restrict the estimator to be linear in the data and find the linear estimator that is unbiased and has minimum variance [14]. This estimator is termed *best linear unbiased estimator* (BLUE) and can be found with only knowledge of the first and second moments of the PDF. Since complete knowledge of the PDF is not necessary, the BLUE is suitable for practical implementations.

The BLUE for the data set $\{y_0, y_1, \ldots, y_{T-1}\}$, who's PDF depends on an unknown parameter θ, is

$$\hat{\theta} = \sum_{t=0}^{T-1} a_t y_t, \tag{2.8}$$

where a_t are the constants to be determined. Now θ is constrained to be unbiased, that is

$$\mathrm{E}[\hat{\theta}] = \sum_{t=0}^{T-1} a_t \mathrm{E}[y_t] = \theta. \tag{2.9}$$

The variance of $\hat{\theta}$ is

$$\mathrm{var}(\hat{\theta}) = \mathbf{a}^{\mathsf{T}} \boldsymbol{\Sigma} \mathbf{a}, \tag{2.10}$$

where $\mathbf{a} = [a_0 \; a_1 \ldots a_{T-1}]^{\mathsf{T}}$ and $\boldsymbol{\Sigma}$ is the covariance matrix. $\mathrm{E}[y_t]$ must be linear in θ, that is

$$\mathrm{E}[y_t] = s_t \theta, \tag{2.11}$$

where s_t are known. To find the BLUE we need to minimize the variance ($\mathrm{var}(\hat{\theta}) = \mathbf{a}^{\mathsf{T}} \boldsymbol{\Sigma} \mathbf{a}$) subject to the unbiased constraint, which from (2.9) and (2.11) becomes

$$\begin{aligned} \sum_{t=0}^{T-1} a_t \mathrm{E}[y_t] &= \theta, \\ \sum_{t=0}^{T-1} a_t s_t \theta &= \theta, \\ \sum_{t=0}^{T-1} a_t s_t &= 1, \end{aligned}$$

or

$$\mathbf{a}^{\mathsf{T}} \mathbf{s} = 1,$$

where $\mathbf{s} = [s_0 \; s_1 \ldots s_{T-1}]^{\mathsf{T}}$. The solution to the minimization problem is

$$\mathbf{a}_{\mathrm{opt}} = \frac{\boldsymbol{\Sigma}^{-1} \mathbf{s}}{\mathbf{s}^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \mathbf{s}},$$

so the BLUE is

$$\hat{\theta} = \frac{\mathbf{s}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{y}}{\mathbf{s}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{s}} \tag{2.12}$$

and has minimum variance

$$\text{var}(\hat{\theta}) = \frac{1}{\mathbf{s}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{s}}. \tag{2.13}$$

As mentioned earlier we only need knowledge of the first and second moments of the PDF, that is

1. $\mathbf{s}$ the scaled mean and
2. $\boldsymbol{\Sigma}$ the covariance matrix.

2.6 Maximum Likelihood Estimation

The maximum likelihood estimator (MLE) is a popular approach to obtaining practical estimators. For most cases of interest it is optimal for large data sets [14]. The MLE estimator is said to be *asymptotically optimal*. This means that, for a data set of size M, when $M \to \infty$

$$\text{E}[\hat{\theta}] \to \theta, \tag{2.14}$$

$$\text{var}(\hat{\theta}) \to \text{CRLB}. \tag{2.15}$$

A estimator $\hat{\theta}$ that satisfied (2.15) is said to *asymptotically efficient* and one that satisfies (2.14) is *asymptotically unbiased*. Optimality cannot be determined for finite data sets, but finding better estimators can be difficult [14].

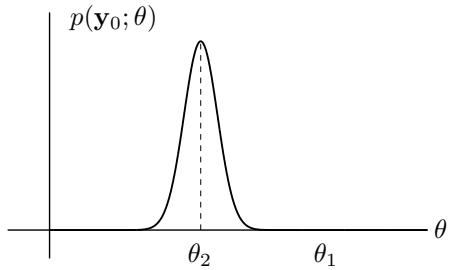

Figure 2.1: Rationale for maximum likelihood estimator.

The MLE for a scalar parameter is defined to be *the value of* θ *that maximizes* $p(\mathbf{y};\theta)$ *for* $\mathbf{y}$ *fixed* [14]. The rationale for the MLE is that $p(\mathbf{y};\theta)dy$ gives the

probability of observing $\mathbf{y}$ in a small volume for a given θ. In Figure 2.6, the PDF is evaluated at $\mathbf{y} = \mathbf{y}_0$ and plotted versus θ. The probability of observing $\mathbf{y} = \mathbf{y}_0$ in the region of $\hat{\theta} = \theta_2$ is more "likely" than for other values of θ.

To find the MLE we must find the most "likely" value of the estimator. In Figure 2.6 this corresponds to finding the peak value of the PDF. This is done by differentiating the PDF (likelihood function) and setting it equal to zero. With complex exponentials this may be tedious. The same result may be achieved by differentiating the log of the likelihood function and setting it equal to zero. To make sure that the solution is in fact the maximum and not a minimum, the second derivative must be found to be positive where the first derivative is equal zero. The following example illustrates the procedure.

Example 2.2 DC Level in white noise
Let us consider the observed data set

$$y_t = A + \boldsymbol{\epsilon}_t \qquad n = 0, 1, \ldots, T-1,$$

where A is unknown and $\boldsymbol{\epsilon}_t$ is WGN with variance σ^2. The PDF is

$$p(\mathbf{y}; A) = \frac{1}{(2\pi\sigma^2)^{\frac{T}{2}}} \exp\left[-\frac{1}{2\sigma^2}\sum_{t=0}^{T-1}(y_t - A)^2\right], \tag{2.16}$$

and the log-likelihood function is

$$\log p(\mathbf{y}; A) = \log \frac{1}{(2\pi\sigma^2)^{\frac{T}{2}}} - \frac{1}{2\sigma^2}\sum_{t=0}^{T-1}(y_t - A)^2.$$

Taking the derivative of the log-likelihood function with respect to A produces

$$\frac{\partial \log p(\mathbf{y}; A)}{\partial A} = \frac{1}{\sigma^2}\sum_{t=0}^{T-1}(y_t - A),$$

which set to zero, yields the MLE

$$\hat{A} = \frac{1}{T}\sum_{t=0}^{T-1} y_t.$$

We can assure ourselves that $\hat{A}$ gives the maximum value of (2.16) by noting $\hat{A}$ results in a zero value of the exponent which is otherwise less than one.

$\diamond$

One advantage of the MLE is that we are always able to find it numerically for a given data set. The reason being that MLE is always found by maximizing the likelihood function. The safest way to do this is to do a grid search. However, this assumes that we know a specified range of possible values, which is not always the case. When a grid search is not viable we resort to iterative methods such as the Newton-Raphson method, the scoring approach or the EM algorithm [14]. In general, these methods produce the MLE if the initial guess is close enough. If the initial guess is not good enough, the methods may not converge or may converge to a local maximum. Unlike general maximization problems, the function to be maximized is not know *a priori*. The likelihood function changes with each data set, so we are in fact maximizing a random function.

Chapter 3

The Expectation-Maximization Algorithm

The EM algorithm [8] is often an efficient way to compute ML estimates in the presence of missing or hidden (latent) data [16]. The EM algorithm is not a specific algorithm, but rather a general approach to a maximum likelihood estimator [17]. The EM algorithm has been used widely in a number of fields and has been proven to be extremely useful in many areas of signal processing [9].

The EM algorithm is an iterative process and consists of two steps, the expectation step and the maximization step (E- and M-step). In the E-step, the expectation is done with respect to the latent variables, using the current estimate of the parameters and conditioned upon the observations. In the M-step, a new estimate of the parameters is found. These two steps are iterated until convergence.

Typically, we would like to estimate a parameter vector θ by maximizing a likelihood function. However, often the data to compute the likelihood function is not available. The EM algorithm does not maximize the log-likelihood function directly, but rather the so called EM-functional, i.e., the expected value of the

complete log-likelihood function.

3.1 Derivation Of The EM Algorithm[1]

Assume $\mathbf{y}$ is a random vector and we wish to find θ such that $p(\mathbf{y}|\theta)$ is a maximum. The log-likelihood function of $p(\mathbf{y}|\theta)$ is

$$L(\theta) = \log p(\mathbf{y}|\theta). \tag{3.1}$$

Since $\log(x)$ is a strictly increasing function, the value of θ that maximizes $L(\theta)$ will also maximize $p(\mathbf{y}|\theta)$.

The EM algorithm is an iterative procedure. Assume that after the kth iteration of the algorithm the current parameter estimation is θ_k. Because the goal is to maximize $L(\theta)$, the new estimate of $L(\theta)$ must be larger value than the previous one,

$$L(\theta) > L(\theta_k).$$

This means that the intermediate result of $L(\theta_k)$ at iteration k will always be less than the maximum value of $L(\theta)$. Likewise, we can maximize the difference

$$L(\theta) - L(\theta_k) = \log p(\mathbf{y}|\theta) - \log p(\mathbf{y}|\theta_k). \tag{3.2}$$

Let $\mathbf{u}$ now denote a latent or hidden random vector. Using the law of total probability, $p(\mathbf{y}|\theta)$, may be written in terms of $\mathbf{u}$ as

$$p(\mathbf{y}|\theta) = \sum_{\mathbf{u}} p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta).$$

Equation (3.2) can now be rewritten as

$$L(\theta) - L(\theta_k) = \log\left(\sum_{\mathbf{u}} p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta)\right) - \log p(\mathbf{y}|\theta_k). \tag{3.3}$$

Noting that $p(\mathbf{u}|\mathbf{y},\theta) \geq 0$ and $\sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k) = 1$ and using Jensens inequality [18]

$$\log \sum_{i=1}^{n} \lambda_i x_i \geq \sum_{i=1}^{n} \lambda_i \log x_i$$

[1] Based on derivation in [16].

for constants $\lambda_i \geq 1$ and $\sum_{i=1}^{n} \lambda_i = 1$, then (3.2) becomes

$$\begin{aligned}
L(\theta) - L(\theta_k) &= \log\left(\sum_{\mathbf{u}} p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta)\right) - \log p(\mathbf{y}|\theta_k) \\
&= \log\left(\sum_{\mathbf{u}} p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta_m)\frac{p(\mathbf{u}|\mathbf{y},\theta_k)}{p(\mathbf{u}|\mathbf{y},\theta_k)}\right) - \log p(\mathbf{y}|\theta_k) \\
&= \log\left(\sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k)\frac{p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta)}{p(\mathbf{u}|\mathbf{y},\theta_k)}\right) - \log p(\mathbf{y}|\theta_k) \\
&\geq \sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k)\log\left(\frac{p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta)}{p(\mathbf{u}|\mathbf{y},\theta_k)}\right) - \log p(\mathbf{y}|\theta_k) \\
&= \sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k)\log\left(\frac{p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta)}{p(\mathbf{u}|\mathbf{y},\theta_k)p(\mathbf{y}|\theta_k)}\right) \\
&\stackrel{\triangle}{=} \Delta(\theta|\theta_k). \qquad (3.4)
\end{aligned}$$

We now define the EM functional as

$$\mathrm{EMf}(\theta|\theta_k) \stackrel{\triangle}{=} L(\theta_k) + \Delta(\theta_k|\theta_k)$$

and write

$$L(\theta) \geq L(\theta_k) + \Delta(\theta_k|\theta_k) = \mathrm{EMf}(\theta_k|\theta_k).$$

The function $\mathrm{EMf}(\theta|\theta_k)$ is bounded above by the likelihood function $L(\theta)$ and if $\theta = \theta_k$, then $\mathrm{EMf}(\theta_k|\theta_k) = L(\theta_k)$. So any θ that increases $\mathrm{EMf}(\theta|\theta_k)$ will also increase $L(\theta)$. The EM algorithm selects θ such that $\mathrm{EMf}(\theta|\theta_k)$ is maximized in each iteration.

$$\begin{aligned}
\theta_{n+1} &= \arg\max_{\theta}\{\mathrm{EMf}(\theta|\theta_k)\} \\
&= \arg\max_{\theta}\left\{L(\theta) + \sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k)\log\left(\frac{p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta)}{p(\mathbf{u}|\mathbf{y},\theta_k)p(\mathbf{y}|\theta_k)}\right)\right\} \\
&= \arg\max_{\theta}\left\{\sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k)\log p(\mathbf{y}|\mathbf{u},\theta)p(\mathbf{u}|\theta)\right\} \\
&= \arg\max_{\theta}\left\{\sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k)\log\left(\frac{p(\mathbf{y}|\mathbf{u},\theta)}{p(\mathbf{u},\theta)}\frac{p(\mathbf{u}|\theta)}{P(\theta)}\right)\right\} \\
&= \arg\max_{\theta}\left\{\sum_{\mathbf{u}} p(\mathbf{u}|\mathbf{y},\theta_k)\log p(\mathbf{y},\mathbf{u}|\theta)\right\} \\
&= \arg\max_{\theta}\mathrm{E}[\log p(\mathbf{y},\mathbf{u}|\theta)|\mathbf{y},\theta_k]. \qquad (3.5)
\end{aligned}$$

In (3.5) the expectation and maximization steps are apparent. The EM algorithm consists of iterating two steps.

1. E-step. Calculate the EM functional, $\mathrm{E}[\log p(\mathbf{y}, \mathbf{u}|\theta)|\mathbf{y}, \theta_k]$, using the current estimate of the parameters.

2. M-step. Maximize the EM functional with respect to θ.

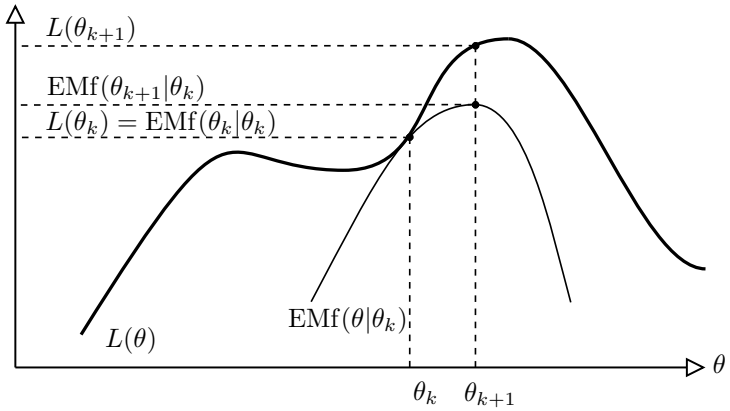

Figure 3.1: Graphical interpretation of a single iteration of the EM algorithm. $\mathrm{EMf}(\theta|\theta_k)$ is bounded from above by the likelihood function $L(\theta)$. In each iteration the value of θ is chosen to maximize $\mathrm{EMf}(\theta|\theta_k)$.

3.2 Convergence of the EM Algorithm

At each iteration of the algorithm, the value parameter is computed so that the likelihood function does not decrease. Thus θ_{k+1} is the estimate that maximizes the difference $\Delta(\theta|\theta_k)$. For the current estimate, $\Delta(\theta_k|\theta_k) = 0$. Since θ_{k+1} is chosen to maximize $\Delta(\theta|\theta_k)$ we can state the

$$\Delta(\theta_{k+1}|\theta_k) \geq \Delta(\theta_k|\theta_k) = 0,$$

so for each iteration, the likelihood function $L(\theta)$ is nondecreasing.

Despite the fact that the EM algorithm converges, there is no guarantee that it will converge to a global maximum. If the likelihood function has multiple maxima, then the local maxima that the algorithm converges to will depend on the initial value of θ_0. Further discussions about the convergence of the EM algorithm can be found in [19].

Chapter 4

Principal Component Analysis

Principal Component Analysis (PCA) [1], [2] is one of best known methods for multivariate analysis. The central idea of PCA is to reduce the dimensionality of a data set with interrelated variables, while retaining as much as possible of the variations in the data set. The dimensionality reduction is achieved by linearly transforming the data into a new set of variables, the principal components (PCs), which are uncorrelated. The PCs are orthogonal to each other and are ordered such that the first component has greatest variance, the second component has second greatest variance and so on.

PCA is derived here in two ways, first by maximizing the variance of projection along each component, and secondly using singular value decomposition. There are other ways to derive PCA, such as using the minimum reconstruction error, i.e., the squared distance between the original data and its estimate [20].

4.1 PCA using Maximum Variance

Let $\mathbf{x}$ be a vector of M random variables. The covariances or correlations between the M variables are of interest. The number of variables may be large and looking at the variances of all M variables or the $\frac{1}{2}M(M-1)$ correlations

or covariances may not give any insight into the data. What is done instead is to look at a few derived variables that preserve most of the information given by the variances and correlations or covariances.

PCA concentrates on variances but it does not completely ignore covariances and correlations. The first step in PCA is to look for a linear function $\mathbf{a}_1^\mathsf{T}\mathbf{x}$ of the elements of $\mathbf{x}$ having maximum variance, where $\mathbf{a}_1$ is a vector of M constants $a_{11}, a_{12}, \ldots, a_{1M}$. So the first PC is given by

$$\mathbf{a}_1^\mathsf{T}\mathbf{x} = \sum_{j=1}^{M} \alpha_{1j} x_j.$$

Next, a second linear function $\mathbf{a}_2^\mathsf{T}\mathbf{x}$, uncorrelated to $\mathbf{a}_1^\mathsf{T}\mathbf{x}$ having maximum variance is found and so on. So the kth linear function $\mathbf{a}_k^\mathsf{T}\mathbf{x}$ has maximum variance and is uncorrelated to all functions $\mathbf{a}_j^\mathsf{T}\mathbf{x}$ where $j < k$. The kth derived variable, $\mathbf{a}_k^\mathsf{T}\mathbf{x}$ is the kth PC.

Now that the PCs have been defined we need to find them. Let us consider the case where the covariance matrix ($\mathbf{\Sigma}$) of the random variables $\mathbf{x}$ is known. The (i, j)th element of $\mathbf{\Sigma}$ is the covariance between the ith and jth elements of $\mathbf{x}$ when $i \neq j$, and the variance of the jth element when $i = j$.

In the linear function $\mathbf{a}_1^\mathsf{T}\mathbf{x}$, $\mathbf{a}_1$ maximizes the variance, $\text{var}[\mathbf{a}_1^\mathsf{T}\mathbf{x}] = \mathbf{a}_1^\mathsf{T}\mathbf{\Sigma}\mathbf{a}_1$. The normalization constraint, $\mathbf{a}_1^\mathsf{T}\mathbf{a}_1 = 1$, is now imposed on $\mathbf{a}_1$. Other constraints may be used but they will result in more difficult optimization problems and produce derived variables different from PCs [2].

Lagrange multipliers are used to maximize $\mathbf{a}_1\mathbf{\Sigma}\mathbf{a}$ subject to $\mathbf{a}_1^\mathsf{T}\mathbf{a}_1 = 1$, i.e.

$$\mathbf{a}_1^\mathsf{T}\mathbf{\Sigma}\mathbf{a}_1 - \lambda(\mathbf{a}_1^\mathsf{T}\mathbf{a} - 1),$$

where λ is the Lagrange multiplier. Differentiation with respect to $\mathbf{a}_1$ gives

$$\mathbf{\Sigma}\mathbf{a}_1 - \lambda\mathbf{a}_1 = 0,$$

which can be rewritten as

$$(\mathbf{\Sigma} - \lambda\mathbf{I}_m)\mathbf{a}_1 = 0,$$

where $\mathbf{I}_M$ is the $M \times M$ identity matrix. We see that λ is a eigenvalue of $\mathbf{\Sigma}$ and $\mathbf{a}_1$ is the corresponding eigenvector. We also see from the above equations that the following needs to be maximized

$$\mathbf{a}_1^\mathsf{T}\mathbf{\Sigma}\mathbf{a}_1 = \mathbf{a}_1^\mathsf{T}\lambda\mathbf{a}_1 = \lambda\mathbf{a}_1^\mathsf{T}\mathbf{a}_1 = \lambda.$$

Therefore λ must be as large as possible. This means that $\mathbf{a}_1$ is the eigenvector to the largest eigenvalue of $\mathbf{\Sigma}$, and $\text{var}(\mathbf{a}_1^\mathsf{T}\mathbf{x}) = \lambda_1$, the largest eigenvalue.

The kth PC of $\mathbf{x}$ is $\mathbf{a}_k^\mathsf{T}\mathbf{x}$ and $\text{var}(\mathbf{a}_k^\mathsf{T}\mathbf{x}) = \lambda_k$, where λ_k is the kth largest eigenvalue of $\mathbf{\Sigma}$ and $\mathbf{a}_k$ is the corresponding eigenvector.

4.2 PCA using SVD[1]

Another method for finding PCs is to make use of singular value decomposition (SVD). Very efficient algorithms are available for computing SVD and it is the most widely used method for finding PCs.

Now, let $\mathbf{X}$ be an $M \times T$ matrix and $\mathbf{X}^\mathsf{T}\mathbf{X}$ be of rank r, square, symmetric and of size $M \times M$. Let us now define the following:

1. $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ is a set of orthonormal $M \times 1$ eigenvectors with associated eigenvalues $\{\lambda_1, \lambda_2, \ldots, \lambda_r\}$ for $\mathbf{X}^\mathsf{T}\mathbf{X}$. Then
$$(\mathbf{X}^\mathsf{T}\mathbf{X})\mathbf{v}_i = \lambda_i \mathbf{v}_i.$$
2. $\sigma_i = \sqrt{\lambda_i}$ are positive real and termed *singular values*.
3. $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r\}$ is the set of $T \times 1$ vectors defined by $\mathbf{u}_i = \frac{1}{\sigma_i}\mathbf{X}\mathbf{v}_i$.
4. $\mathbf{u}_j\mathbf{u}_i = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$
5. $||\mathbf{X}\mathbf{v}_i|| = \sigma_i$

The scalar version of SVD is in fact definition 3. above rewritten as

$$\mathbf{X}\mathbf{v}_i = \sigma_i \mathbf{u}_i. \tag{4.1}$$

Note that (4.1) tells us that $\mathbf{X}$ multiplied by a eigenvector of $\mathbf{X}^\mathsf{T}\mathbf{X}$ is equal to a scalar times another vector.

Now let us define the following matrix

$$\mathbf{D} \equiv \begin{bmatrix} \sigma_1 & & & & \\ & \ddots & & \mathbf{0} & \\ & & \sigma_r & & \\ & \mathbf{0} & & 0 & \\ & & & & \ddots \\ & & & & & 0 \end{bmatrix}$$

[1] Based on [21].

where $\sigma_1 \geq \sigma_2 \ldots \geq \sigma_r$ are the rank-ordered singular values. Now we define two orthogonal matrices.

$$\mathbf{V} = [\mathbf{v}_1\ \mathbf{v}_2 \ldots \mathbf{v}_M],$$
$$\mathbf{U} = [\mathbf{u}_1\ \mathbf{u}_2 \ldots \mathbf{u}_T].$$

We have appended zeros to fill up the matrices. We can now write the SVD as

$$\mathbf{XV} = \mathbf{UD}.$$

Because $\mathbf{V}$ is orthogonal we can multiply both sides from the right by $\mathbf{V}^{-1} = \mathbf{V}^\mathsf{T}$ and arrive at the final form of the decomposition

$$\mathbf{XVV}^T = \mathbf{UDV}^\mathsf{T} \Rightarrow \mathbf{X} = \mathbf{UDV}^\mathsf{T}. \tag{4.2}$$

We see that any matrix can be converted into an orthogonal matrix, diagonal matrix and another orthogonal matrix.

Now we define a new matrix $\mathbf{Y}$ of size $T \times M$ by

$$\mathbf{Y} = \frac{1}{\sqrt{T-1}}\mathbf{X}^\mathsf{T}.$$

Each column of $\mathbf{Y}$ has zero mean. We note that

$$\begin{aligned}\mathbf{Y}^\mathsf{T}\mathbf{Y} &= \left(\frac{1}{\sqrt{T-1}}\mathbf{X}^\mathsf{T}\right)^\mathsf{T}\left(\frac{1}{\sqrt{T-1}}\mathbf{X}^\mathsf{T}\right) \\ &= \frac{1}{T-1}\mathbf{X}\mathbf{X}^\mathsf{T} \\ &= \mathbf{\Sigma}_\mathbf{X}.\end{aligned}$$

The PCs of $\mathbf{X}$ are the eigenvectors of $\mathbf{\Sigma}_\mathbf{X}$. By calculating the SVD of $\mathbf{Y}$ we find the principal components of $\mathbf{X}$ in the columns of $\mathbf{V}$.

Chapter 5

Noisy PCA

The formulation of the PCA in the previous chapter was based on linearly projecting data onto a subspace of lower dimension than the original data space. PCA can also be derived with the use of maximum likelihood (ML) estimation of parameters in a latent variable model [5], [6]. This formulation of PCA has several advantages to conventional PCA, including the following:

- An EM algorithm can be derived for PCA that is computationally efficient.
- By combining the ML model and EM gives us the possibility to deal with missing values in the data.
- The likelihood functions allows a direct comparison to other density models.

Let us now introduce a latent (hidden) variable $\mathbf{u}_n$ with a prior Gaussian distribution $p(\mathbf{u}_n)$ and the Gaussian distribution $p(\mathbf{y}_n; \theta)$ over the observed variable $\mathbf{y}_n$.

Let the observed variable $\mathbf{y}_n$ be $T \times 1$, $\mathbf{u}_n$ be $r \times 1$ and

$$\mathbf{y}_n = \boldsymbol{\mu} + \mathbf{B}\mathbf{u}_n + \boldsymbol{\epsilon}_n, \qquad n = 0, \ldots, M-1. \tag{5.1}$$

The $T \times r$ matrix $\mathbf{B}$ relates the two variables $\mathbf{u}_n$ and $\mathbf{y}_n$. Here $\boldsymbol{\epsilon}_n \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_T)$, $\mathbf{u}_n \sim \mathcal{N}(0, \mathbf{I}_r)$, $\mathbf{y}_n \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\boldsymbol{\mu}$ represents the mean. Also, $\boldsymbol{\epsilon}_n$ and $\mathbf{u}_n$ are independent. This model is closely related to the factor analysis model [22], [23].

The covariance matrix $\boldsymbol{\Sigma}$ $(T \times T)$ is

$$\begin{aligned}
\boldsymbol{\Sigma} = \mathrm{E}[\mathbf{y}\mathbf{y}^\mathsf{T}] =& \mathrm{E}[(\mathbf{B}\mathbf{u} + \boldsymbol{\epsilon})(\mathbf{B}\mathbf{u} + \boldsymbol{\epsilon})^\mathsf{T}] \\
=& \mathrm{E}[\mathbf{B}\mathbf{u}(\mathbf{B}\mathbf{u})^\mathsf{T} + \mathbf{B}\mathbf{u}\boldsymbol{\epsilon}^\mathsf{T} + \boldsymbol{\epsilon}(\mathbf{B}\mathbf{u})^\mathsf{T} + \boldsymbol{\epsilon}\boldsymbol{\epsilon}^\mathsf{T}] \\
=& \mathbf{B}\mathrm{E}[\mathbf{u}\mathbf{u}^\mathsf{T}]\mathbf{B}^\mathsf{T} + \underbrace{\mathbf{B}\mathrm{E}[\mathbf{u}\boldsymbol{\epsilon}^\mathsf{T}] + \mathrm{E}[\boldsymbol{\epsilon}\mathbf{u}^\mathsf{T}]\mathbf{B}^\mathsf{T}}_{=0 \text{ } (\mathbf{u} \text{ and } \boldsymbol{\epsilon} \text{ are independent})} + \mathrm{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\mathsf{T}] \\
=& \mathbf{B}\mathrm{E}[\mathbf{u}\mathbf{u}^\mathsf{T}]\mathbf{B}^\mathsf{T} + \mathrm{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\mathsf{T}] \\
=& \mathbf{B}\mathbf{B}^\mathsf{T} + \sigma^2 \mathbf{I}_T.
\end{aligned}$$

By maximizing the log-likelihood of $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_M\}$ we determine the maximum likelihood estimates (MLE) of the parameters $\boldsymbol{\theta} = (\mathbf{B}, \sigma^2)$. The log-likelihood function is

$$L(\boldsymbol{\theta}) = \sum_{n=0}^{M-1} \log\big(p(\mathbf{y}_n)\big). \tag{5.2}$$

The closed form result for maximizing (5.2) is [6]

$$\mathbf{B} = \mathbf{W}(\boldsymbol{\Lambda} - \sigma^2 \mathbf{I}_r)^{1/2}\mathbf{R},$$

and

$$\sigma^2 = \frac{1}{T - r} \sum_{j=r+1}^{T} \lambda_j.$$

The columns of $\mathbf{W}$ are the eigenvectors of the covariance matrix of $\{\mathbf{y}_n\}$. The diagonal $\boldsymbol{\Lambda}$ hold the corresponding eigenvalues (λ_j) and $\mathbf{R}$ is an arbitrary rotation matrix. Here $\mathbf{R}$ is chosen as the identity matrix $(\mathbf{R} = \mathbf{I})$.

The complete likelihood function of the observed variable $\mathbf{y}_n$ in model (5.1) is given by

$$\begin{aligned}
p(\mathbf{y}_n, \mathbf{u}_n) =& p(\mathbf{y}_n | \mathbf{u}_n) p(\mathbf{u}_n) \\
=& (2\pi\sigma^2)^{\frac{-T}{2}} \exp\left(\frac{-\|((\mathbf{y}_n - \boldsymbol{\mu}) - \mathbf{B}\mathbf{u}_n\|^2}{2\sigma^2}\right) \\
& \times (2\pi)^{\frac{-r}{2}} \exp\left(\frac{-\|\mathbf{u}_n\|^2}{2}\right).
\end{aligned} \tag{5.3}$$

Taking the logarithm of both sides of (5.5) and omitting non relevant terms gives

$$\begin{aligned}\log(p(\mathbf{y}_n, \mathbf{u}_n)) =& \frac{-T}{2}\log\sigma^2 - \frac{1}{2\sigma^2}(\mathbf{y}_n - \boldsymbol{\mu})^\mathsf{T}(\mathbf{y}_n - \boldsymbol{\mu}) \\ &+ \frac{1}{\sigma^2}\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}(\mathbf{y}_n - \boldsymbol{\mu}) - \frac{1}{2\sigma^2}\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{B}\mathbf{u}_n. \end{aligned} \tag{5.4}$$

Using trace algebra we can write

$$\begin{aligned}\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{B}\mathbf{u}_n &= \mathrm{tr}(\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{B}\mathbf{u}_n) \\ &= \mathrm{tr}(\mathbf{u}_n\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{B}),\end{aligned}$$

and rewrite (5.4) as

$$\begin{aligned}\log(p(\mathbf{y}_n, \mathbf{u}_n)) =& \frac{-T}{2}\log\sigma^2 - \frac{1}{2\sigma^2}(\mathbf{y}_n - \boldsymbol{\mu})^\mathsf{T}(\mathbf{y}_n - \boldsymbol{\mu}) \\ &+ \frac{1}{\sigma^2}\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}(\mathbf{y}_n - \boldsymbol{\mu}) - \frac{1}{2\sigma^2}\mathrm{tr}(\mathbf{u}_n\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{B}). \end{aligned} \tag{5.5}$$

The logarithm of the complete likelihood function for the model in (5.1), for $n = 0, \ldots, M-1$ is given by

$$\begin{aligned}L(\boldsymbol{\theta}) &= \sum_{n=0}^{M-1} \log\big(p(\mathbf{y}_n, \mathbf{u}_n)\big) \\ &= \sum_{n=0}^{M-1} \left\{ \frac{-T}{2}\log\sigma^2 - \frac{1}{2\sigma^2}(\mathbf{y}_n - \boldsymbol{\mu})^\mathsf{T}(\mathbf{y}_n - \boldsymbol{\mu}) + \frac{1}{\sigma^2}\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}(\mathbf{y}_n - \boldsymbol{\mu}) \right. \\ &\qquad \left. - \frac{1}{2\sigma^2}\mathrm{tr}(\mathbf{u}_n\mathbf{u}_n^\mathsf{T}\mathbf{B}^\mathsf{T}\mathbf{B}) \right\}. \end{aligned} \tag{5.6}$$

Now the expectation of (5.6), conditioned on the data ($\mathbf{Y}$) and the current estimate of the parameters ($\boldsymbol{\theta}_k$), is taken to obtain the EM functional which we will now denote EMf,

$$\mathrm{EMf} = \mathrm{E}[L(\boldsymbol{\theta})|\mathbf{Y}, \boldsymbol{\theta}_k]. \tag{5.7}$$

The following notations will also be used,

$$\mathbf{z}_n = \mathrm{E}[\mathbf{u}_n|\mathbf{Y}, \boldsymbol{\theta}_k], \tag{5.8}$$

$$\mathbf{W}_n = \mathrm{E}[\mathbf{u}_n\mathbf{u}_n^\mathsf{T}|\mathbf{Y}, \boldsymbol{\theta}_k]. \tag{5.9}$$

The EMf can be rewritten in the following manner

$$\begin{aligned}\text{EMf} = -\sum_{n=0}^{M-1}\Big\{ &\frac{T}{2}\log\sigma^2 + \frac{1}{2\sigma^2}(\mathbf{y}_n-\boldsymbol{\mu})^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) \\ &- \frac{1}{\sigma^2}\mathbf{z}_n^{\mathsf{T}}\mathbf{B}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) + \frac{1}{2(\sigma^2)^2}\mathrm{tr}(\mathbf{W}_n\mathbf{B}^{\mathsf{T}}\mathbf{B})\Big\}. \end{aligned} \tag{5.10}$$

Taking the expected value of (5.6) is the E-step in the EM algorithm.

Using that the posterior distribution of $\mathbf{u}_n$ given $\mathbf{y}_n$ is given by [24]

$$p(\mathbf{u}_n|\mathbf{y}_n) \sim \mathcal{N}(\mathbf{M}^{-1}\mathbf{B}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}), \sigma^2\mathbf{M}^{-1}), \tag{5.11}$$

and the fact that $\mathbf{W}_n = \mathrm{cov}(\mathbf{u}_n) + \mathbf{z}_n\mathbf{z}_n^{\mathsf{T}}$ we obtain

$$\mathbf{z}_n = \mathbf{M}^{-1}\mathbf{B}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}), \tag{5.12}$$

$$\mathbf{W}_n = \sigma^2\mathbf{M}^{-1} + \mathbf{z}_n\mathbf{z}_n^{\mathsf{T}}, \tag{5.13}$$

where

$$\mathbf{M} = \mathbf{B}^{\mathsf{T}}\mathbf{B} + \sigma^2\mathbf{I}_r. \tag{5.14}$$

Now EMf must be maximized with respect to $\mathbf{B}$ and σ^2, respectively. This is the M-step in the algorithm. To maximize the function, the derivative of EMf with respect to σ^2 is set to zero. This is a maximum because of the quadratic nature of the PDF. The derivative is

$$\begin{aligned}\frac{\partial\text{EMf}}{\partial\sigma^2} &= -\sum_{n=0}^{M-1}\Big\{ \frac{T}{2\sigma^2} - \frac{1}{2(\sigma^2)^2}(\mathbf{y}_n-\boldsymbol{\mu})^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) \\ &\qquad + \frac{1}{(\sigma^2)^2}\mathbf{z}_n^{\mathsf{T}}\mathbf{B}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) - \frac{1}{2(\sigma^2)^2}\mathrm{tr}(\mathbf{W}_n\mathbf{B}^{\mathsf{T}}\mathbf{B})\Big\} \\ &= -\frac{MT}{2\sigma^2} - \frac{1}{2(\sigma^2)^2}\sum_{n=0}^{M-1}\Big\{ -(\mathbf{y}_n-\boldsymbol{\mu})^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) + 2\mathbf{z}_n^{\mathsf{T}}\mathbf{B}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) \\ &\qquad - \frac{1}{2(\sigma^2)^2}\mathrm{tr}(\mathbf{W}_n\mathbf{B}^{\mathsf{T}}\mathbf{B})\Big\}. \end{aligned} \tag{5.15}$$

Now (5.15) is set equal to zero, giving

$$\begin{aligned}\frac{MT}{2\sigma^2} = \frac{1}{2(\sigma^2)^2}\sum_{n=0}^{M-1}\Big\{ &(\mathbf{y}_n-\boldsymbol{\mu})^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) - 2\mathbf{z}_n^{\mathsf{T}}\mathbf{B}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) \\ &- \frac{1}{2(\sigma^2)^2}\mathrm{tr}(\mathbf{W}_n\mathbf{B}^{\mathsf{T}}\mathbf{B})\Big\}. \end{aligned} \tag{5.16}$$

Simplifying (5.16) gives

$$\sigma^2 = \frac{1}{MT}\sum_{n=0}^{M-1}\Big\{(\mathbf{y}_n-\boldsymbol{\mu})^\mathsf{T}(\mathbf{y}_n-\boldsymbol{\mu}) - 2\mathrm{E}[\mathbf{z}_n^\mathsf{T}\mathbf{B}^\mathsf{T}(\mathbf{y}_n-\boldsymbol{\mu}) + \mathrm{tr}(\mathbf{W}_n\mathbf{B}^\mathsf{T}\mathbf{B})\Big\}. \tag{5.17}$$

The differential[1] with respect to $\mathbf{B}$ is now found as

$$\begin{aligned} d\mathrm{EMf} &= -\sum_{n=0}^{M-1}\Big\{-\frac{1}{\sigma^2}\mathrm{tr}(\mathrm{E}[\mathbf{u}_n^\mathsf{T}]^\mathsf{T} d\mathbf{B}(\mathbf{y}_n-\boldsymbol{\mu})) + \frac{1}{2\sigma^2}\mathrm{tr}(\mathbf{W}_n d\mathbf{B}^\mathsf{T}\mathbf{B}) \\ &\quad + \frac{1}{2\sigma^2}\mathrm{tr}(\mathbf{W}_n\mathbf{B}^\mathsf{T} d\mathbf{B})\Big\} \\ &= -\sum_{n=0}^{M-1}\Big\{-\frac{1}{\sigma^2}\mathrm{tr}(\mathrm{E}[\mathbf{u}_n](\mathbf{y}_n-\boldsymbol{\mu})^\mathsf{T} d\mathbf{B}) + \frac{1}{\sigma^2}\mathrm{tr}(\mathbf{W}_n\mathbf{B}^\mathsf{T} d\mathbf{B})\Big\} \\ &= -\sum_{n=0}^{M-1}\Big\{-\frac{1}{\sigma^2}\mathrm{tr}\big((\mathbf{z}_n(\mathbf{y}_n-\boldsymbol{\mu})^\mathsf{T} + \mathbf{W}_n\mathbf{B}^\mathsf{T}) d\mathbf{B}\big)\Big\}. \end{aligned} \tag{5.18}$$

Now the differential, dEMf, is set equal to zero to find $\mathbf{B}$, as

$$\begin{aligned} \mathbf{0} &= -\sum_{n=0}^{M-1}\Big\{-\frac{1}{\sigma^2}\mathrm{tr}\big((\mathbf{z}_n(\mathbf{y}_n-\boldsymbol{\mu})^\mathsf{T} + \mathbf{W}_n\mathbf{B}^\mathsf{T}) d\mathbf{B}\big)\Big\} \\ \Rightarrow \mathbf{B} &= \left[\sum_{n=0}^{M-1}(\mathbf{y}_n-\boldsymbol{\mu})\mathbf{z}_n^\mathsf{T}\right]\left[\sum_{n=0}^{M-1}\mathbf{W}_n\right]^{-1}. \end{aligned} \tag{5.19}$$

Here, $\mathbf{B}$ and σ^2 are found iteratively with an EM algorithm, and σ^2 and $\mathbf{B}$ are first initialized with random values. Then (5.12) and (5.13) are computed and the results used in (5.19) and (5.17). (5.12, 5.13) and (5.17, 5.19) are then alternately computed and the process is repeated until $\mathbf{B}$ and σ^2 have converged. This process is called an EM algorithm, where the E-step involves calculating the expected value of the EM functional, EMf, and the M step maximizes EMf.

The EM algorithm for nPCA is given in Algorithm 1. The algorithm is implemented in Matlab.

[1] See Appendix A for details about differentials.

Algorithm 1: EM algorithm for nPCA.

Initialize $\mathbf{B}_1$ and σ_1^2 randomly.
for $k = 1, ...$ **do**

$$
\begin{aligned}
\mathbf{M}_k &= \mathbf{B}_k^\mathsf{T}\mathbf{B}_k + \sigma_k^2\mathbf{I} \\
\mathbf{F}_k &= \mathbf{M}_k^{-1}\mathbf{B}_k^\mathsf{T}\mathbf{Y} \\
\mathbf{V}_k &= \mathbf{F}_k\mathbf{F}_k^\mathsf{T} \\
\mathbf{B}_{k+1} &= \mathbf{Y}\mathbf{F}_k[M\sigma^2\mathbf{M}^{-1} + \mathbf{V}_0]^{-1} \\
\sigma_{k+1}^2 &= \frac{1}{MT}\Big[\mathrm{tr}(\mathbf{Y}^\mathsf{T}\mathbf{Y}) - 2\mathrm{tr}(\mathbf{B}_{k+1}^\mathsf{T}\mathbf{Y}\mathbf{F}_k^\mathsf{T}) \\
&\quad + M\mathrm{tr}(\sigma_k^2\mathbf{M}_k^{-1}\mathbf{B}_{k+1}^\mathsf{T}\mathbf{B}_{k+1}) \\
&\quad + \mathrm{tr}(\mathbf{B}_{k+1}^\mathsf{T}\mathbf{B}_{k+1}\mathbf{V}_k^\mathsf{T})\Big]
\end{aligned}
$$

Terminate when σ^2 and $\mathbf{B}$ have converged .

Chapter 6

nPCA Using Basis Expansion

In [13], the model given in (5.1) is modified by setting $\mathbf{G} = \mathbf{\Phi B}$, where $\mathbf{\Phi}$ is a $T \times T$ matrix of smooth basis functions. In this thesis we focus on the Fourier basis. The model is now

$$\mathbf{y}_n = \boldsymbol{\mu} + \mathbf{G}\mathbf{u}_n + \boldsymbol{\epsilon}_n, \quad n = 0 \ldots, M-1. \tag{6.1}$$

Each row of the Fourier matrix $\mathbf{\Phi}$ is defined as

$$\mathbf{\Phi}_t^{\mathsf{T}} = \left[\cos\left(\frac{2\pi tk}{T}\right)\Big|_{k=0} \quad \sin\left(\frac{2\pi tk}{T}\right)\Big|_{k=0} \quad \ldots \right.$$
$$\left. \cos\left(\frac{2\pi tk}{T}\right)\Big|_{k=\frac{T-1}{2}} \quad \sin\left(\frac{2\pi tk}{T}\right)\Big|_{k=\frac{T-1}{2}} \right]. \tag{6.2}$$

The distribution of the observed data is $\mathbf{y}_n \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ where $\mathbf{\Sigma} = \mathbf{\Phi B B}^{\mathsf{T}}\mathbf{\Phi}^{\mathsf{T}} + \sigma^2\mathbf{I}_T$. The closed-form solution given in [13] for $\mathbf{B}$ is

$$\hat{\mathbf{B}} = \mathbf{K}(\mathbf{D} - \sigma^2\mathbf{I})^{1/2}, \tag{6.3}$$

where $\mathbf{K}$ is a matrix of eigenvectors of

$$\mathbf{S}_{\mathbf{\Phi}} = \mathbf{\Phi}^{\mathsf{T}}\mathbf{S}_y\mathbf{\Phi}, \tag{6.4}$$

$\mathbf{D}$ is a diagonal matrix that contains the corresponding eigenvalues and

$$\mathbf{S}_y = \frac{1}{M}\sum_{n=0}^{M-1}(y_n - \boldsymbol{\mu})(y_n - \boldsymbol{\mu})^{\mathsf{T}}$$

is the sample covariance. The general ML solution for σ^2 is

$$\hat{\sigma}^2 = \frac{\mathrm{tr}(\mathbf{S}_y) - \mathrm{tr}(\mathbf{B}^{\mathsf{T}}\mathbf{S}_{\boldsymbol{\Phi}}\mathbf{B})}{T - r}. \tag{6.5}$$

and r is the number of PCs. Setting $\mathbf{B} = \hat{\mathbf{B}}$ and $\mathbf{S}_{\boldsymbol{\Phi}} = \hat{\mathbf{B}}\mathbf{D}\hat{\mathbf{B}}^{\mathsf{T}}$, (6.5) can be rewritten as

$$\hat{\sigma}^2 = \frac{\mathrm{tr}(\mathbf{S}_y) - \mathrm{tr}(\mathbf{D})}{T - r}, \tag{6.6}$$

An EM algorithm can also be used to optimize $\mathbf{B}$ and σ^2. This is done by modifying the observed data $\mathbf{y}_n$ by setting $\mathbf{y}'_n = \boldsymbol{\Phi}^{\mathsf{T}}\mathbf{y}_n$ and then solving for $\boldsymbol{\Phi}\mathbf{B}$ and σ^2.

If the $\boldsymbol{\Phi}$ matrix is full (of size $T \times T$) it will result in a full nPCA solution. By removing columns from the $\boldsymbol{\Phi}$ matrix specific frequency components can be removed from the solution and accordingly the computational requirements are reduced.

To determine the number of PCs and the number of basis functions, the AIC [25] and the BIC [26] are used. In this case, the usage is unusual because of the need to determine two parameters. The AIC is given by

$$\mathrm{AIC}(T, r) = -2\mathcal{L}_\theta + 2\mathrm{dim}(\hat{\boldsymbol{\theta}}), \tag{6.7}$$

where $\hat{\boldsymbol{\theta}}$ is the maximum likelihood estimate of the parameter vector $\boldsymbol{\theta}$ and $\mathrm{dim}(\hat{\boldsymbol{\theta}})$ is the number of free parameters given by

$$\mathrm{dim}(\hat{\boldsymbol{\theta}}) = Tr - r(r-1)/2 + r + 1. \tag{6.8}$$

The BIC has $\log(M)$ in place of the 2 in the second term of the AIC and is given by

$$\mathrm{BIC}(T, q) = -2\mathcal{L}_\theta + \log(M)\mathrm{dim}(\hat{\boldsymbol{\theta}}). \tag{6.9}$$

Chapter 7

nPCA Using a Roughness Penalty

Ridge regression (Tikhonov regularization) [27] is a well known method used to circumvent numerical errors when trying to solve problems that are ill-posed. Ridge regression includes a regularization term in the minimization. Similarly, we include a penalty term in our proposed minimization. Therefore it is prudent to start with a short introduction to ridge regression.

7.1 Ridge Regression

A problem in ordinary Multiple Linear Regression is often that the predictors are collinear. Given the model in (6.1), $\hat{\mathbf{u}}_n$ can be be found with linear regression in the following manner,

$$\hat{\mathbf{u}}_n = (\hat{\mathbf{G}}^\mathsf{T}\hat{\mathbf{G}})^{-1}\hat{\mathbf{G}}^\mathsf{T}\mathbf{y}_n. \tag{7.1}$$

Having collinear predictors makes the $\mathbf{G}^\mathsf{T}\mathbf{G}$ matrix (often denoted as $\mathbf{X}^\mathsf{T}\mathbf{X}$) ill-conditioned which means that the determinant is nearly zero and calculating its inverse is highly susceptible to numerical errors. If one of the predictors is

a linear combination of other predictors, then the $\mathbf{G}$ matrix will not be of full rank and its inverse simply does not exist.

Ridge regression has been proposed to circumvent the problems of predictors collinearity [27]. In ridge regression, the least squares (LS) method for parameter estimation based on $\mathbf{G}^\mathsf{T}\mathbf{G}$ is not used but rather an estimation based on the matrix

$$[\mathbf{G}^\mathsf{T}\mathbf{G} + k\mathbf{I}], \qquad k \geq 0. \tag{7.2}$$

The determinant of this modified matrix will be appreciably different from zero.

This modification will introduce a bias to the parameter estimation, however the variance of the new parameters will be smaller than the parameters estimated by the least squares method. In fact, the variance may be reduced so much that their MSE may also be smaller than the MSE when LS is used. The prediction errors in the ridge model will also be more accurate if the predictors exhibit near collinearity.

However, an extra parameter has been introduced in the model. The parameter determines how much the ridge regression deviates from the LS regression. If the parameter is chosen to small, the collinearity will not be efficiently countered. If the parameter is to large, the bias of the estimated parameters will be to large. A optimum value for the parameter can often be found using cross-validation [28].

The only matrix inversion needed when calculating the nPCA solution with an EM algorithm are inversions of matrices that have the form of the matrix in (7.2).

7.2 Penalized Likelihood Estimation

In the framework of penalized likelihood estimation, we now need to find the maximum likelihood solution to a function called the penalized log-likelihood that has the following form

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \Psi(\mathbf{x}),$$

where

$$\Psi(\mathbf{x}) = \log p(\mathbf{y}|\mathbf{x}) - hR(x). \tag{7.3}$$

The first term in (7.3) is the complete log-likelihood function, which quantifies the disagreement between $\mathbf{x}$ and the measurements $\mathbf{y}$. The second term is a regularizing penalty function that penalizes an object $\mathbf{x}$ in accordance to how severely it breaks our given assumptions.

The parameter h is a regularization parameter and controls the tradeoff between the fit of the data and a desired property of $\hat{\mathbf{x}}$. The desired property may be a smoothness, sparseness or something else. Considering smoothness, if h is chosen small, $\hat{\mathbf{y}}$ will closely fit the data, but in the presence of noise, the estimate will be noisy. For large values of h, the estimate will usually be smooth with low noise, while sacrificing the closeness of the fit.

7.3 A First Order Roughness Penalty

Let us now consider the one dimensional problem where the elements of a vector $\mathbf{x}$ correspond to consecutive values. The vector $\mathbf{x}$ can be viewed as representing some smooth natural process. A natural way to measure roughness of $\mathbf{x}$ is by using the penalty function

$$R(\mathbf{x}) = \sum_{t=2}^{T} \frac{1}{2}(x_t - x_{t-1})^2. \tag{7.4}$$

This penalty function will assign a high cost when neighboring values differ greatly. This will discourage roughness and temporally smooth estimates will be preferred [29].

To translate this penalty function into matrix-vector form, we define the $(T-1) \times T$ matrix $\mathbf{D}$ as

$$\mathbf{D} = \begin{bmatrix} -1 & 1 & 0 & 0 & \dots & 0 \\ 0 & -1 & 1 & 0 & \dots & 0 \\ & & \ddots & & & \\ 0 & 0 & \dots & 0 & -1 & 1 \end{bmatrix}. \tag{7.5}$$

Next, we write

$$\mathbf{Dx} = \begin{bmatrix} x_2 - x_1 \\ \vdots \\ x_T - x_{T-1} \end{bmatrix}. \tag{7.6}$$

It can be seen that $[\mathbf{Dx}]_k = x_{k+1} - x_k$. Now (7.4) can be rewritten as

$$R(\mathbf{x}) = \sum_{k=1}^{T-1} \frac{1}{2}([\mathbf{Dx}]_k)^2 = \frac{1}{2}\|\mathbf{Dx}\|_F^2 = \frac{1}{2}\mathbf{x}^\mathsf{T}\mathbf{D}^\mathsf{T}\mathbf{Dx} = \frac{1}{2}\mathbf{x}^\mathsf{T}\mathbf{Rx} \tag{7.7}$$

where $\|\cdot\|_F$ is the Frobenius norm and $\mathbf{R}$ is a "nearly Toeplitz" matrix given by

$$\mathbf{R} = \begin{bmatrix} 1 & -1 & 0 & 0 & \dots & 0 \\ -1 & 2 & -1 & 0 & \dots & 0 \\ & & \ddots & & & \\ 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & -1 & 1 \end{bmatrix}. \tag{7.8}$$

$R(\mathbf{x})$ is quadratic in $\mathbf{x}$ and this type of penalty function is called a quadratic penalty.

Other more general penalty functions may be used instead of the $\mathbf{R}$ matrix used here. For example, when analyzing periodic data it could be more natural to use a harmonic acceleration operator [30].

7.4 nPCA Using a First Order Roughness Penalty

Now we will enforce a requirement of smoothness on $\mathbf{G}$ in our model,

$$\mathbf{y}_n = \boldsymbol{\mu} + \mathbf{G}\mathbf{u}_n + \boldsymbol{\epsilon}_n, \quad n = 0, \dots, M-1, \tag{7.9}$$

by adding a penalty term to the log-likelihood function [31]. The aim is now to solve a maximum penalized likelihood estimation. The log-likelihood function that we will be maximizing has the form

$$L(\boldsymbol{\theta}) - hL_p(\boldsymbol{\theta}), \tag{7.10}$$

where $L_p(\boldsymbol{\theta})$ is the roughness function and the non-negative parameter h is the regularization parameter. The penalty term used is a first order roughness penalty term, given by

$$L_p(\boldsymbol{\theta}) = \frac{1}{2\sigma^2 M}\|\mathbf{DG}\|_F^2, \tag{7.11}$$

where

$$\|\mathbf{DG}\|_F^2 = \mathrm{tr}\big((\mathbf{DG})^\mathsf{T}\mathbf{DG}\big) = \mathrm{tr}(\mathbf{G}^\mathsf{T}\mathbf{RG}), \tag{7.12}$$

and $\mathbf{D}$ and $\mathbf{R}$ are given in (7.5) and (7.8), respectively.

The model given in (6.1) is used and a first order roughness penalty term is added to the complete log-likelihood function (5.6) to obtain a new log-likelihood function,

$$L_{1\text{st}}(\boldsymbol{\theta}) = L(\boldsymbol{\theta}) - \frac{h}{2\sigma^2 M}\|\mathbf{DG}\|_F^2, \tag{7.13}$$

and an EM algorithm is used to maximize it. The EM functional is found by taking expectation of (7.13), conditioned on the data and the parameters, and can be written in the following manner

$$\begin{aligned}\text{EMf} = -\sum_{n=0}^{M-1}\Bigg\{ & \frac{T}{2}\log\sigma^2 + \frac{1}{2\sigma^2}(\mathbf{y}_n-\boldsymbol{\mu})^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) \\ & - \frac{1}{\sigma^2}\mathbf{z}_n^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}) + \frac{1}{2\sigma^2}\text{tr}(\mathbf{W}_n\mathbf{G}^{\mathsf{T}}\mathbf{G}) \\ & + \frac{h}{2\sigma^2}\|\mathbf{DG}\|_F^2\Bigg\},\end{aligned} \tag{7.14}$$

where we are using the same notation as described in (5.7)-(5.9). The non relevant terms have been omitted. This step, taking the expected value of the complete log-likelihood function, is what is called the E-step in the EM algorithm. To perform the maximization step (M-step) in the algorithm, the first differential of (7.14) with respect to $\mathbf{G}$ is first found, obtaining

$$\begin{aligned} d\text{EMf} &= -\sum_{n=0}^{M-1}\Bigg\{-\frac{1}{\sigma^2}\text{tr}\big(\mathbf{z}_n(\mathbf{y}_n-\boldsymbol{\mu})^{\mathsf{T}}d\mathbf{G}\big) \\ &\quad + \frac{1}{\sigma^2}\text{tr}\big(\mathbf{W}_n\mathbf{G}^{\mathsf{T}}d\mathbf{G}\big) + \frac{h}{\sigma^2}\text{tr}\big(\mathbf{D}^{\mathsf{T}}\mathbf{D}\mathbf{G}d\mathbf{G}\big)\Bigg\} \\ &= -\sum_{n=0}^{M-1}\Bigg\{-\frac{1}{\sigma^2}\text{tr}\big(((\mathbf{y}_n-\boldsymbol{\mu})\mathbf{z}_n^{\mathsf{T}} \\ &\quad + \mathbf{G}\mathbf{W}_n + h\mathbf{D}^{\mathsf{T}}\mathbf{D}\mathbf{G})d\mathbf{G}\big)\Bigg\}. \end{aligned} \tag{7.15}$$

Simplifying and setting (7.15) equal to zero gives

$$\left[Mh\mathbf{D}^{\mathsf{T}}\mathbf{D}\right]\mathbf{G} + \mathbf{G}\left[\sum_{n=0}^{M-1}\mathbf{W}_n^{\mathsf{T}}\right] = \left[\sum_{n=0}^{M-1}(\mathbf{y}_n-\boldsymbol{\mu})\mathbf{z}_n^{\mathsf{T}}\right]. \tag{7.16}$$

Using that the conditional distribution of $\mathbf{u}_n$ given $\mathbf{y}_n$ is

$$p(\mathbf{u}_n|\mathbf{y}_n) \sim \mathcal{N}(\mathbf{M}^{-1}\mathbf{G}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}), \sigma^2\mathbf{M}^{-1}), \tag{7.17}$$

and the fact that $\mathbf{W}_n = \text{cov}(\mathbf{u}_n) + \mathbf{z}_n\mathbf{z}_n^{\mathsf{T}}$ we obtain

$$\mathbf{z}_n = \mathbf{M}^{-1}\mathbf{G}^{\mathsf{T}}(\mathbf{y}_n-\boldsymbol{\mu}), \tag{7.18}$$

$$\mathbf{W}_n = \sigma^2\mathbf{M}^{-1} + \mathbf{z}_n\mathbf{z}_n^{\mathsf{T}}, \tag{7.19}$$

and $\mathbf{M} = \mathbf{G}^\mathsf{T}\mathbf{G} + \sigma^2\mathbf{I}$. Equation (7.16) is a *Sylvester equation* ($\mathbf{KG} + \mathbf{GL} = \mathbf{Q}$) and can be solved for $\mathbf{G}$ with the Bartels-Stewart algorithm by transforming $\mathbf{K}$ and $\mathbf{L}$ into Schur form by a QR algorithm, and then solving the resulting triangular system with back-substitution [32].

In a similar manner σ^2 is found by taking the expected value of the log-likelihood function, finding the derivative with respect to σ^2 and setting equal to zero. Giving

$$\sigma^2 = \frac{1}{T}\mathrm{tr}(\mathbf{S}_y) + \frac{1}{MT}\sum_{n=0}^{M-1}\Big\{-2\mathbf{z}_n^\mathsf{T}\mathbf{G}^\mathsf{T}\mathbf{\Phi}(\mathbf{y}_n - \boldsymbol{\mu}) + \mathrm{tr}(\mathbf{W}_n\mathbf{G}^\mathsf{T}\mathbf{G}) + h\,\mathrm{tr}(\mathbf{G}^\mathsf{T}\mathbf{D}^\mathsf{T}\mathbf{D}\mathbf{G})\Big\}, \tag{7.20}$$

where

$$\mathbf{S}_y = \frac{1}{M}\sum_{n=0}^{M-1}\mathbf{Y}^\mathsf{T}\mathbf{Y}$$

is the covariance matrix of $\mathbf{Y}$. Here $\mathbf{G}$ and σ^2 are found by iteratively calculating (7.16) and (7.20).

The EM algorithm for nPCA using a first order roughness penalty is given in Algorithm 2. The algorithm is implemented in Matlab.

Algorithm 2: EM algorithm for nPCA using a first order roughness penalty.

Initialize $\mathbf{G}_1$ and σ_1^2 randomly.
for $k = 1, \ldots$ **do**

$$\begin{aligned}
\mathbf{M}_k &= \mathbf{G}_k^\mathsf{T}\mathbf{G}_k + \sigma_k^2\mathbf{I}\\
\mathbf{F}_k &= \mathbf{M}_k^{-1}\mathbf{G}_k^\mathsf{T}\mathbf{\Phi}\mathbf{Y}\\
\mathbf{V}_k &= \mathbf{F}_k\mathbf{F}_k^\mathsf{T}\\
\mathbf{G}_{k+1} &= \mathrm{lyap}(Mh\mathbf{D}^\mathsf{T}\mathbf{D}, M\sigma_k^2\mathbf{M}_k^{-1} + \mathbf{V}_k, -\mathbf{Y}\mathbf{F}_k)\\
\sigma_{k+1}^2 &= \frac{1}{MT}\Big[\mathrm{tr}(\mathbf{Y}^\mathsf{T}\mathbf{Y}) - 2\mathrm{tr}(\mathbf{G}_{k+1}^\mathsf{T}\mathbf{\Phi}\mathbf{Y}\mathbf{F}_k^\mathsf{T})\\
&\quad + M\mathrm{tr}(\sigma_k^2\mathbf{M}_k^{-1}\mathbf{G}_{k+1}^\mathsf{T}\mathbf{G}_{k+1})\\
&\quad + \mathrm{tr}(\mathbf{G}_{k+1}^\mathsf{T}\mathbf{G}_{k+1}\mathbf{V}_k^\mathsf{T})\\
&\quad + Mh\mathrm{tr}(\mathbf{G}_{k+1}^\mathsf{T}\mathbf{D}^\mathsf{T}\mathbf{D}\mathbf{G}_{k+1})\Big].
\end{aligned}$$

Terminate when σ^2 and $\mathbf{G}$ have converged .

7.5 Choosing the Regularization Parameter

To choose a value for the regularization parameter we will use the cross-validation (CV) method. The idea behind CV is to separate the data into two sets, a training set and a testing set.

The risk estimate we wish to minimize is

$$\text{Risk} = \frac{1}{M}\sum_{n=0}^{M-1} \|\mathbf{G}\mathbf{u}_n - \hat{\mathbf{G}}\hat{\mathbf{u}}_n\|^2. \tag{7.21}$$

In practice, we cannot directly estimate (7.21) because we do not know the value of $\mathbf{G}\mathbf{u}_n$. Instead we will use CV to estimate a prediction error. We will be estimating how well the model found with the training data predicts the test data. The prediction error is given by

$$\text{PE}_h = \frac{1}{M_q}\sum_{n=0}^{M_q-1} \|\mathbf{y}_n - \hat{\mathbf{y}}_n\|^2 = \frac{1}{M_q}\sum_{n=0}^{M_q-1} \|\mathbf{y}_n - \hat{\mathbf{G}}\hat{\mathbf{u}}_n\|^2, \tag{7.22}$$

where M_q is the size of the test set. Algorithm 2 is used to find $\hat{\mathbf{G}}$ and we will use simple linear regression to estimate $\hat{\mathbf{u}}_n$ for the CV method, that is

$$\hat{\mathbf{u}}_n = (\hat{\mathbf{G}}^\mathsf{T}\hat{\mathbf{G}})^{-1}\hat{\mathbf{G}}^\mathsf{T}\mathbf{y}_n. \tag{7.23}$$

To implement the CV, the data is split into Q equally large data sets. The training data consists of $Q-1$ data sets and the testing data consists of the one data set not belonging to the training set. Then PE_h for different values of h is calculated. This is repeated Q times, choosing a different test set each time, and the value of h that gives the minimum average PE_h is chosen. Algorithm 3 illustrates this procedure.

Algorithm 3: Cross-validation algorithm for choosing the regularization parameter h.

Initialize by splitting the data set randomly into Q equally large data sets: $D(q), \quad q = 1, \ldots, Q$

for $q = 1, \ldots, Q$ **do**

- Set test data
- $D_{test} = D(q)$
- Set test data
- $D_{train} = D(n \neq q), \quad n = 1, \ldots, Q$
- **for** all predefined values of h **do**
  - Calculate $\hat{\mathbf{G}}$ using D_{train}
  - Calculate PE_h using $\hat{\mathbf{G}}$ and D_{test}

Choose the value of h that results in the minimum average PE_h.

Chapter 8

Experimental Results

We will evaluate and compare the two methods in the previous chapters, nPCA using a First Order Roughness Penalty and nPCA using Basis Expansion. To evaluate the methods, we will use both simulated data and real fMRI data. We will begin with the simulation data.

8.1 Simulation

8.1.1 The Simulated Data

Two signals ($\mathbf{s}_1$ and $\mathbf{s}_2$) of length $T = 100$, with zero mean and unit norm are created and shown in Figure 8.1.

The simulation data is

$$\mathbf{Y} = \mathbf{G}\mathbf{u} + \boldsymbol{\epsilon}, \tag{8.1}$$

where $\mathbf{G} = [\mathbf{s}_1\ \mathbf{s}_2]$, $\boldsymbol{\epsilon}$ is white Gaussian noise with variance σ^2, and $\mathbf{u} = [\mathbf{g}_1\ \mathbf{g}_2]^\mathsf{T}$, where $\mathbf{g}_1$ and $\mathbf{g}_2$ are all the elements in two 64×64 matrices, respectively, regarded as single columns. The upper part of the first matrix is set to one and the lower part in the second is set to one. The areas in the matrices having ones are overlapping. $\mathbf{Y}$ is set to have zero mean.

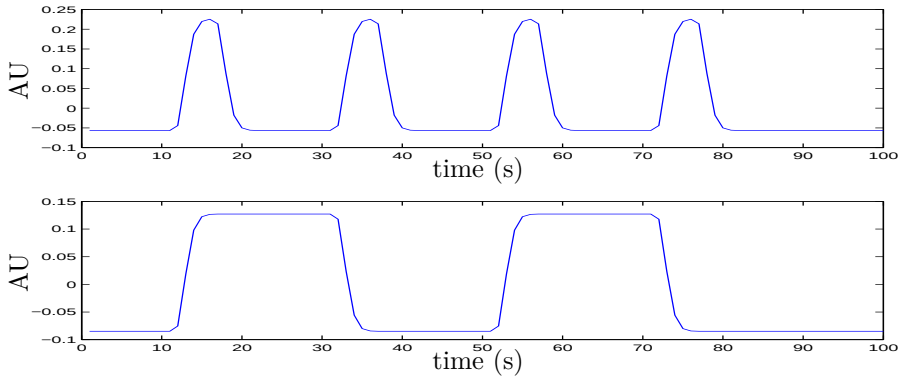

Figure 8.1: The two signals $\mathbf{s}_1$ (upper) and $\mathbf{s}_2$ (lower) used in the simulation.

For testing purposes the number of principal components are assumed to be known. The test data created has two signals and that is the number of components considered. In real world scenarios, the number of components is not known and has to be estimated in some manner.

To evaluate the performance of the estimators, the mean squared error defined as

$$\text{MSE} = \|\mathbf{G}\mathbf{u} - \hat{\mathbf{G}}\hat{\mathbf{u}}\|_F^2, \tag{8.2}$$

will be calculated. $\hat{\mathbf{G}}$ and $\hat{\sigma}^2$ are found using the estimators and $\hat{\mathbf{u}}$ is given in (7.18),

$$\hat{\mathbf{u}} = \text{E}[\mathbf{u}] = (\hat{\mathbf{G}}^\mathsf{T}\hat{\mathbf{G}} + \hat{\sigma}^2\mathbf{I})^{-1}\hat{\mathbf{G}}^\mathsf{T}\mathbf{Y}. \tag{8.3}$$

In most cases, calculating the MSE is not possible since the correct value of $\mathbf{G}$ is not available. However, for testing purposes it may be valuable to consider the MSE. In addition the "smoothness" of the principal components will be evaluated visually.

The following is used to calculate the signal to noise ratio

$$\text{SNR} = \frac{\text{E}[\mathbf{u}_n^\mathsf{T}\mathbf{G}^\mathsf{T}\mathbf{G}\mathbf{u}_n]}{\text{E}[\boldsymbol{\epsilon}_n^\mathsf{T}\boldsymbol{\epsilon}_n]} = \frac{\text{tr}(\text{E}[\mathbf{u}_n\mathbf{u}_n^\mathsf{T}]\mathbf{G}^\mathsf{T}\mathbf{G})}{\text{tr}(\sigma^2\mathbf{I}_T)} = \frac{\text{tr}(\mathbf{G}^\mathsf{T}\mathbf{G})}{T\sigma^2}. \tag{8.4}$$

Figures 8.2 and 8.3 depict two columns of the test data matrix $\mathbf{Y}$ where the signals $\mathbf{s}_1$ and $\mathbf{s}_2$ are apparent.

In Figure 8.4 the first 50 eigenvalues of $\mathbf{Y}$ can be seen when $\sigma^2 = 0.1^2$. This plot is called a scree plot of eigenvalues and is often used to determine the number of PCs. The scree plot of eigenvalues in Figure 8.4 shows the fraction of total

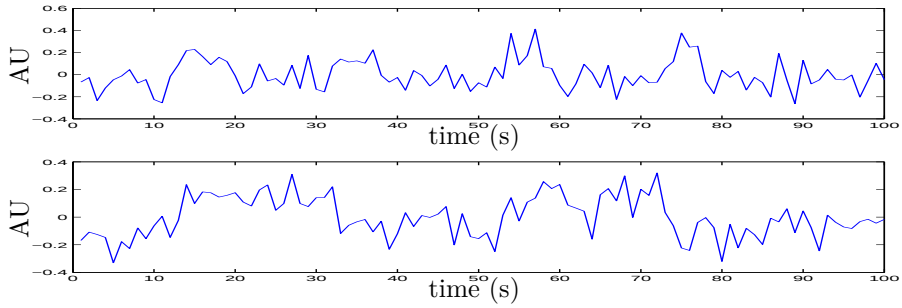

Figure 8.2: Depicted here are two columns in the test data matrix $\mathbf{Y}$, y_1 and y_2 with added noise, $\sigma^2 = 0.1^2$.

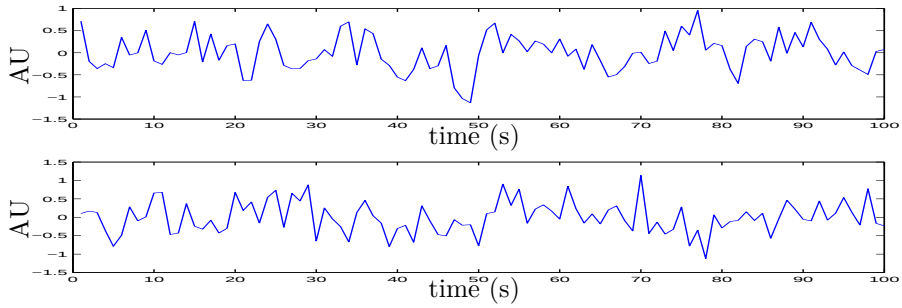

Figure 8.3: Depicted here are two columns in the test data matrix $\mathbf{Y}$, y_1 and y_2 with added noise, $\sigma^2 = 0.4^2$.

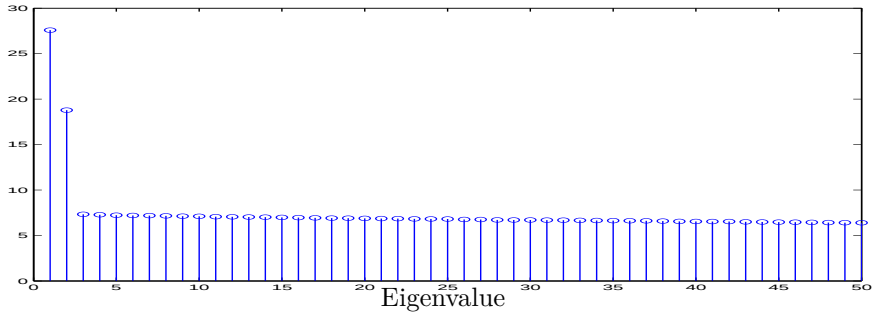

Figure 8.4: First 50 eigenvalues of $\mathbf{Y}$ when $\sigma^2 = 0.1^2$.

variance in the data as explained or represented by each PC. In this case, the first two eigenvalues are much larger than the rest indicating that two PCs is

appropriate for this data, which we know to be correct.

In Sections 8.1.5.1 and 8.1.5.2 alternate simulation data sets are used. The data sets is similar to the data set described above, however with the matrices of size 12×12, with and with different vectors, $\mathbf{s}_n$. The $\mathbf{u}$ vectors now $\mathcal{N}(0, \mathbf{I}_r)$. The new signals are depicted in Figures 8.5 and 8.6.

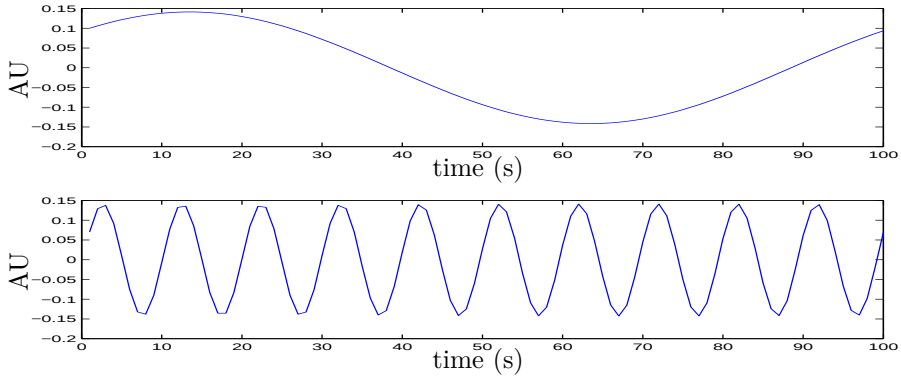

Figure 8.5: Simulation data set number two. The two sine signals $\mathbf{s}_1$ and $\mathbf{s}_2$ used.

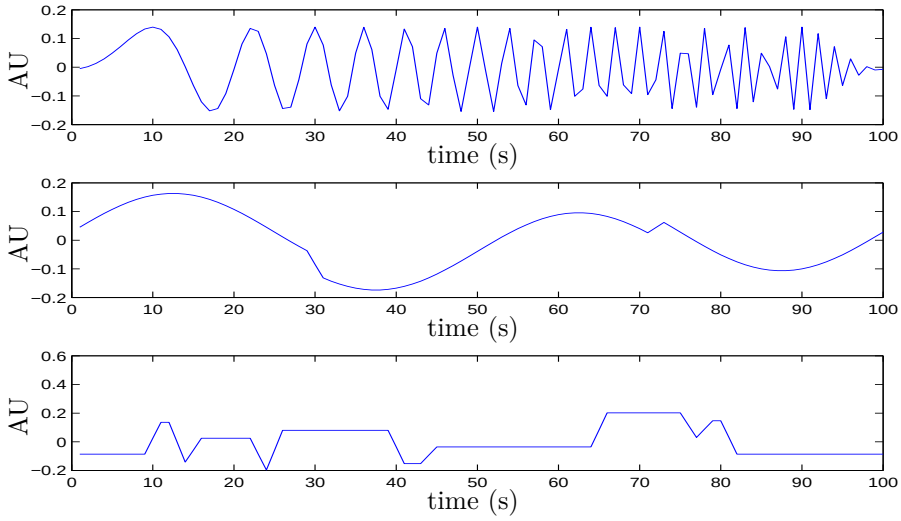

Figure 8.6: Simulation Data set number three. The three signals $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$ used.

8.1.2 nPCA Using Basis Expansion

The $\mathbf{\Phi}$ matrix will be truncated by removing columns. By truncating the $\mathbf{\Phi}$ matrix, high frequency components from the solution will be removed.

Noise is added to the model in (8.1). Setting $\sigma^2 = 0.3^2$ gives SNR$= -2.05$dB according to (8.4). In Figure 8.7, a plot of the MSE against the number of basis functions, m, is shown. In Figure 8.8, a plot of the BIC given in (6.9) is shown. The minimum BIC is when $m = 71$. The MSE error stays approximately the same when $m \geq 31$, however the minimum MSE is when $m = 100$. The BIC also assigns approximately the same value all $m \geq 31$, which means that the BIC criteria hardly distinguishes between choosing any $m \geq 31$. Choosing few basis functions can however introduce unwanted oscillations in the results, this is apparent when $m = 31$ is chosen. Model selection using the basis expansion approach is detailed in [13].

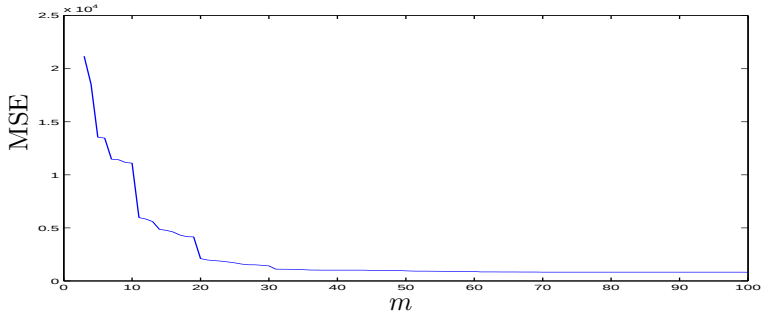

Figure 8.7: The MSE plotted against the number of basis functions used.

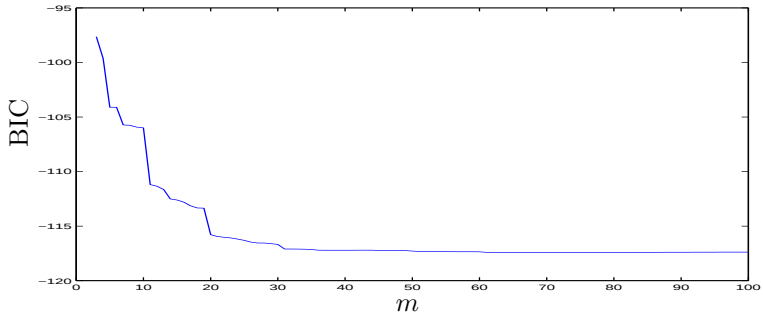

Figure 8.8: The BIC plotted against the number of basis functions used.

The basis expansion method is a very useful PCA method and has proven to

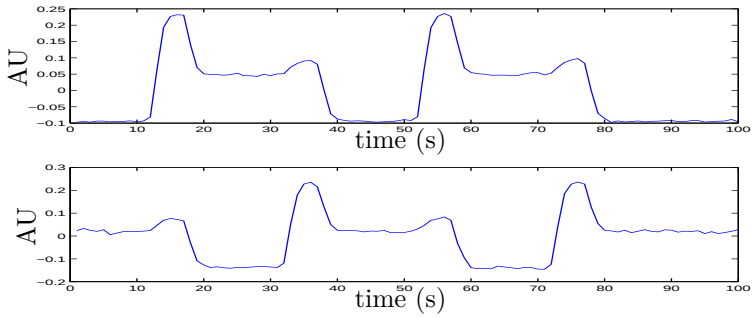

Figure 8.9: The first two PCs when all 100 basis functions are used.

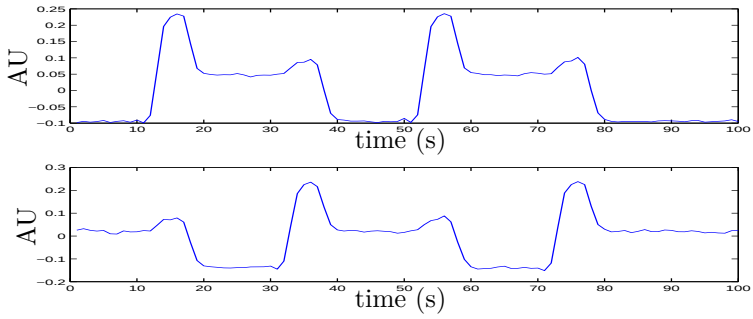

Figure 8.10: The first two PCs when the first 71 basis functions are used. Choosing $m = 71$ results in the minimum of the BIC.

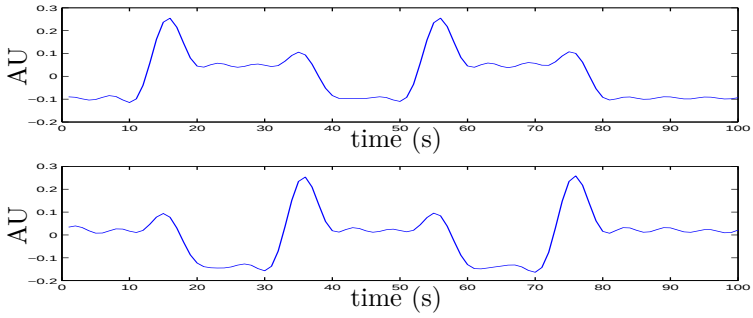

Figure 8.11: The first two PCs when the first 31 basis functions are used. Choosing fewer than $m = 31$ will increase the MSE significantly.

give good results using fMRI data [13]. A closed form solution is available which means that a iterative algorithm is not necessary. This, combined with the fact that removing columns from the basis (Φ) matrix reduces the complexity of the computations, means that the time needed to calculate the solution is much shorter than calculating solutions requiring iterative algorithms, such as the EM algorithm.

Care must be taken not to truncate the basis excessively, which will result in a over smoothed result with unwanted oscillations.

8.1.3 nPCA Using a first Order Roughness Penalty

We will now evaluate the CV method. The simulation data is used with added noise. This will be done by adding different amounts of noise to model (8.1). The SNR will range from -22.5dB to 13.5dB. The number of datasets is $Q = 10$.

The CV method is used to calculate a value for the smoothing parameter h. We would like the value of h found by the CV method to minimize the MSE. In Figure 8.12, the values of h found using CV are shown and also the values of h that gives the minimum MSE using (8.2).

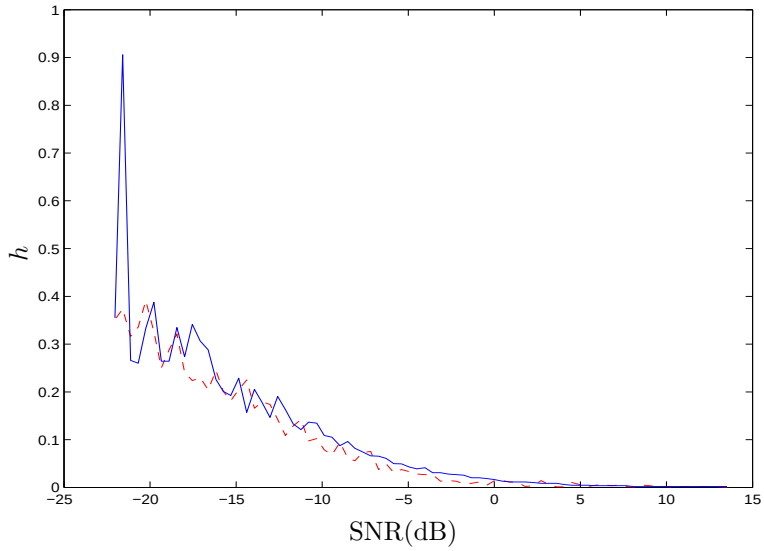

Figure 8.12: The solid line shows the values of h found using cross-validation. The dashed line shows the values of h that minimize the MSE.

The values of h found using CV are very close to the values found by minimizing the MSE when the SNR> 0dB. When the SNR drops below zeros, the CV method does not find the exact value that values but it does give very satisfactory results.

The estimate of the variance $\hat{\sigma}^2$ is very close to the true value, the relative difference between the true value and the estimate is order of magnitude 10^{-3} for all SNR values.

With increasing noise levels, the optimal value for the smoothing constant, h, that resulted in a minimum MSE increases. Increasing the noise (lowering the SNR) calls for larger a value for the smoothing parameter.

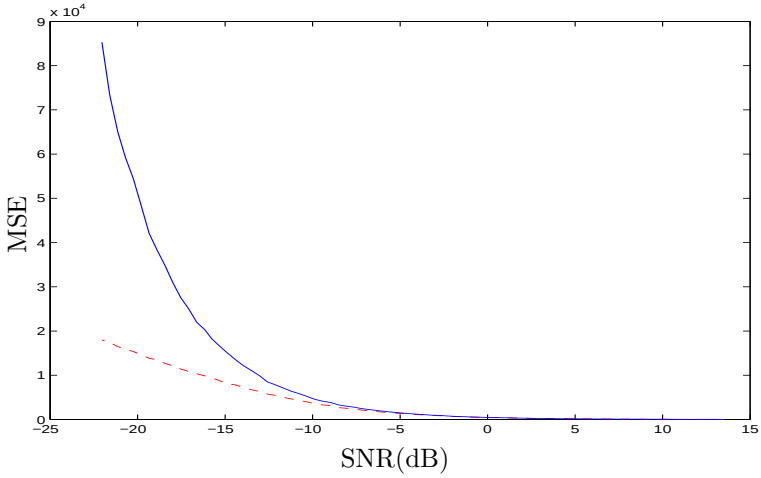

Figure 8.13: The solid line shows the MSE when $\hat{\mathbf{u}}$ is found using (7.23). The dashed line shows the MSE when $\hat{\mathbf{u}}$ is found using (7.18).

The MSE is higher when $\hat{\mathbf{u}}$ is found using (7.23) then when $\hat{\mathbf{u}}$ is found using (7.18) for low SNR values. However when the SNR is larger than 0dB, the MSE for both methods converge.

Now, the smoothing parameter, h, will be changed and the effects it has on the MSE will be more closely examined. The SNR will be between -10.5dB and 7.5dB, and the smoothing parameter will be varied from 0 to 0.25. Plots of the MSE for selected SNR values are shown in Figure 8.14.

The first two principal components are plotted for the selected SNR values are shown in Figure 8.15. For each SNR, three values of h are chosen, $h = 0$, $h = 1$

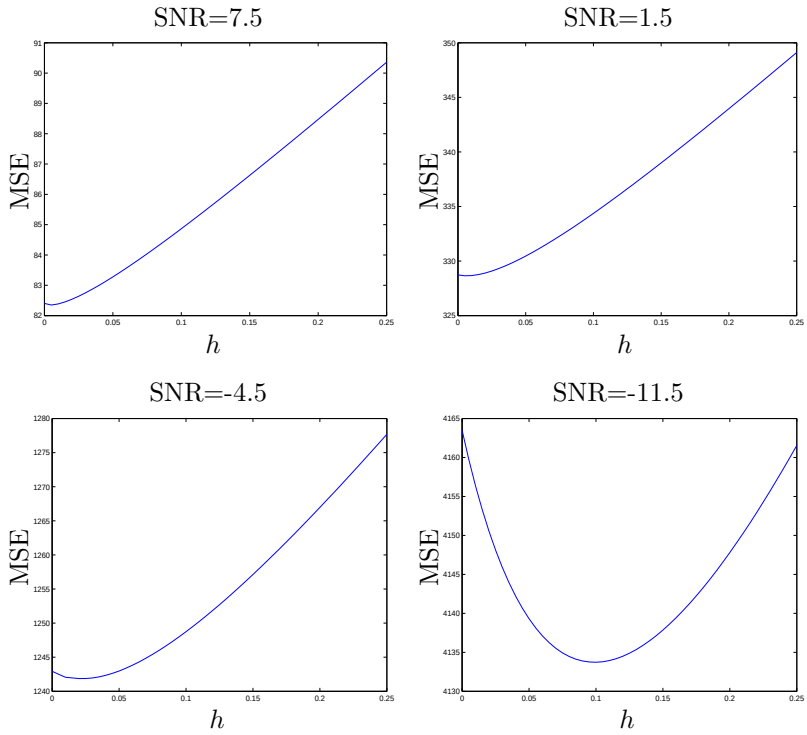

Figure 8.14: MSE plotted against the parameter h for SNR values 7.5dB, 1.5dB, -4.5dB and -11.5dB with minimum values of h, 0.005, 0.005, 0.02 and 0.1 respectively.

and h found using cross-validation with $Q = 10$ data sets.

Table 8.1 shows the values of h found using cross validation. For SNR values larger then approximately 0, CV gives a fairly accurate result but for low SNR values, CV results is values of h that are larger the value that results in the minimum MSE.

SNR(dB)	7.5	1.5	-4.5	-11.5
h_{mse}	0.005	0.005	0.02	0.1
h_{cv}	0.005	0.01	0.045	0.14

Table 8.1: The values of h found using CV and by minimizing the MSE.

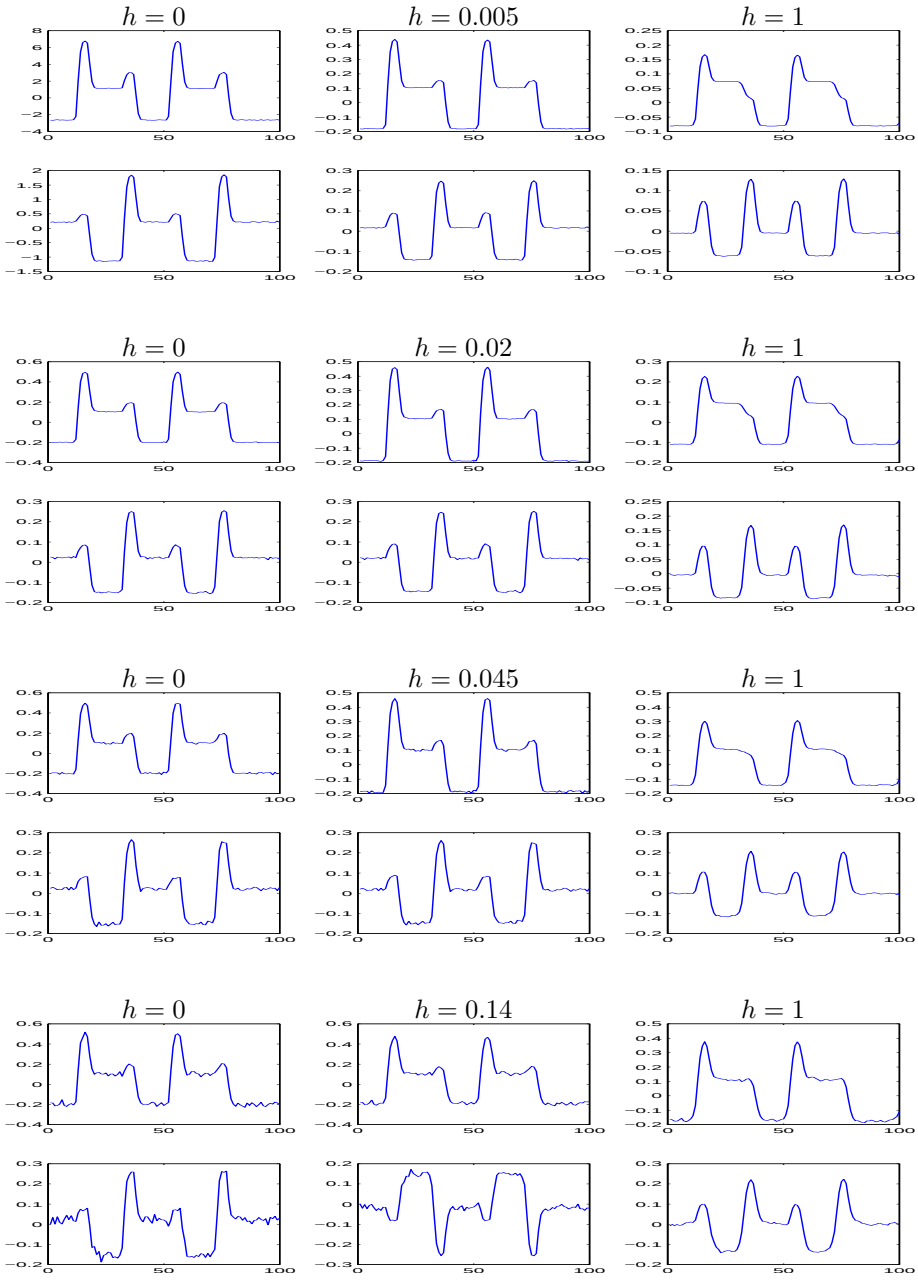

Figure 8.15: The first two PCs when $SNR = 7.5$dB (top), 1.5dB, -4.5dB and -11.5dB (bottom).

8.1.4 Convergence and Computations

The convergence of the log-likelihood function can be seen in Figure 8.16. The simulated data is used with SNR=-2dB. The smoothing constant found using CV is $h = 0.0225$. The values of h used in the CV ranged from $h = 0$ to $h = 0.25$ in 0.0025 increments. The log-likelihood function is strictly increasing and converges in 210 iterations. On an Intel 3.6Ghz Core Duo E8400 desktop computer with 2Gb of memory, the algorithm converged in $\approx 0.52s$. Calculating the CV however took 127.9s.

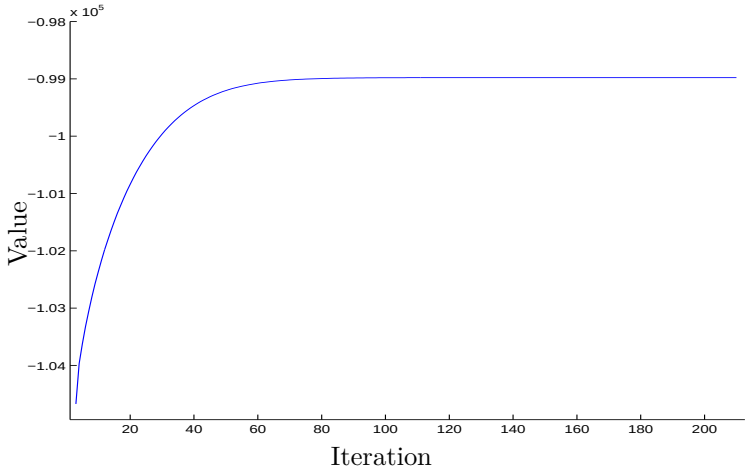

Figure 8.16: The log-likelihood function converges to its final value in 210 iterations.

The number of iterations needed for the algorithm to converge to its final value and computational time needed is dependent on a number of issues, such as:

- the size of the data,
- the number of principal components,
- the value of the smoothing constant,
- the variance of the noise in the data.

Solving the Sylvester equation is the most computationally demanding and time consuming part of the algorithm, approximately 35% of time is spent on this issue. The Matlab routine, Lyap(), is used for solving the Sylvester equation.

8.1.5 Comparison of Methods

Now a comparison of the basis expansion method and the roughness penalty method will be done. To find the number of basis functions we will use the BIC method. For the roughness penalty method, the smoothing parameter, h is found using cross-validation. A plot of the MSE against SNR is shown in Figure 8.17. The number of PCs is two for both methods.

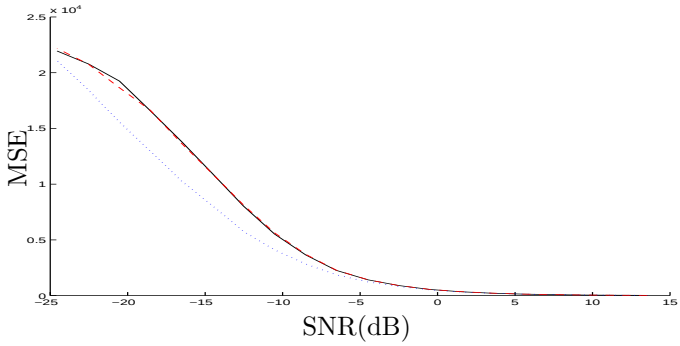

Figure 8.17: A comparison of the MSE for the methods. Normal PCA is solid, basis expansion method is dashed and the roughness penalty method is dotted.

The basis expansion method and normal PCA have similar MSE while the roughness penalty methods has the lowest MSE. This becomes more apparent with lower SNR. When the SNR is higher than zero, the MSE for all methods is similar.

In Figures 8.18 and 8.19, the first two PCs when SNR $= -14.5$dB and SNR $= -22.5$dB are shown, respectively. When SNR$= -14.5$dB, the PCs are similar for both methods, however the basis expansion method favors a slightly smoother solution. With low SNR values, this difference becomes more apparent, the solution is much smoother and unwanted oscillations appear in the PCs.

As was expected, neither method is able to separate the test signals, respectively, into PCs. This is because we do not attempt to identify the rotation matrix in the PCA solution, it is chosen as the identity matrix. We can see that the simulation signals (shown in Figure 8.1) are mixed in the first two PCs in all the solutions. Other methods, such as Independence Component Analysis (ICA) [33], are able to separate to signals into their respective PCs.

Minimizing the MSE is not the only way to evaluate the performance of the methods. Visually inspecting the results and choosing a smoothing parameter based on other criteria may be feasible.

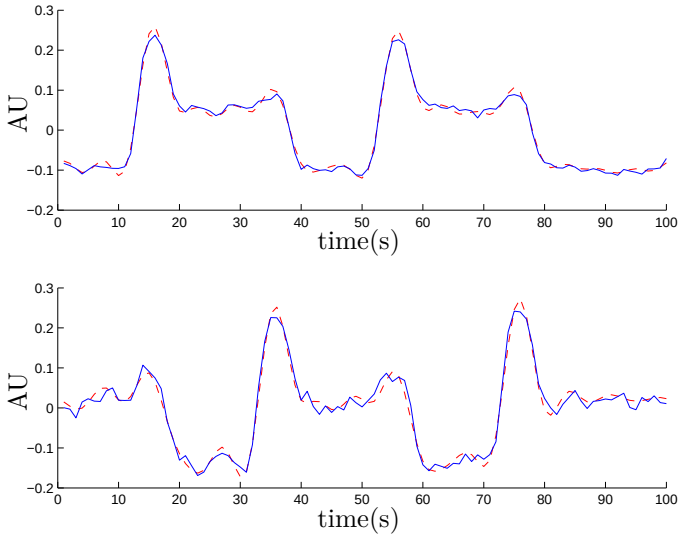

Figure 8.18: The first two PCs when SNR = −14.5dB. The basis expansion method is dashed and the roughness penalty method is solid.

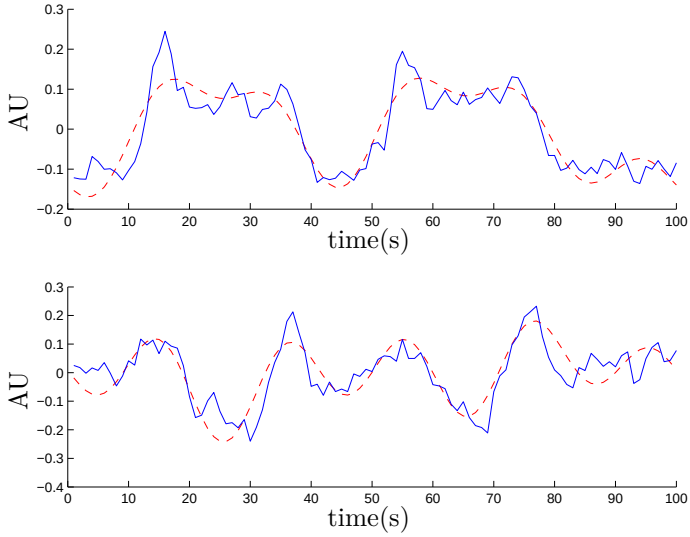

Figure 8.19: The first two PCs when SNR = −22.5dB. The basis expansion method is dashed and the roughness penalty method is solid.

8.1.5.1 Simulation Data Set Number Two

Now a comparison of the two methods will be done using the sine data set described in the end of section 8.1.1. Again we vary the noise and compare the methods. In Figure 8.20 we see that the cross-validation method chooses a parameter that is higher than the one resulting in the minimum MSE.

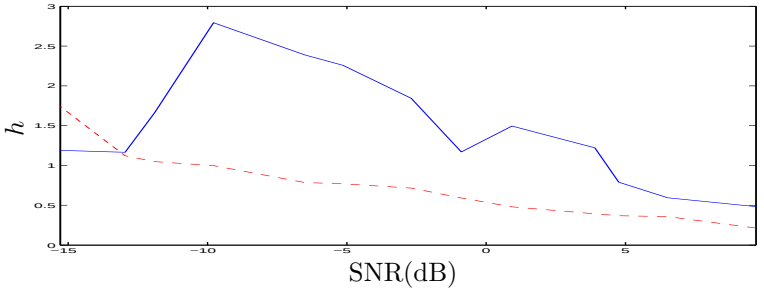

Figure 8.20: The solid line shows the values of h found using cross-validation. The dashed line shows the values of h that minimize the MSE.

In Figure 8.21 the MSE is depicted. On this data set both methods perform similarly and have a lower MSE than normal PCA. However, the basis expansion method has a slightly lower MSE than the roughness penalty method.

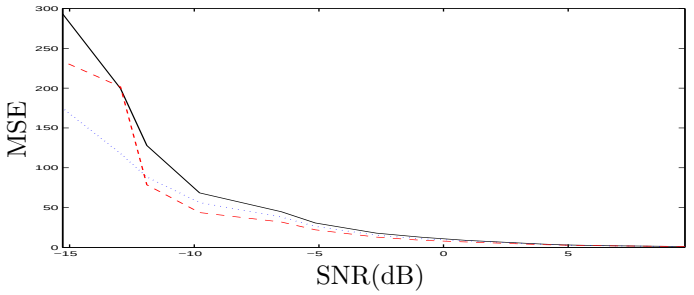

Figure 8.21: A comparison of the MSE for the methods. Normal PCA is solid, basis expansion method is dashed and the roughness penalty method is dotted.

In Figures 8.22 and 8.23, the first two principal components when SNR = 4.7dB and SNR = −9.8dB are shown, respectively.

The basis expansion method has a slight slightly lower MSE than the roughness penalty method. However the basis expansion method does not do a good job

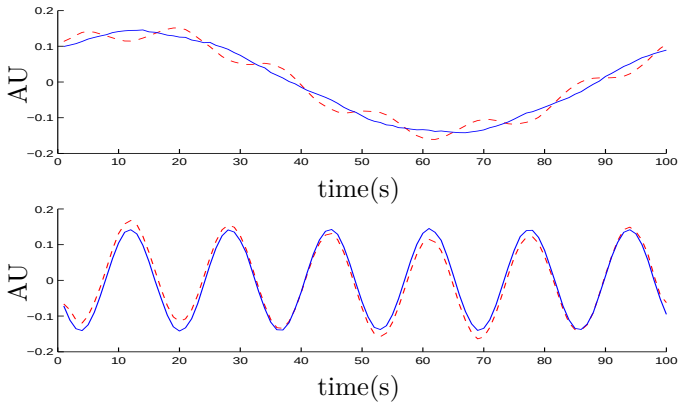

Figure 8.22: The first two PCs when SNR = 4.7dB. The basis expansion method is dashed and the roughness penalty method is solid.

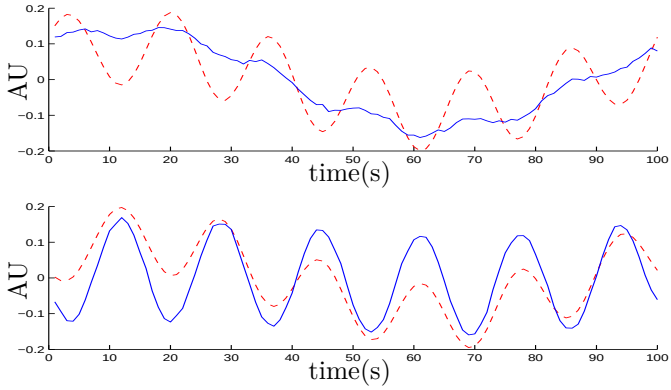

Figure 8.23: The first two PCs when SNR = −9.8dB. The basis expansion method is dashed and the roughness penalty method is solid.

of separating the signals into their respective PCs. The oscillations in the first PC in Figures 8.22 and 8.23 show that the signals are mixed. The same thing is apparent in for the roughness penalty method in Figure 8.23 but in a much lower degree.

8.1.5.2 Simulation Data Set Number Three

We repeat the procedure again with the third data set. In Figure 8.24 we see that the cross-validation method again chooses a parameter that is a bit higher than the one resulting in the minimum MSE.

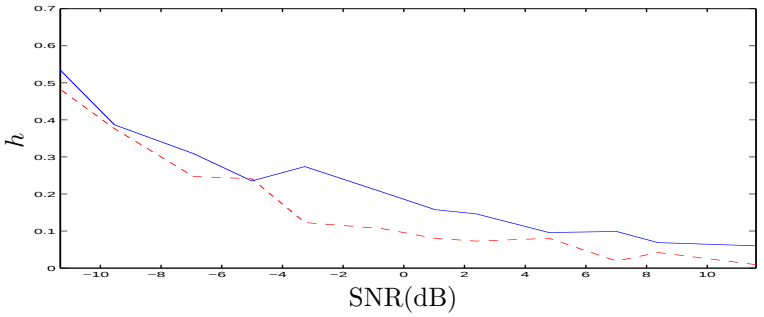

Figure 8.24: The solid line shows the values of h found using cross-validation. The dashed line shows the values of h that minimize the MSE.

In Figure 8.25 the MSE is depicted. Both methods perform similarly when the SNR is larger than zero, however when the SNR becomes lower than approximately 4dB the basis expansion method chooses very few bases and the MSE rises. The roughness penalty method has the lowest MSE.

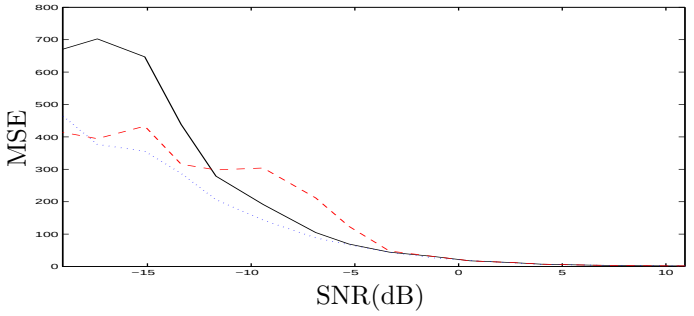

Figure 8.25: A comparison of the MSE for the methods. Normal PCA is solid, basis expansion method is dashed and the roughness penalty method is dotted.

In Figures 8.26 and 8.27, the first two PCs when SNR = 7.1dB and SNR = −7.6dB are shown, respectively.

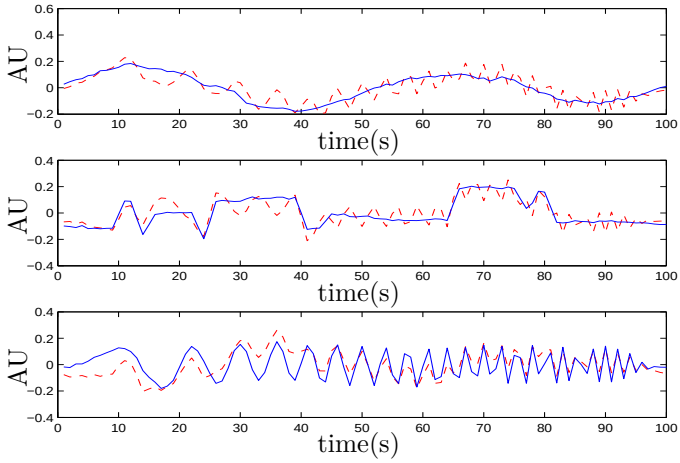

Figure 8.26: The first two PCs when SNR = 7.1dB. The basis expansion method is dashed and the roughness penalty method is solid.

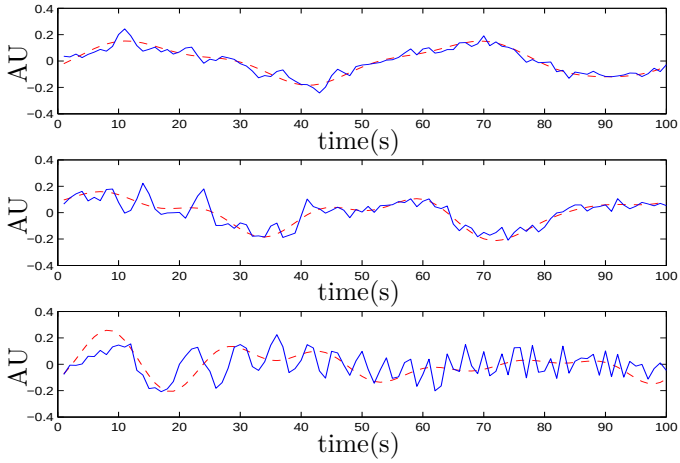

Figure 8.27: The first two PCs when SNR = −7.6dB. The basis expansion method is dashed and the roughness penalty method is solid.

Again we see that the basis expansion method has trouble separating the signals into their respective PCs, this is clearly apparent in Figure 8.26. In Figure 8.27, we see the effects of choosing few bases for the basis expansion method. It results in over smoothing and the oscillations in the third PC (comparable to signal one in Figure 8.6) have all but been removed.

8.1.5.3 Comparison Summary

Three datasets where used in the comparison of the basis expansion method and the roughness penalty method. With all three datasets, the cross validation method is able to select a useful value for the smoothing parameter h. With the first dataset it is able to choose values very close to those that minimize the MSE. With the latter two datasets a value is choses slightly higher than the optimum value (in the MSE sense).

The roughness penalty method results in a lower MSE for the first and third dataset while the basis expansion method results in a slightly lower MSE for the second dataset.

The PCs are similar for the first dataset for both methods, with the basis expansion method favoring a more smooth solution. For the second and third dataset, the roughness penalty method is better able to separate the signals into their respective PCs.

A closed form solution is available for the basis expansion method and not the roughness penalty method. This being the case, the basis expansion method is much faster and less computationally intensive than the roughness penalty method.

8.2 fMRI data

We will now evaluate the methods on real *Functional Magnetic Resonance Imaging* (fMRI) data. fMRI uses Magnetic Resonance scanner to measure brain functions. A person is put in a large magnetic field and radio waves are transmitted into the person. The waves affect the magnetic field of the hydrogen molecules in water. The frequency of these waves are then measured by a radio receiver. The frequency of the waves is changed by changing the magnetic field while they are being received. These frequency changes allows the images to be created.

The data used here was created with a 3T MRI scanner with 2 sec TR. TR is the time between reading data from the same location in the brain. The data consists of 100 time points and $M = 4096 = 64^2$ voxels (volumetric pixels). An fMRI experiment was performed where a patient performed sequential finger-thumb opposition according to a stimulus signal. The images received have dimensions 64×64. One brain image and the stimulus signal are shown in Figure 8.2.

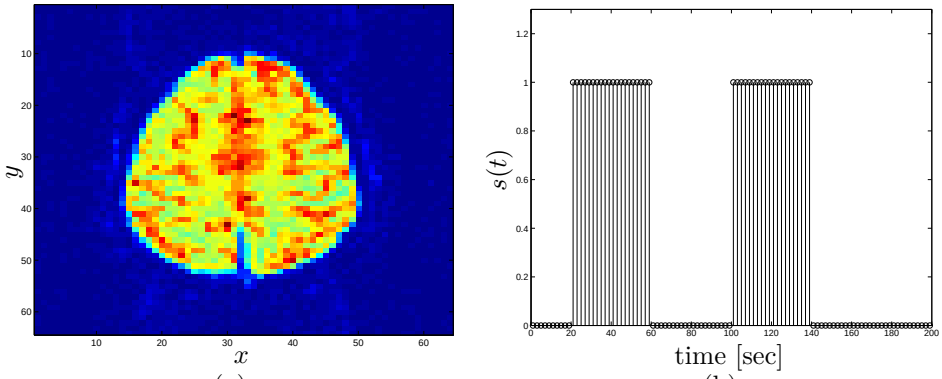

Figure 8.28: A fMRI brain image (a), and the stimulus signal (b).

Blood Oxygen Level Dependency (BOLD) fMRI is the most common fMRI technique. The technique measures the hemodynamic response (change in blood flow) related to neural activity. The magnetic susceptibilities of oxyhemoglobin and deoxyhemoglobin differ slightly. BOLD fMRI is based on this difference. After neural activation in the brain, oxygen rich blood flows into the area of activation thus increasing the amount of oxyhemoglobin in the area. This leads to a rise in intensity of the observed signal.

We are using BOLD fMRI data here. Change in blood flow in the brain is a natural smooth process, and BOLD fMRI is detecting change in blood flow it

is valuable to be able to incorporate this knowledge about the signal into the solution, that is, constraining the solution to be smooth function.

A plot of the first three PC's using normal PCA is in Figure 8.29. By visually comparing the PC's to the stimulus function and calculating the correlation between the stimulus and the PC's it shown that the second PC corresponds best to the stimulus signal.

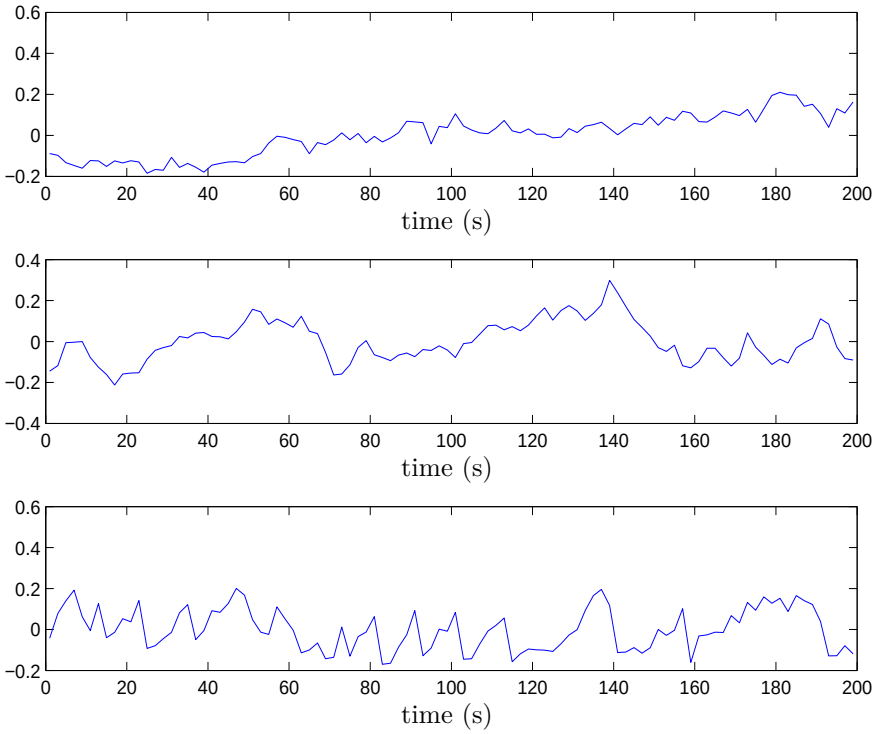

Figure 8.29: The first three PC's found using normal PCA.

The BIC according to (6.9) is calculated. The minimum of the BIC is when the number of principal components is 5 and the number of basis functions is 48. A plot of the BIC is shown in Figure 8.30

To find a estimate of a suitable value for the smoothing parameter, the cross-validation method is used. PE_h has minimum when $h = 0.475$. The second PC calculated with $h = 0.475$ and using the basis expansion method with parameters found by minimizing the BIC can be seen in Figure 8.32.

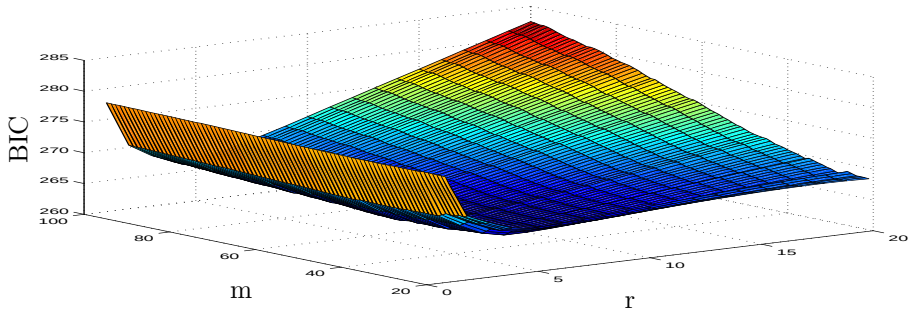

Figure 8.30: The BIC. The minimum occurs when the number of PC's is $r = 5$ and the number of basis functions is $m = 48$.

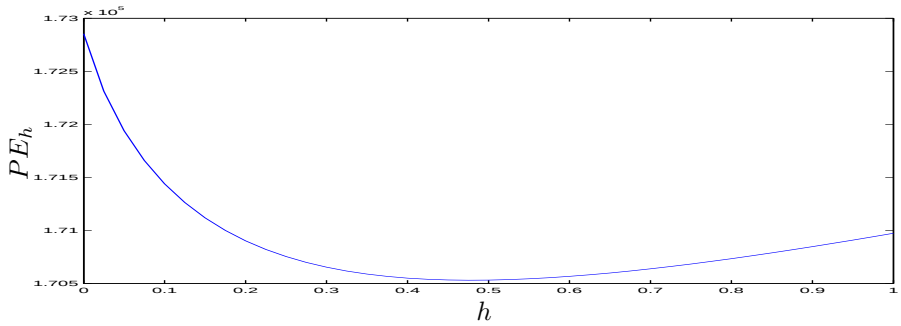

Figure 8.31: The average PE_h error when calculating h. The minimum occurs when $h = 0.475$.

Both the roughness penalty method and the basis expansion method yield similar solutions as can be seem in Figure 8.32.

Choosing the h parameter using CV gives satisfactory results here but other criteria may also be used such as visually inspecting the results. The second PC along with the stimulus signal for $h = 1, 5, 10$ is shown in Figure 8.33.

To find the regions of the fMRI image that are most responsible for the second PC we now create an activation map. The smoothing parameter h is chosen as 0.475 and the second PC is regressed on the fMRI data. A spatial plot of the regressed data is given in Figure 8.34. The plot has high values where the fit to the second PC is good. We can see that the quality of the fit is highest in three regions of the brain. These three regions are the motor cortex regions and are known to be connected to hand-eye coordination.

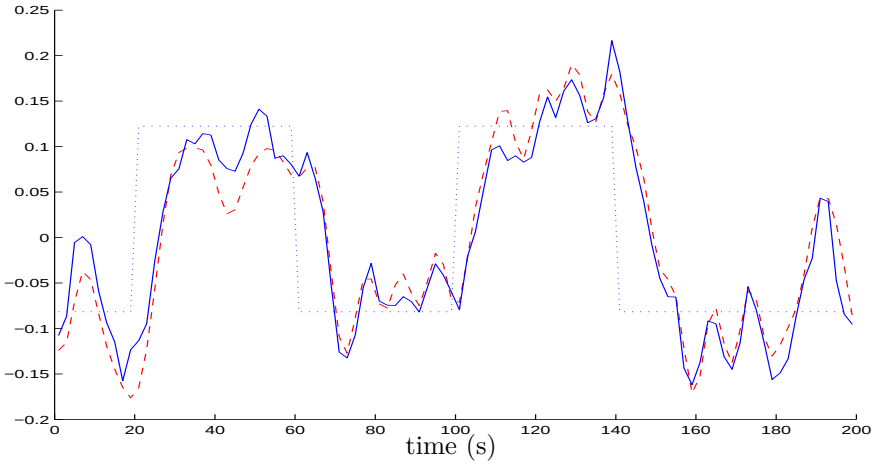

Figure 8.32: The second PC calculated using the roughness penalty method (blue) with $h = 0.475$ and using the basis expansion (red) method with parameters found using the BIC. The stimulus signal is dashed.

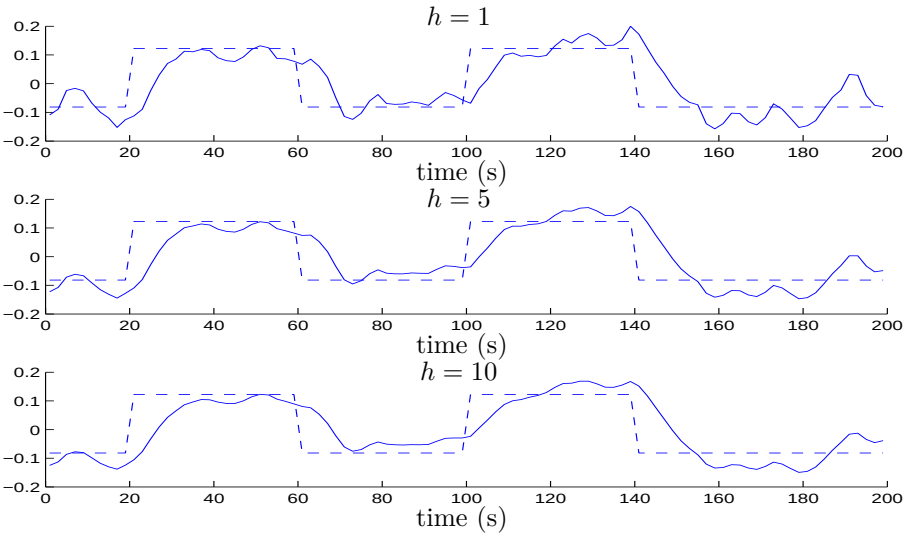

Figure 8.33: The second PC calculated with $h = 1, 5, 10$.

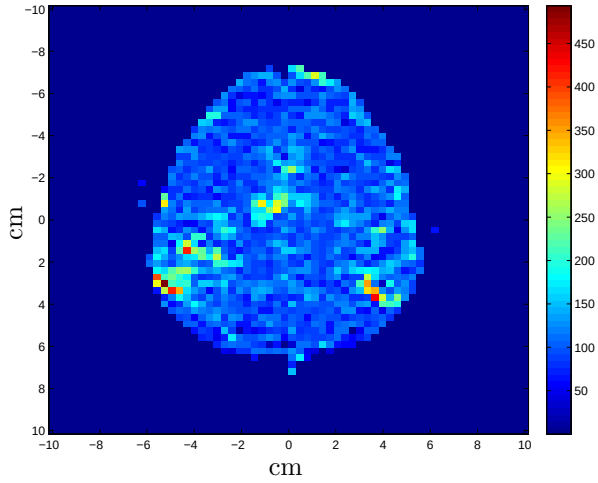

Figure 8.34: A spatial plot of the second principal component regressed on the fMRI data.

Chapter 9

Conclusions and Further Work

A method for smooth nPCA has been developed. This method adds a first order penalty terms to the log-likelihood function of nPCA using basis expansion. This penalized log-likelihood function is then maximized with an EM algorithm.

The degree of smoothness can be changed with a smoothing parameter. A cross-validation method for determining the smoothing parameter is used and gave good results, both using simulated and real fMRI data. More general roughness penalties can easily be used without changing the estimation algorithm.

The method showed improvements for simulated noisy data in the sense that the MSE could be reduced, compared to normal PCA, if the value of the smoothing parameter is chosen correctly. Noise was removed from the data without sacrificing to much of the signal characteristics.

The method was tested on real fMRI data and showed promising results. The smoothness of the principal component of interest was increased.

9.1 Further work

There are a number of issues that need further work, such as:

- The number of principal components needs to be determined.
- Using cross-validation is very computationally intensive, finding a less intensive method would be preferable. Other methods that will be investigated are Stein's unbiased risk estimate (SURE) [34] and nearly unbiased risk estimation (NURE) [35].
- In this work, the rotation matrix in the PCA solution was chosen as the identity matrix. We would like to devise a method for identifying the correct rotation matrix in the PCA solution. Identifying the rotation matrix will help in separating signals from different sources. We will compare this method to ICA (Independent Component Analysis), which is better equipped to separate signals from different sources.
- Testing the method in other data sets in different fields.

Appendix A

Matrix Calculus

A.1 Differentials

Maximum likelihood problems often require derivatives. Calculated matrix derivatives using partials can be tedious, they can however be computed in a much easier manner using matrix manipulations.

Let us define the equation

$$f(x) = y(x + dx) - y(x).$$

The differential $dy(x)$ is the part of $f(x)$ that is linear in dx. As an example, the equation,

$$\mathbf{y}(\mathbf{x} + d\mathbf{x}) = \mathbf{y}(\mathbf{x}) + \mathbf{A}d\mathbf{x} + (\text{higher order terms}).$$

is for example well defined, given that $\mathbf{y}$ satisfied certain continuity properties. The matrix $\mathbf{A}$ is the derivative, called the Jacobian matrix $\mathbf{J}_{x \to y}$.

The derivative of any expression involving matrices can be computed in two steps:

1. Compute the differential.

2. Manipulate the result into canonical form.

Following the two steps, the derivative can be read as a coefficient of dx, $d\mathbf{x}$ or $d\mathbf{X}$ ($\mathbf{x}$ is a column vector and $\mathbf{X}$ is a matrix).

A.2 Important differentials

The fundamental rules of calculus apply to matrix calculus. Assuming that $\mathbf{V}$ and $\mathbf{U}$ are matrix functions and $\mathbf{A}$ is a matrix of real constants, the following rules apply [36]:

$$
\begin{aligned}
d\mathbf{A} &= 0 \\
d(\alpha\mathbf{U}) &= \alpha d\mathbf{U} \\
d(\mathbf{U}+\mathbf{V}) &= d\mathbf{U} + D\mathbf{V} \\
d(\mathbf{U}-\mathbf{V}) &= d\mathbf{U} - Dv \\
d(\mathbf{U}\mathbf{V}) &= (d\mathbf{U})\mathbf{V} + \mathbf{U}d\mathbf{V} \\
d\mathbf{U}^T &= (d\mathbf{U})^\mathsf{T} \\
d\mathrm{tr}(\mathbf{U}) &= \mathrm{tr}(d\mathbf{U})
\end{aligned}
$$

Bibliography

[1] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, pp. 559–572, 1901.

[2] I. T. Jolliffe, *Principal Component Analysis*, Springer, New York, NY, 2nd edition, October 2002.

[3] F. Castells, P. Laguna, L. Sörnmo, A. Bollmann, and J. M. Roig, "Principal component analysis in ECG signal processing," *EURASIP J. Appl. Signal Process.*, vol. 2007, no. 1, pp. 98–98, 2007.

[4] D.B. Rowe and R.G. Hoffmann, "Multivariate statistical analysis in fMRI," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 25, no. 2, pp. 60 – 64, march-april 2006.

[5] D. N. Lawley, "A modified method of estimation in factor analysis and some large sample results," *Nordisk Psykologi Monograph Series. In Uppsala Symposium on Psychological Factor Analysis*, , no. 3, pp. 35–42, 1953.

[6] M. Tipping and C. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society. Series B, statistical methodology*, vol. 61, no. 3, pp. 611–622, 1999.

[7] Michael E. Tipping and Christopher M. Bishop, "Mixtures of Probabilistic Principal Component Analysers," *Neural Computation*, vol. 11, no. 2, pp. 443–482, 1999.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[9] T. K. Moon, "The expectation-maximization algorithm," *Signal Processing Magazine*, vol. 13, no. 6, pp. 47 –60, nov 1996.

[10] A. K. Khambampati, A. Rashid, B. S. Kim, D. Liu, S. Kim, and K. Y. Kim, "Em algorithm applied for estimating non-stationary region boundaries using electrical impedance tomography," *Journal of Physics: Conference Series*, vol. 224, no. 1, pp. 012044, 2010.

[11] D.L. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289 –1306, april 2006.

[12] M. O. Ulfarsson and V. Solo, "Sparse variable noisy PCA using l0 penalty," *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 3950 –3953, mar. 2010.

[13] M. O. Ulfarsson and V. Solo, "Smooth principal component analysis with application to functional magnetic resonance imaging," *Proceedings of Acoustics, Speech and Signal Processing. ICASSP.*, vol. 2, pp. II–II, May 2006.

[14] S. M. Kay, *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory*, Prentice Hall PTR, Upper Saddle River, NJ, USA, March 1993.

[15] R. A. Fisher, "On the mathematical foundations of theoretical statistics," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 222, pp. 309–368, 1922.

[16] S. Borman, "The expectation maximization algorithm – a short tutorial," July 2004.

[17] P. J. Green, "On use of the EM for penalized likelihood estimation," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 52, no. 3, pp. pp. 443–452, 1990.

[18] T. M. Cover and J. A. Thomas, *Elements of information theory*, Wiley-Interscience, New York, NY, USA, 1991.

[19] C. F. Jeff Wu, "On the convergence properties of the EM algorithm," *The Annals of Statistics*, vol. 11, no. 1, pp. 95–103, 1983.

[20] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*, Springer, New York, NY, USA, corrected edition, July 2003.

[21] J. Shlens, "A tutorial on principal component analysis," December 2005.

[22] D. J. Bartholomew, *Latent variable models and factor analysis*, Oxford University Press, Inc., New York, NY, USA, 1987.

[23] A Basilevsky, *Statistical Factor Analysis and Related Methods: Theory and Applications*, Wiley Series in Probability and Statistics. Wiley, Hoboken, NJ, 1994.

[24] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1 edition, August 2006.

[25] H. Akaike, "A new look at the statistical model identification," *Automatic Control, IEEE Transactions on*, vol. 19, no. 6, pp. 716–723, January 2003.

[26] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, March 1978.

[27] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. pp. 55–67, 1970.

[28] G. H. Golub, M. Heath, and G. Wahba, "Generalized cross-validation as a method for choosing a good ridge parameter," *Technometrics*, , no. 21, pp. 215–223, 1979.

[29] P. J. Green and B. W. Silverman, *Nonparametric Regression and Generalized Linear Models. A Roughness Penalty Approach*, Chapman and Hall, New York, NY, first edition, 1994.

[30] J. Ramsay and B. W. Silverman, *Functional Data Analysis (Springer Series in Statistics)*, Springer, 2nd edition, June 2005.

[31] J. Sigurdsson and M. O. Ulfarsson, "Smooth noisy PCA using a 1st order roughness penalty," in *Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on*, September 2010, pp. 325 –330.

[32] R. H. Bartels and G. W. Stewart, "Solution of the matrix equation ax + xb = c [f4]," *Commun. ACM*, vol. 15, no. 9, pp. 820–826, 1972.

[33] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, Wiley-Interscience, New York, NY, USA, 1 edition, May 2001.

[34] C. M. Stein, "Estimation of the mean of a multivariate normal distribution," *The Annals of Statistics*, vol. 9, no. 6, pp. pp. 1135–1151, 1981.

[35] V. Solo, "Transfer function order estimation with a H_∞ criterion," *Proceedings of the 37th IEEE Conference on Decision and Control.*, vol. 4, pp. 4472 –4473 vol.4, dec. 1998.

[36] J. R. Magnus and H. Neudecker, *Matrix differential calculus with applications in statistics and econometrics*, John Wiley & Sons, West Sussex, England, 3rd edition, 2007.