Kernel Principal Component Regression with EM Approach to Nonlinear Principal Components Extraction

Roman Rosipal
Leonard J. Trejo
Andrzej Cichocki

TECHNICAL REPORT

University of Paisley
School of Information and Communication Technologies
Paisley, PA1 2BE
Scotland, UK

November 2000

Kernel Principal Component Regression with EM Approach to Nonlinear Principal Components Extraction

Roman Rosipal
Computational Intelligence Research Unit
School of Information and Communication Technologies
University of Paisley
Paisley PA1 2BE, Scotland
rosi-ci0@paisley.ac.uk

Leonard J. Trejo
Computational Sciences Division
NASA Ames Research Center
Moffett Field, CA
ltrejo@mail.arc.nasa.gov

Andrzej Cichocki
Laboratory for Advanced Brain Signal Processing
Brain Science Institute RIKEN
2-1, Hirosawa, Wako-Shi, Japan
cia@brain.riken.go.jp

Part of the presented work was completed during the stay of the first author at the Laboratory for Advanced Brain Signal Processing, Brain Science Institute, RIKEN, Japan. The author would like to thank the organizers of Summer Program 2000 for providing this possibility.

Contents

1 Introduction **5**

2 RKHS and Representer Theorem **7**

3 Kernel PCA **9**

4 An EM Approach to Kernel PCA **11**

5 Feature Space Regularized Linear Regression **13**
5.1 Kernel Principal Component Regression 13
5.2 Kernel Ridge Regression 15
5.3 Summing Up 17

6 Data Sample Construction **17**
6.1 Chaotic Mackey-Glass Time-Series 17
6.2 Human Signal Detection Performance Monitoring 18

7 Results **19**
7.1 Chaotic Mackey-Glass Time-Series 20
7.2 Human Signal Detection Performance Monitoring 25

8 Discussion and Conclusions **32**

Acknowledgments **33**

References **34**

Appendix A **36**

Appendix B **38**

Abstract

In kernel based methods such as Support Vector Machines, Kernel PCA, Gaussian Processes or Regularization Networks the computational requirements scale as $\mathcal{O}(n^3)$ where n is the number of training points. In this paper we investigate Kernel Principal Component Regression (KPCR) with the Expectation Maximization approach in estimating of the subset of p principal components ($p < n$) in a feature space defined by a positive definite kernel function. The computational requirements of the method are $\mathcal{O}(pn^2)$. Moreover, the algorithm can be implemented with memory requirements $\mathcal{O}(p^2) + \mathcal{O}((p+1)n))$.

We give the theoretical description explaining how by the proper selection of a subset of non-linear principal components desired generalization of the KPCR is achieved. On two data sets we experimentally demonstrate this fact. Moreover, on a noisy chaotic Mackey-Glass time series prediction the best performance is achieved with $p \ll n$ and experiments also suggests that in such cases we can also use significantly reduced training data sets to estimate the non-linear principal components.

The theoretical relation and experimental comparison to Kernel Ridge Regression and ϵ-insensitive Support Vector Regression is also given.

1 Introduction

The kernel-based solution to nonlinear regression has attracted the attention of many researchers. As a result, a unifying framework for Support Vector Machines, Regularization Networks, Gaussian processes and spline methods was given [8, 38, 39, 31, 12]. Although the methods were derived from different theoretical assumptions a straightforward connection between a Reproducing Kernel Hilbert Space (RKHS) and the corresponding feature space representation of the transformed input data gave rise to this unification. In Section 2 we will give a basic description of RKHS and feature space and briefly discuss the main results on non-linear regression tasks formulated in a RKHS.

However, the main drawbacks of these methods is the computational cost of finding the solution which scales as $\mathcal{O}(n^3)$ where n is the number of the training examples. Recently, several different approaches to overcome the problem were proposed [26, 30, 42]. In [21, 23] we proposed the analog of the Principal Component Regression (PCR) method in a high dimensional feature space defined by a positive definite kernel function. We named the method Kernel PCR (KPCR). KPCR is based on the orthogonal rotation of the original regressors in a feature space. The rotation is defined by the $p \leq n$ eigenvectors found by the Kernel Principal Component Analysis (KPCA) method [27, 28]. Although this approach can be very useful in the case where we are dealing with highly correlated/multicorrelated regressors, in general, it does not overcome the computational cost as we still need to diagonalize an $(n \times n)$ matrix. However, as we will show later, the KPCR method belongs to the class of so-called

shrinkage estimators where the lower variance of the estimated regression coefficients is achieved by exploiting only a subset of the eigenvectors which generally correspond to the largest eigenvalues. Moreover, in [23] we also experimentally demonstrated that in some cases the estimation of the eigenvectors from a lower number $m < n$ training data points followed by projection of the remainder of the training data to the extracted eigenvectors does not significantly degrade the final performance. This gives rise to the question of the possibility of using algorithms for iterative estimation of the first p eigenvectors either from the whole $(n \times n)$ or reduced $(m \times m)$ matrices. To this end, in this paper we investigate the performance of KPCR using the non-linear principal components extracted by the Expectation Maximization KPCA (EMKPCA) algorithm originally proposed in [20]. The computational complexity of the algorithm is $\mathcal{O}(pm^2)$ and we do not need to store the whole $(n \times n)$ matrix . In fact the memory requirements are $\mathcal{O}(p^2) + \mathcal{O}((p+1)n))$.

The classical PCR technique is a well known shrinkage estimator designed to deal with multicollinearity (see e.g. [6, 18, 10]). The multicollinearity or near-linear dependence of regressors is a serious problem which can dramatically influence the effectiveness of a regression model. Multicollinearity results in large variances and covariances for the least-squares estimators of the regression coefficients. Multicollinearity can also produce estimates of the regression coefficients that are too large in absolute value. Thus the values and signs of estimated regression coefficients may change considerably given different data samples. This effect can lead to a regression model which fits the training data reasonably well, but in general poor generalization of the model can occur. This fact is in a very close relation to the argument stressed in [31], where the authors have shown that choosing the *flattest* function[1] in a feature space can, based on the smoothing properties of the selected kernel function, lead to a smooth function in the input space.

Two methods which deal with multicollinearity are Ridge Regression (RR) and PCR. We will give the theoretical basis of (Kernel) PCR and will also highlight the relation to the Kernel RR (KRR) [25, 5, 3] technique in a kernel defined feature space. Moreover, we will show that the final solution of KPCR leads to the same form as defined by the Representer Theorem [13] which unifies the solutions of the Support Vector Machines, Regularization Networks, Gaussian processes and spline methods in a RKHS.

On two data sets, the chaotic Mackey-Glass time series prediction and on the problem of estimation of the human signal detection performance from the Evoked Related Potentials (ERPs), we compared KPCR, KRR and Support Vector Regression (SVR)[2] [38, 29, 3] techniques. We demonstrate that both KPCR and KRR techniques achieves similar results, however, in the case of KPCR the final linear model in a feature space is significantly smaller; i.e. only

[1] The *flatness* is defined in the sense of penalizing high values of the regression coefficients estimate.

[2] In this paper we are assuming a SVR model with the $\epsilon-$insensitive cost function.

$p < n$ eigenvectors are utilized. On the real data set, the performance of the KPCR and KRR models using the quadratic cost function is slightly superior to SVR. This suggests that on that particular data set a Gaussian type of noise is more likely; i.e. the regression models with a quadratic cost function are preferable.

In Section 2 a basic definition of a RKHS and formulation of the Representer Theorem is given. Section 3 describes the Kernel PCA algorithm and also shows the connection to the recently used Nyström approximation of the eigenfunction decomposition of a kernel function. In Section 4 the EM approach to Kernel PCA is given. KPCR and KRR are described and compared in Section 5. The question of an unpenalized bias term in the case of KRR is also discussed. Section 6 describes the data sets and the results are given in Section 7. Section 8 provides a short discussion and concludes the paper.

2 RKHS and Representer Theorem

The common aim of Support Vector Machines, Regularization Networks, Gaussian processes and spline methods is to address the poor generalization properties of existing regression techniques. To overcome this problem a regularized formulation of regression is considered as a variational problem

$$\min_{f \in \mathcal{H}} R_{reg}(f) = \frac{1}{n} \sum_{i=1}^{n} V(y_i, f(\mathbf{x}_i)) + \xi \|f\|^2_{\mathcal{H}} \tag{1}$$

leading to a solution of the form

$$f(\mathbf{x}) = \sum_{i=1}^{n} c_i K(\mathbf{x}_i, \mathbf{x}) \tag{2}$$

for any differentiable loss function V [39, 5]. ξ is a positive number (regularization term) to control the tradeoff between approximating properties and the smoothness of f. We assume a training set of regressors $\{\mathbf{x}_i\}_{i=1}^n$ to be a subset of a compact set $X \subseteq R^N$ and $\{y_i\}_{i=1}^n \in R$ to be a set of corresponding outputs. $\|f\|^2_{\mathcal{H}}$ is a norm (sometimes called "stabilizer" in regularization networks domain) in a RKHS $\mathcal{H}$ [1] defined by the positive definite kernel $K(\mathbf{x}, \mathbf{y})$; i.e. a symmetric function of two variables satisfying the Mercer theorem conditions [15]. The fact that for any such positive definite kernel there exists a unique RKHS is well established by the *Moore-Aronszjan theorem* [1]. The form $K(\mathbf{x}, \mathbf{y})$ has the following *reproducing property*

$$f(\mathbf{y}) = \langle f(\mathbf{x}), K(\mathbf{x}, \mathbf{y}) \rangle_{\mathcal{H}} \quad \forall f \in \mathcal{H}$$

where $\langle ., . \rangle_{\mathcal{H}}$ is the scalar product in $\mathcal{H}$. The function K is called *reproducing kernel* for $\mathcal{H}$.

It follows from Mercer's theorem that each positive definite kernel $K(\mathbf{x}, \mathbf{y})$ can by written in the form

$$K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{M} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{y}) \quad M \leq \infty \tag{3}$$

where $\{\phi_i(.)\}_{i=1}^{M}$ are the eigenfunctions of the integral operator $\mathbf{\Gamma}_K : L_2(X) \rightarrow L_2(X)$

$$(\mathbf{\Gamma}_K f)(\mathbf{x}) = \int_X K(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) d\mathbf{y} \;\; \forall f \in L_2(X)$$

and $\{\lambda_i > 0\}_{i=1}^{M}$ are the corresponding positive eigenvalues. The sequence $\{\phi_i(.)\}_{i=1}^{M}$ creates an orthonormal basis of $\mathcal{H}$ and we can express any function $f \in \mathcal{H}$ as $f(\mathbf{x}) = \sum_{i=1}^{M} a_i \phi_i(\mathbf{x})$ for any $a_i \in R$. This allows us to define a scalar product in $\mathcal{H}$:

$$\langle \sum_{i=1}^{M} a_i \phi_i(\mathbf{x}), \sum_{i=1}^{M} b_i \phi_i(\mathbf{x}) \rangle_{\mathcal{H}} \equiv \sum_{i=1}^{M} \frac{a_i b_i}{\lambda_i}$$

and the norm $\|f\|_{\mathcal{H}}^2 = \sum_{i=1}^{M} \frac{a_i^2}{\lambda_i}$. Rewriting (3) in the form

$$K(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{M} \sqrt{\lambda_i} \phi_i(\mathbf{x}) \sqrt{\lambda_i} \phi_i(\mathbf{y}) = (\Phi(\mathbf{x}), \Phi(\mathbf{y}))_{\mathcal{F}}$$

it becomes clear that any kernel $K(\mathbf{x}, \mathbf{y})$ also corresponds to a canonical (Euclidean) dot product in a possibly high dimensional space $\mathcal{F}$ where the input data are mapped by $\Phi : R^N \rightarrow \mathcal{F}$. The space $\mathcal{F}$ of all functions that are spanned by the basis $\{\phi_i(.)\}_{i=1}^{M} \in \mathcal{H}$ is usually de-noted as a *feature space*. The number of basis functions $\phi_i(.)$ defines the dimensionality of $\mathcal{F}$. It is worth noting, that we can also construct a RKHS and a corresponding feature space by choosing a sequence of linearly independent functions (not necessary orthogonal) $\{\psi_i(\mathbf{x})\}_{i=1}^{M}$ and positive numbers α_i to define a series (in the case of $M = \infty$ absolutely and uniformly convergent)

$$K(\mathbf{x}, \mathbf{y}) := \sum_{i=1}^{M} \alpha_i \psi_i(\mathbf{x}) \psi_i(\mathbf{y}).$$

This also gives the connection between the RKHS and Gaussian processes [39] where the K is assumed to represent the correlation function of a zero-mean Gaussian process evaluated at points $\mathbf{x}$ and $\mathbf{y}$.

Until now, we assumed that K is a positive definite kernel. However, the above results can be extended even for the case when K is a positive semidefinite. Is such a case a RKHS $\mathcal{H}$ contain a subspace of functions f with a zero norm

$\|f\|^2_{\mathcal{H}}$ (the null space). It was shown in [13] that in such a case the solution of (1) has the more general form

$$f(\mathbf{x}) = \sum_{i=1}^{n} c_i K(\mathbf{x}_i, \mathbf{x}) + \sum_{j=1}^{l} b_j \zeta_j(\mathbf{x}) \tag{4}$$

where the functions $\{\zeta_j(.)\}_{j=1}^{l}$ span the null space of $\mathcal{H}$ and the coefficients $\{c_i\}_{i=1}^{n}$, $\{b_j\}_{j=1}^{l}$ are again given by the data. For the purposes of this paper we will assume $l = 1$ and $\zeta_1(\mathbf{x}) = const \ \ \forall \mathbf{x}$. Using the positive definite kernel K, similar to the Gaussian kernel, leads to the construction of the RKHS with an empty null space. To make the regression techniques discussed later invariant with respect to the variable locations it may be very useful to redefine a kernel K to contain a null space of constant functions. We will return to this point in Section 5.

3 Kernel PCA

The PCA problem in a high-dimensional feature space $\mathcal{F}$ can be formulated as the diagonalization of an n-sample estimate of the covariance matrix

$$\hat{\mathbf{C}} = \frac{1}{n} \sum_{i=1}^{n} \Phi(\mathbf{x}_i)\Phi(\mathbf{x}_i)^T, \tag{5}$$

where $\Phi(\mathbf{x}_i)$ are centered nonlinear mappings of the input variables $\{\mathbf{x}_i\}_{i=1}^{n} \in R^N$ (we will return to the point of centralization of the data in $\mathcal{F}$ at the end of this section). The diagonalization represents a transformation of the original data to new coordinates defined by orthogonal eigenvectors $\mathbf{V}$. We have to find eigenvalues $\lambda \geq 0$ and non-zero eigenvectors $\mathbf{V} \in \mathcal{F}$ satisfying the eigenvalue equation

$$\lambda \mathbf{V} = \hat{\mathbf{C}} \mathbf{V}.$$

Realizing, that all solutions $\mathbf{V}$ with $\lambda \neq 0$ lie in the span of mappings $\Phi(\mathbf{x}_1)$ $, \ldots, \Phi(\mathbf{x}_n)$, the equivalent eigenvalue problem was derived [27, 28]

$$n\lambda \boldsymbol{\alpha} = \mathbf{K}\boldsymbol{\alpha}, \tag{6}$$

where $\boldsymbol{\alpha}$ denotes the column vector with coefficients $\alpha_1, ..., \alpha_n$ such that

$$\mathbf{V} = \sum_{i=1}^{n} \alpha_i \Phi(\mathbf{x}_i) \tag{7}$$

and $\mathbf{K}$ is a symmetric $(n \times n)$ *Gram* matrix with the elements

$$\mathrm{K}_{ij} = (\Phi(\mathbf{x}_i).\Phi(\mathbf{x}_j)) := K(\mathbf{x}_i, \mathbf{x}_j).$$

Normalizing the solutions $\mathbf{V}^k$ corresponding to the non-zero eigenvalues $\bar{\lambda}_k = n\lambda_k$ of the matrix $\mathbf{K}$, translates into the condition $\bar{\lambda}_k(\boldsymbol{\alpha}^k.\boldsymbol{\alpha}^k) = 1$ [27]. Finally, we can compute the projection of $\Phi(\mathbf{x})$ onto the k-th nonlinear principal component by

$$\beta_k(\mathbf{x}) = (\mathbf{V}^k.\Phi(\mathbf{x})) = \frac{1}{\sqrt{n\lambda_k}} \sum_{i=1}^{n} \alpha_i^k K(\mathbf{x}_i, \mathbf{x}) \ . \tag{8}$$

We used $\bar{\lambda}_k$ to stress the difference between the eigenvalues of the $\mathbf{K}$ matrix and the eigenvalues of (6) to give the connection to the Nyström approximation recently used in [42]. The authors have used the eigenfunction decomposition of the kernel function K to find the orthogonal eigenfunctions spanning the RKHS space $\mathcal{H}$.[3] The Nyström approximation to the k-th eigenfunction is given by

$$\phi_k(\mathbf{x}) = \frac{1}{n\lambda_k} \sum_{i=1}^{n} \phi_k(\mathbf{x}_i) K(\mathbf{x}_i, \mathbf{x}), \tag{9}$$

where the eigenfunctions $\{\phi_k(\mathbf{x}_i)\}_{k=1}^n$ at the points $\{\mathbf{x}_i\}_{i=1}^n$ and corresponding eigenvalues $\{\lambda_k = \bar{\lambda}_k/n\}_{k=1}^n$ are given by the solution of matrix problem (6). Thus, although in the case of infinite dimensional RKHS[4] we can theoretically recover an infinite number of the eigenfunctions with positive eigenvalues in practice we are constrained to the number specified by the number of observations.

In fact, KPCA transforms the problem of the diagonalization of the covariance matrix in the feature space $\mathcal{F}$ to the same matrix approximation of the eigenfunctions decomposition of kernel K as used in [42]; i.e. equation (6). In the case that we have n linearly independent vectors $\{\Phi(\mathbf{x}_i)\}_{i=1}^n$, we can recover the orthonormal basis of eigenvectors with positive eigenvalues spanning a corresponding n-dimensional subspace of a feature space. Then, the eigenvector $\mathbf{V}^k$ can be seen as a feature space representation of the k-th eigenfunction $\phi_k(\mathbf{x})$ (evaluated at the points $\{\mathbf{x}_i\}_{i=1}^n$) "embedded" into a n-dimensional subspace of $\mathcal{F}$ defined by vectors $\{\Phi(\mathbf{x}_i)\}_{i=1}^n$. Moreover, the uniqnuess of the solution of (6) allows us to use notation $\{\phi_k(\mathbf{x}_i) = \alpha_k(\mathbf{x}_i) := \alpha_i^k\}_{k=1}^n$ leading to the equality $\beta_k(\mathbf{x}) = \phi_k(\mathbf{x})\sqrt{n\lambda_k}$.

In fact, it was pointed out in [42] that the projections (8) and (9) are the same up to the scaling factor. In the case of (8), the scaling in all possible

[3] The authors studied the more general eigenfunctions problem $\int K(\mathbf{x}, \mathbf{y}) p(\mathbf{x}) \phi_i(\mathbf{x}) d\mathbf{x} = \lambda_i \phi_i(\mathbf{y})$, where also the probability density $p(\mathbf{x})$ of the input data points was taken into account. This leads to interesting theoretical results about dependence of the eigenspectrum and eigenfunctions on the input data distribution (see [42]). However, for the purposes of our comparison we will not explicitly use this fact as we can still implicitly assume that our data are sampled by some $p(\mathbf{x})$.

[4] Here we define the dimensionality of RKHS by the dimensionality of corresponding feature space $\mathcal{F}$, i.e. by the number of eigenfunctions with a positive eigenvalues spanning the feature space.

eigendirections of the n-dimensional subspace of $\mathcal{F}$ defined by the observed data is the same. On the other hand, the projection (9) scales the data in the individual directions by the factor $\frac{1}{\sqrt{n\lambda_k}}$. This may lead to different results on methods which are not scale invariant. In Section 5 we will show that this is not a case when KPCR is assumed.

At the beginning of the section we assumed that we are dealing with centered data $\Phi(\mathbf{x})$ in a feature space. However, in practical computation, the centralization of the data leads to the modification of (6) to the form [27]

$$n\tilde{\lambda}\tilde{\boldsymbol{\alpha}} = \tilde{\mathbf{K}}\tilde{\boldsymbol{\alpha}}, \tag{10}$$

where the requirement of centered data $\Phi(\mathbf{x})$ was transformed to the change of $\mathbf{K}$ matrix to $\tilde{\mathbf{K}} = \mathbf{K} - \mathbf{1}_n\mathbf{K} - \mathbf{K}\mathbf{1}_n + \mathbf{1}_n\mathbf{K}\mathbf{1}_n$ where $\mathbf{1}_n$ is an $(n \times n)$ matrix of $1/n$ elements. Similarly we have to change the $(n_t \times n)$ "test" matrix $\mathbf{K}^{test}$ whose elements are $\mathrm{K}_{ij}^{test} := K(\mathbf{x}_i, \mathbf{x}_j)$ where $\{\mathbf{x}_i\}_{i=1}^{n_t}$ and $\{\mathbf{x}_j\}_{j=1}^{n}$ are testing and training points, respectively. The centralization of the matrix $\mathbf{K}^{test}$ is given by $\tilde{\mathbf{K}}^{test} = \mathbf{K}^{test} - \mathbf{1}_{n_t}\mathbf{K} - \mathbf{K}^{test}\mathbf{1}_n + \mathbf{1}_{n_t}\mathbf{K}\mathbf{1}_n$ where $\mathbf{1}_{n_t}$ is now an $(n_t \times n)$ matrix with the same entries $1/n$. The eigenfunction decomposition of the $\tilde{\mathbf{K}}$ matrix will now lead to the new eigenfunctions basis $\{\tilde{\phi}_i(\mathbf{x})\}_{i=1}^{n}$ which is only orthogonally rotated compared to the eigenfunctions basis $\{\phi_i(\mathbf{x})\}_{i=1}^{n}$ of the matrix $\mathbf{K}$, however, in general the estimated eigenspectrums will differ.

In the case where we would like to extract only a subset of principal components and/or we are dealing with large data sets, then the direct diagonalization (if it is computationally possible) of the $\mathbf{K}$ matrix can be disadvantageous compared for the estimation of the predefined number of principal components. In the next section we describe a probabilistic PCA model in a feature space using the EM algorithm to find the desired principal subspace.

4 An EM Approach to Kernel PCA

Let us first define a probabilistic PCA model $\mathbf{x} = \mathbf{Q}\mathbf{y} + \mathbf{v}$ in the input space R^N, where $\mathbf{Q}$ is a $R^{N\times p}$ matrix and the observation vector and latent variable vectors are given as $\mathbf{x} \in R^N$ and $\mathbf{y} \in R^p$, respectively. The latent variables are normally distributed with zero mean and identity covariance. The zero mean noise $\mathbf{v}$ is also normally distributed with a covariance matrix defined as $\boldsymbol{\Psi}$. It is shown in [24, 34] that as the noise level in the model becomes infinitesimal the PCA model is recovered. The posterior density then becomes a delta function $P(\mathbf{y}|\mathbf{x}) = \delta(\mathbf{y} - (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{x})$ and the EM algorithm is effectively a straightforward least squares projection [24] which is given below. We denote the matrix of data observations as $\mathbf{X} \in R^{N\times n}$ and the matrix of latent variables as $\mathbf{Y} \in R^{p\times n}$. Then

E-Step $\mathbf{Y} = (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\mathbf{X}$

M-Step $\mathbf{Q}^{new} = \mathbf{X}\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-1}$.

Motivated by this result, in [20][5], we proposed the EM approach to Kernel PCA which is based on the nonlinear mapping of the input data to feature space $\mathcal{F}$ by a map $\Phi : R^N \to \mathcal{F}$.

Again, the centering in the feature space $\mathcal{F}$ can be carried out in a straightforward manner by "centering" the kernel matrix $\mathbf{K}$ outlined in the end of the previous section.

Realizing that the columns of $\mathbf{Q}$ are scaled and rotated eigenvectors computed by diagonalization of the sample covariance matrix we can express the r^{th} column of $\mathbf{Q}$ as $\mathbf{Q}^r = \sum_{j=1}^n \gamma_j^r \Phi(\mathbf{x}_j)$ [34, 27] and write it in matrix notation as $\mathbf{\Phi}^T\mathbf{\Gamma}$, where the matrix $\mathbf{\Phi}$ is the matrix which has individual rows consisting of the vectors $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_n)$ of the mapped observed data and the $\mathbf{\Gamma}$ is an $(n \times p)$ matrix of the coefficients $\{\gamma_i^r : i = 1, \ldots, n; r = 1, \ldots, p\}$. Using the "kernel" trick, i.e. $\Phi(\mathbf{x}_1)^T\Phi(\mathbf{x}_2) = K(\mathbf{x}_1, \mathbf{x}_2)$ we can see that the E-step will now be

$$\mathbf{Y} = (\mathbf{\Gamma}^T\mathbf{K}\mathbf{\Gamma})^{-1}\mathbf{\Gamma}^T\mathbf{K} \,. \tag{11}$$

Now let us consider the M-Step. Denote the term $\mathbf{A} \equiv \mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-1}$. As a consequence of the fact that the columns of $\mathbf{Q}$ lie in the span of $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_n)$ we can consider the set of equations

$$\Phi^T(\mathbf{x}_j)\mathbf{Q}^{new} = \Phi^T(\mathbf{x}_j)\mathbf{\Phi}^T\mathbf{A} \;\text{ for all }\; j = 1, \ldots, n$$

which can be also written as

$$\left[\sum_{i=1}^n \gamma_i^1 K(\mathbf{x}_j, \mathbf{x}_i), \ldots, \sum_{i=1}^n \gamma_i^k K(\mathbf{x}_j, \mathbf{x}_i)\right] = [K(\mathbf{x}_j, \mathbf{x}_1), \ldots, K(\mathbf{x}_j, \mathbf{y}_n)]\,\mathbf{A}$$

for all $j = 1, \ldots, n$. We can write it in matrix form $\mathbf{K}\mathbf{\Gamma}^{new} = \mathbf{K}\mathbf{A}$[6]. The M-step then follows as

$$\mathbf{\Gamma}^{new} = \mathbf{A} = \mathbf{Y}^\mathbf{T}(\mathbf{Y}\mathbf{Y}^\mathbf{T})^{-\mathbf{1}} \,. \tag{12}$$

It has been shown in [34], that in the case of infinitesimal noise in our model, i.e. $\mathbf{\Psi} = \lim_{\sigma^2 \to 0} \sigma^2\mathbf{I}$, the matrix $\mathbf{Q}$ at convergence will be equal to $\mathbf{Q} = \mathbf{V}\mathbf{\Lambda}^{1/2}\mathbf{R}$, where the columns of the $\mathbf{V}$ matrix are the eigenvectors of the sample covariance matrix with corresponding eigenvalues $\lambda_1, ..., \lambda_p$ creating the diagonal matrix $\mathbf{\Lambda}$, and $\mathbf{R}$ is an arbitrary orthogonal rotation matrix. In [34] the authors also pointed out that taking the columns of $\mathbf{R}^T$ to be equal to the eigenvectors of the $\mathbf{Q}^T\mathbf{Q}$ matrix we can recover the true principal axes. Thus, in our case the

[5] For the clarity with the other sections we switched the $\mathbf{x}$ and $\mathbf{y}$ notation comparing to the original paper and also used $\mathbf{Q}$ matrix notation instead of $\mathbf{C}$.

[6] In general, $\mathbf{K}$ is a positive semidefinite matrix, to get the unique solution we need to guarantee the linear independence of the vectors $\{\mathbf{\Phi}(\mathbf{x}_i)\}_{i=1}^n$, i.e. positive definiteness of $\mathbf{K}$, by appropriate selection of the training data set $\{\mathbf{x}_i\}_{i=1}^n$. In practice we can do this by adding arbitrary small positive values on the diagonal. In the case that $M < n$ we are getting one of the possible solutions for $\mathbf{\Gamma}$.

projection of the test point $\mathbf{x}$ onto the p nonlinear principal components is now given by

$$\beta(\mathbf{x}) := \mathbf{R}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\Phi(\mathbf{x}) = \mathbf{R}(\mathbf{\Gamma}^T\mathbf{K}\mathbf{\Gamma})^{-1}\mathbf{\Gamma}^T\mathbf{K}^{test}\ , \tag{13}$$

where $\mathbf{K}^{test}$ is the vector $[K(\mathbf{x}_1,\mathbf{x}),...,K(\mathbf{x}_n,\mathbf{x})]^T$. Moreover, in Appendix A we have shown how the estimation of eigenvalues $\{\lambda_i\}_{i=1}^p$ and consequently the normalization of the projection to avoid the different scaling in the individual eigendirections can be achieved.

We should note a number of points regarding this method for performing Kernel PCA. Firstly, due to the use of the Mercer kernels the method is independent of the dimensionality of the input space. Secondly, the computational complexity, per iteration, of the proposed EM method for Kernel PCA is $\mathcal{O}(pn^2)$ where n is the number of data points and p is the number of extracted components. Where a small number of eigenvectors require to be extracted and there are a large number of data points available this method is comparable in complexity to the iterative power method which has complexity $\mathcal{O}(n^2)$. Direct diagonalization of a symmetric $\mathbf{K}$ matrix to solve the eigenvalue problem for Kernel PCA [27] has complexity of the order $\mathcal{O}(n^3)$. From the equations (11) and (12) we can also see, that individual EM steps can be performed without storing whole $(n \times n)$ matrix $\mathbf{K}$. In such a case memory requirements scale as $\mathcal{O}(p^2) + \mathcal{O}((p+1)n)$ (in Appendix B possible implementations of the algorithm are discussed). However, this will slow down the computations as the elements of $\mathbf{K}$ have to be computed repeatedly. The balance between the speed and memory requirements can be achieved by allocating another $(p \times n)$ matrix (see Appendix B).

5 Feature Space Regularized Linear Regression

5.1 Kernel Principal Component Regression

Consider the standard regression model in feature space $\mathcal{F}$

$$\mathbf{y} = \mathbf{\Phi}\boldsymbol{\gamma} + \boldsymbol{\epsilon}, \tag{14}$$

where $\mathbf{y}$ is a vector of n observations of the dependent variable, $\mathbf{\Phi}$ is an $(n \times M)$ matrix of regressors whose i-th row is again the vector $\Phi(\mathbf{x}_i)$ of the mapped $\mathbf{x}_i$ observation into $M \leq \infty$ dimensional feature space $\mathcal{F}$, $\boldsymbol{\gamma}$ is a vector of regression coefficients and $\boldsymbol{\epsilon}$ is the vector of error terms whose elements have equal variance σ^2, and are independent of each other. We also assume that regressors $\{\Phi_j(\mathbf{x})\}_{j=1}^M$ are zero-mean. Thus $\mathbf{\Phi}^T\mathbf{\Phi}$ is proportional to the sample covariance matrix and Kernel PCA can be performed to extract M eigenvalues $\{\lambda_j\}_{j=1}^M$ and corresponding eigenvectors $\{\mathbf{V}^j\}_{j=1}^M$. The projection of the $\Phi(\mathbf{x})$ onto the k-th nonlinear principal component is given by (8). By projection of

all original regressors onto the principal components we can rewrite (14) as

$$\mathbf{y} = \mathbf{B}\mathbf{w} + \boldsymbol{\epsilon}, \tag{15}$$

where $\mathbf{B} = \mathbf{\Phi}\mathbf{V}$ is now an $(n \times M)$ matrix of transformed regressors and $\mathbf{V}$ is an $(M \times M)$ matrix whose k-th column is the eigenvector $\mathbf{V}^k$. The columns of the matrix $\mathbf{B}$ are now orthogonal and the least squares estimate of the coefficients $\mathbf{w}$ becomes

$$\hat{\mathbf{w}} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{y} = \mathbf{\Lambda}^{-1}\mathbf{B}^T\mathbf{y}, \tag{16}$$

where $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_M)$. The results obtained using all principal components for the projection of the original regressor variables for (15) is equivalent to that obtained by least squares using the original regressors.[7]

In fact we can express the estimate $\hat{\boldsymbol{\gamma}}$ of the original model (14) as

$$\hat{\boldsymbol{\gamma}} = \mathbf{V}\hat{\mathbf{w}} = \mathbf{V}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{y} = \sum_{i=1}^{M} \lambda_i^{-1}\mathbf{V}^i(\mathbf{V}^i)^T\mathbf{\Phi}^T\mathbf{y}$$

and its corresponding variance-covariance matrix [10] as

$$cov(\hat{\boldsymbol{\gamma}}) = \sigma^2\mathbf{V}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{V}^T = \sigma^2\mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{V}^T = \sigma^2\sum_{i=1}^{M}\lambda_i^{-1}\mathbf{V}^i(\mathbf{V}^i)^T \tag{17}$$

To avoid the problem of multicollinearity PCR uses only some of the principal components. It is clear from (17) that the influence of small eigenvalues can significantly increase the overall variance of the estimate. PCR simply deletes the principal components corresponding to small values of the eigenvalues λ_i, i.e. the principal components where multicollinearity may appear. The penalty we have to pay for the decrease in variance of the regression coefficient estimate is bias in the final estimate. However, if multicollinearity is a serious problem, the introduced bias can have a less significant effect in comparison to a high variance estimate. If the elements of $\mathbf{w}$ corresponding to deleted regressors are zero, an unbiased estimate is achieved [10].

Using the first p-nonlinear principal components (8) to create a linear model based on orthogonal regressors in feature space $\mathcal{F}$ we can formulate KPCR model as

$$f(\mathbf{x}, \mathbf{c}) = \sum_{k=1}^{p} w_k\beta(\mathbf{x})_k + b = \sum_{k=1}^{p} w_k \sum_{i=1}^{n} \alpha_i^k K(\mathbf{x}_i, \mathbf{x}) + b = \sum_{i=1}^{n} c_i K(\mathbf{x}_i, \mathbf{x}) + b, \tag{18}$$

[7] PCR, as well as other biased regression techniques, is not invariant to the relative scaling of the original regressors [6]. However, similar to ordinary least squares regression the solution of (15) does not depend on a possibly different scaling in individual eigendirections used in KPCA transformation. The solution is also invariant to the orthogonal rotation and in the case of EM approach to KPCA we do not need to use rotation given by the matrix $\mathbf{R}$ (13).

where $\{c_i = \sum_{k=1}^{p} w_k \alpha_i^k\}_{i=1}^{n}$.

We have shown that by removing the principal components whose variances are very small we can eliminate large variances of the estimate due to multicollinearities. However, if the orthogonal regressors corresponding to those principal components have a large correlation with the dependent variable y such deletion is undesirable (experimentally demonstrated in [22]). There are several different strategies for selecting the appropriate orthogonal regressors for the final model (see [10, 11] and ref. therein). In [22] we considered the Covariance Inflation Criterion [33] for model selection in KPCR as a novel alternative to methods such as cross-validation.

5.2 Kernel Ridge Regression

KRR is another technique to deal with multicollinearity by assuming the linear regression model (14) whose solution is now achieved by minimizing

$$R_{rr}(\boldsymbol{\gamma}) = \sum_{i=1}^{n} [y_i - f(\mathbf{x}_i, \boldsymbol{\gamma})]^2 + \xi \|\boldsymbol{\gamma}\|^2, \tag{19}$$

where $f(\mathbf{x}, \boldsymbol{\gamma}) = \boldsymbol{\gamma}^T \Phi(\mathbf{x}) + b$ and ξ is a regularization term. The least-squares estimate of $\boldsymbol{\gamma}$ is biased but the variance is decreased. Similar to the KPCR case we can express the variance-covariance matrix of the $\boldsymbol{\gamma}$ estimate [10] as

$$cov(\hat{\boldsymbol{\gamma}}) = \sigma^2 \sum_{i=1}^{M} \lambda_i (\lambda_i + \xi)^{-2} \mathbf{V}^i (\mathbf{V}^i)^T.$$

We can see, that in contrast to KPCR, the variance reduction in KRR is achieved by giving less weight to small eigenvalue principal components via the factor ξ.

In practice we usually do not know the explicit mapping $\Phi(.)$ or its computation in the high-dimensional feature space $\mathcal{F}$ may be numerically intractable. In [25], using the dual representation of the linear RR model the authors derived the formula for estimation of the weights $\boldsymbol{\gamma}$ for the linear RR model $y = \boldsymbol{\gamma}^T \Phi(\mathbf{x})$ in feature space $\mathcal{F}$; i.e. (non-linear) KRR. Again, using the fact that $K(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x})^T \Phi(\mathbf{y})$ we can express the final KRR model in the dot product form [25, 3]

$$f(\mathbf{x}) = \mathbf{c}^T \mathbf{k} = \mathbf{y}^T (\mathbf{K} + \xi I)^{-1} \mathbf{k}\,, \tag{20}$$

where $\mathbf{K}$ is again an $(n \times n)$ Gram matrix consisting of dot products $\mathrm{K}_{ij} = (\Phi(\mathbf{x}_i).\Phi(\mathbf{x}_j))$ $i, j = 1, \ldots, n$; $\mathbf{k}$ is the vector of dot products of a new mapped input example $\Phi(\mathbf{x})$ and the vectors of the training set; $\mathrm{k}_i = (\Phi(\mathbf{x}_i).\Phi(\mathbf{x}))$. It is worth noting that the same solution to the RR problem in the feature space $\mathcal{F}$ can also be derived based on the dual representation of the Regularization Networks minimizing the cost function (1) using the quadratic loss function

$V(y_i, f(\mathbf{x}_i)) = (y_i - f(\mathbf{x}_i))^2$ (see e.g. [9]) or through the techniques derived from Gaussian processes [41, 3].

We can see that including a possible bias term into the model leads to its penalization through the ξ term. However, in the case of regression or classification tasks there is no reason to penalize the shift of $f(.)$ by a constant. It was pointed out in [5], that in the case of a radial kernel we can overcome this by using a new kernel of the form

$$\hat{K}(\mathbf{x}, \mathbf{y}) = K(\mathbf{x}, \mathbf{y}) - \xi_0.$$

The ξ_0 is chosen to construct a new RKHS consisting only of zero-mean functions; i.e. $\hat{K}$ without the zeroth order Fourier component. Effectively, the new kernel $\hat{K}$ induces the null space of the constant functions which are not included in a new RKHS norm and based on the cost function (1) are not penalized[8]. Now, the solution (4) will take the form [9, 39, 5]

$$f(\mathbf{x}) = \sum_{i=1}^{n} c_i \hat{K}(\mathbf{x}, \mathbf{x}_i) + \hat{b} = \sum_{i=1}^{n} c_i (K(\mathbf{x}, \mathbf{x}_i) - \xi_0) + \hat{b} = \sum_{i=1}^{n} c_i K(\mathbf{x}, \mathbf{x}_i) + b \quad (21)$$

and the unknown coefficients $\{c_i\}_{i=1}^n, b = \sum_{i=1}^n c_i \xi_0 + \hat{b}$ can be found by solving the following system of linear equations [9, 5]

$$(\hat{\mathbf{K}} + \xi I)\mathbf{c} + \mathbf{1}b = (\mathbf{K} + (\xi - \xi_0)I)\mathbf{c} + \mathbf{1}b = (\mathbf{K} + \xi_{new} I)\mathbf{c} + \mathbf{1}b = \mathbf{y},$$

$$\sum_{i=1}^{n} c_i = 0. \quad (22)$$

Thus we still can use a positive definite kernel K as the only change is to estimate new b and ξ_{new} terms. Let us note that the solution of the SVR, i.e. assuming the linear regression model $y = \boldsymbol{\gamma}^T \Phi(\mathbf{x}) + b$ in the feature space $\mathcal{F}$, leads to the non-linear regression model (21). In fact, in [32] the authors have shown that using the quadratic loss function in the case of SVR model transforms the general quadratic optimization problem [29] for finding the estimate of the weights $\boldsymbol{\gamma} = \sum_{i=1}^n c_i \Phi(\mathbf{x}_i)$ and b to the solution of the linear equations (22).

Another technique in removing a "bias" term problem is to "centralize" the regression problem in feature space; i.e. assume the sample mean of the mapped data $\tilde{\Phi}(\mathbf{x}_i)$ and targets $\tilde{y}$ to be zero. This will lead to the regression problem $\tilde{y} = \tilde{\boldsymbol{\gamma}}^T \tilde{\Phi}(\mathbf{x})$ without the bias term. The centralization of the individual mapped data points $\Phi(\mathbf{x})$ can be done by the same "centralization" of the Gram matrix $\mathbf{K}$ and vector $\mathbf{k}$ as described in Section 3. The solution is then given by modification of (20) to the form

$$\tilde{f}(\mathbf{x}) = \tilde{\mathbf{y}}^T (\tilde{\mathbf{K}} + \xi I)^{-1} \tilde{\mathbf{k}} \,. \quad (23)$$

[8] In fact, we do not need to constrain ourself to the construction of a RKHS with only constant functions not included in the norm. Similar to the SVR, we can construct a new kernel $\hat{K}$ and by using the same stabilizer $\|f\|^2_{\mathcal{H}}$ we can "balance" the penalization of the potential constant feature of a initial kernel K by a new extra, not penalized, constant feature.

5.3 Summing Up

Using the analogy with PCR and RR in input data space, a connection between regularized linear regression models in feature space $\mathcal{F}$ corresponding to KPCR and KRR has been established. Both methods belong to the class of shrinkage estimators; i.e. they shrink the ordinary least squares solution from the directions of low data spread to directions of larger data spread. This effectively means that we can achieve the lower variance of estimated regression coefficients at the cost of a biased estimate. Whilst with KPCR we project the data mainly to the principal components corresponding to larger eigenvalues, with KRR we are giving less weight to the smaller eigenvalues. Thus, in both cases we are faced with a model selection problem; i.e. selection of non-linear principal components in KPCR and setting the regularization term ξ in KRR, respectively. In KPCR one of the straightforward model selection criteria is based on choosing the first p principal components describing the predefined amount of overall variance. If $p \ll n \leq M$, then using the proposed EMKPCA method having complexity $\mathcal{O}(pn^2)$ can be highly advantageous. Moreover, on large scale regression problems, where we can not diagonalize the Gram matrix $\mathbf{K}$, the estimate of the $p \ll n \leq M$ principal components seems to be the only possible choice. However, in Section 7 we will demonstrate that in some cases the principal components corresponding to relatively small eigenvalues can have a significant contribution to the performance of KPCR. In such cases, if it is computationally possible, the extraction of almost the whole spectra of principal components and using the appropriate model selection criteria can lead to improvement of KPCR. On the other hand, in the case of KRR the computation complexity scales with $\mathcal{O}(n^3)$ and in general we also have to do the search through a wide range of possible values to find the optimal regularization term ξ . Moreover, on large data set we can not use the direct method; i.e. inversion of the $(n \times n)$ matrix $\mathbf{K}$, and we need to do some approximations as proposed in [7, 35, 30].

6 Data Sample Construction

6.1 Chaotic Mackey-Glass Time-Series

The chaotic Mackey-Glass time-series is defined by the differential equation

$$\frac{ds(t)}{dt} = -bs(t) + a\frac{s(t-\tau)}{1+s(t-\tau)^{10}}$$

with $a = 0.2$, $b = 0.1$. The data were generated with $\tau = 17$ and using a second-order Runge-Kutta method with a step size 0.1. Training data is from t=200 to t=3200 while test data is in the range t= 5000 to 5500. To this generated time-series we added noise with normal distribution and with different levels

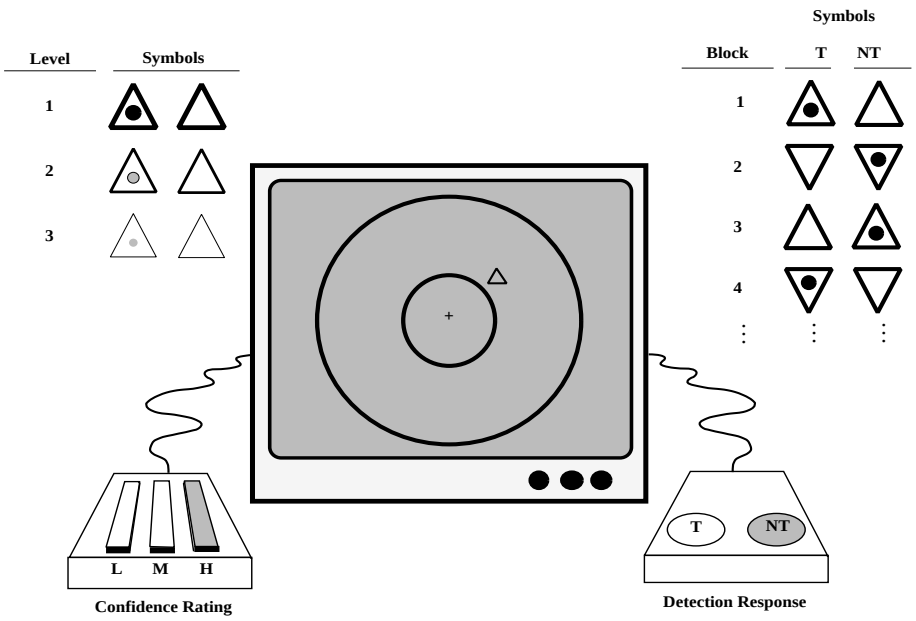

Figure 1: Display, input device configuration and symbols for task-relevant stimuli for the signal detection task.

corresponding to ratios of the standard deviation of the noise and the clean Mackey-Glass time-series.

6.2 Human Signal Detection Performance Monitoring

We have used Event Related Potentials (ERPs) and performance data from an earlier study [37, 36, 14]. Eight male Navy technicians experienced in the operation of display systems performed a signal detection task. Each technician was trained to a stable level of performance and tested in multiple blocks of 50–72 trials each on two separate days. Blocks were separated by 1-minute rest intervals. A set of 1000 trials were performed by each subject. Inter-trial intervals were of random duration with a mean of 3s and a range of 2.5–3.5s. The entire experiment was computer-controlled and performed with a 19-inch color CRT display (Figure 1). Triangular symbols subtending 42 minutes of arc and of three different luminance contrasts (0.17, 0.43, or 0.53) were presented parafoveally at a constant eccentricity of 2 degrees visual angle. One symbol was designated as the target, the other as the non-target. On some blocks, targets contained a central dot whereas the non-targets did not. However, the association of symbols to targets was alternated between blocks to prevent the development of automatic processing. A single symbol was presented per trial,

at a randomly selected position on a 2-degree annulus. Fixation was monitored with an infrared eye tracking device. Subjects were required to classify the symbols as targets or non-targets using button presses and then to indicate their subjective confidence on a 3-point scale using a 3-button mouse. Performance was measured as a linear composite of speed, accuracy, and confidence. A single measure, PF1, was derived using factor analysis of the performance data for all subjects, and validated within subjects. The computational formula for PF1 was

$$\text{PF1} = 0.33*\text{Accuracy} + 0.53*\text{Confidence} - 0.51*\text{Reaction Time}$$

using standard scores for accuracy, confidence, and reaction time based on the mean and variance of their distributions across all subjects. PF1 varied continuously, being high for fast, accurate, and confident responses and low for slow, inaccurate, and unconfident responses. In our experiments we linearly normalized PF1 to have a range of 0 to 1.

ERPs were recorded from midline frontal, central, and parietal electrodes (Fz, Cz, and Pz), referred to average mastoids, filtered digitally to a bandpass of 0.1 to 25 Hz, and decimated to a final sampling rate of 50 Hz. The prestimulus baseline (200 ms) was adjusted to zero to remove any DC offset. Vertical and horizontal electrooculograms (EOG) were also recorded. Epochs containing artifacts were rejected and EOG-contaminated epochs were corrected. Furthermore, any trial in which no detection response or confidence rating was made by a subject was excluded along with the corresponding ERP.

Within each block of trials, a running-mean ERP was computed for each trial (Figure 2). Each running-mean ERP was the average of the ERPs over a window that included the current trial plus the 9 preceding trials for a maximum of 10 trials per average. Within this 10-trial window, a minimum of 7 artifact-free ERPs were required to compute the running-mean ERP. If fewer than 7 were available, the running mean for that trial was excluded. Thus each running mean was based on at least 7 but no more than 10 artifact-free ERPs. This 10-trial window corresponds to about 30s of task time. The PF1 scores for each trial were also averaged using the same running-mean window applied to the ERPs, excluding PF1 scores for trials in which ERPs were rejected. Prior to analysis, the running-mean ERPs were clipped to extend from time zero (stimulus onset time) to 1500 ms post-stimulus, for a total of 75 time points.

7 Results

The present work was carried out with Gaussian kernels; $K(\mathbf{x}, \mathbf{y}) = e^{-(\frac{\|\mathbf{x}-\mathbf{y}\|^2}{L})}$, where L determines the width of the Gaussian function. The Gaussian kernel possesses a good smoothness properties (suppression of the higher frequency components) and in the case we do not have a priori knowledge about the regression problem we would prefer a smooth estimate [9, 31].

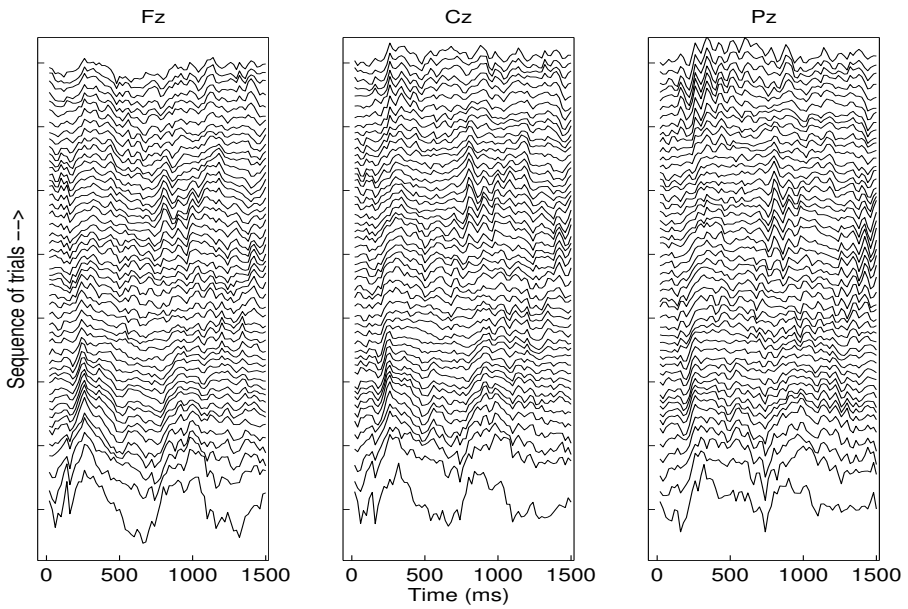

Figure 2: Running-mean ERPs at sites Fz, Cz and Pz for subject B in the first 50 running-mean ERPs.

7.1 Chaotic Mackey-Glass Time-Series

The standard EM convergence proofs [4] and the results on the convergence of EM for PCA [34] guarantee that after reaching a stable local extremum (which is the global maximum) the true principal subspace will be recovered. However, for our purposes we would like to experimentally compare the performance of the Kernel Principal Component Regression (KPCR) with EM for Kernel PCA (EMKPCA) with the case when the main principal components are extracted by KPCA. It also seems to be meaningful to investigate how many EM steps we need, or in other words how fast the EMKPCA algorithm is to get the desired performance of KPCR. To compare the performance of EMKPCA with the other iterative methods for estimation of only a few main principal components we also used the `eigs` Matlab procedure. This is an implementation of a implicitly restarted Arnoldi method [19, 40] for computing a few selected eigenvalues of large structured matrices. We refer to this method as iKPCR.

All regression models were trained to predict the value at time $t + 85$ from inputs at time $t, t - 6, t - 12, t - 18$. The training data partitions were constructed by moving a "sliding window" over the 3000 training samples in steps of 500 samples. This window had two sizes - 500 samples and 1000 samples, respectively. This created six partitions of size 500 samples and five partitions of size 1000 samples. We estimated the variance of the overall clean training

set and based on this estimate $\hat{\sigma}^2 \doteq 0.05$ we repeated our simulations for the width L from the range $\langle 0.2\hat{\sigma}^2, 20\hat{\sigma}^2 \rangle$ using the step size 0.01. A fixed test set of size 500 data points (see Section 6.1) was used in all experiments. The performance of the regression models to predict "clean" Mackey-Glass time series was evaluated in terms of normalized root mean squared error (NRMSE). The stopping criterion in the case of EMKPCA was the change of the $\mathbf{\Gamma}$ matrix in two consecutive EM steps - in reported experiments we used the Frobenius norm and the threshold equal 1.

The best results on test set averaged over all individual runs are summarized in Table 1. We can not observe any significant difference between the performance of the methods. With the iKPCR method we did not achieve convergence on all training data pairs on tasks to compute more than 500 eigenvectors (several different stopping criteria and initializations were used). We hypothesize that this is due to the low rate between small eigenvalues $\frac{\lambda_{i+1}}{\lambda_i} \rightarrow 0$ of the $\mathbf{K}$ matrix .

In Figure 3 we depicted the number of EM steps (in dependence on number of extracted principal components) for which the number of floating point operations computed by Matlab's `flops` function was less than direct diagonalization of the Gram matrix $\mathbf{K}$. Individual plots represent the dependence for different widths L of the Gaussian kernel used in experiments reported in Table 1. From both graphs we can see that even for relatively small training data sets (500 (left) and 1000 (right)) using EMKPCA can be advantageous when the number of desired principal components scales up to 2-3 hundred. With further significantly greater increases of the number of training data points comparing to the number of desired principal components, the use of EMKPCA will be even more profitable. However, we have to note, that in the case of EMKPCA we did not included the computational costs associated with finding the corresponding eigenvalues (Appendix A). This is important when the estimate of the proportion of the variance described by the extracted principal components is needed. We observed, that adding these extra computations and using 15 EM steps, the EMKPCA approach will still be profitable when extraction of up to 150 principal components is desired. Similar to Figure 3, in Figure 4 we depicted the number of EM steps for which the number of floating point operations was less than using Matlab's `eigs` iterative method with default convergence tolerance $1e^{-10}$. Although, theoretically, the complexity of the EMKPCA and `eigs` methods is comparable only for the extraction of a lower number of principal components we can see that in terms of a number of floating points operations the EMKPCA may be advantageous also for the extraction of a relatively high number of principal components. We also observed that with increasing width of the Gaussian kernel the number of principal components on which `eigs` converged was decreased. In the case of the Gaussian kernel the eigenvalues decay more rapidly with increasing width and we may conjecture that this would be the reason why the convergence for higher number of eigenvalues was not

Method	*n/s=0.0%*		*n/s=11%*		*n/s=22%*	
	500	1000	500	1000	500	1000
KPCR *with*	0.038	0.008	0.308	0.281	0.442	0.413
EMKPCA	(0.025)	(0.004)	(0.028)	(0.003)	(0.033)	(0.008)
# of EM steps	5.7	22	10.8	13.4	8.2	11.6
KPCR *with*	0.038	0.008	0.307	0.280	0.443	0.414
KPCA	(0.025)	(0.004)	(0.030)	(0.003)	(0.033)	(0.010)
KPCR *with*	0.038	-	0.310	0.280	0.448	0.414
iKPCR	(0.025)	-	(0.033)	(0.004)	(0.034)	(0.010)

Table 1: The comparison of the approximation errors (NRMSE) of prediction for 2 different sizes of Mackey-Glass training set. The values represent an average of 6 simulations in the case of 500 training points and 5 simulations in the case of 1000 training points, respectively. iKPCR represents the method where only the first N eigenvectors were estimated (`eigs` Matlab procedure). Corresponding standard deviation is presented in parentheses. The last row of KPCR with the EM approach to the KPCA method represents the average number of EM steps we used. *n/s* represents the ratio between the standard deviation of the added Gaussian noise and the underlying time-series. For KPCR computed on 500 training points we used the first $N = 495, 100$ and 50 nonlinear principal components corresponding to the case of *n/s=0.0%*, *n/s=11%* and *n/s=22%*, respectively. For KPCR computed on 1000 training points we used the first $N = 750, 125$ and 75 nonlinear principal components.

reached.

In the KPCA algorithm we need to solve the eigenvalue problem (6). However, direct diagonalization of the Gram $\mathbf{K}$ matrix can be numerically unstable when we are dealing with a matrix of high dimensionality. Moreover, in [23] it was shown that on some data sets the estimation of the eigenvectors and eigenvalues using only a fraction of training data points followed by projection of the remaining training data points onto the nonlinear principal components does not degrade overall performance. In Table 2, we demonstrate that fact by using the first half of the training data points (500) to estimate the eigenvectors and eigenvalues of the (500×500) Gram $\mathbf{K}$ matrix. However, it is worth noting that this reduction does not lead to a significant degradation of the performance on noisy data. The best performance on noisy data was achieved using a significantly smaller number of the principal components which can be seen as a noise reduction on the feature space data representation [23]. The significant difference between the prediction accuracy on the clean and on the noisy Mackey-Glass time series gives rise to the question of whether it is at all possible to sufficiently reduce the level of the noise in feature space due to the violation of the additive and uncorrelated essence of the noise introduced by the nonlinear transformation. This may potentially have a stronger effect on the

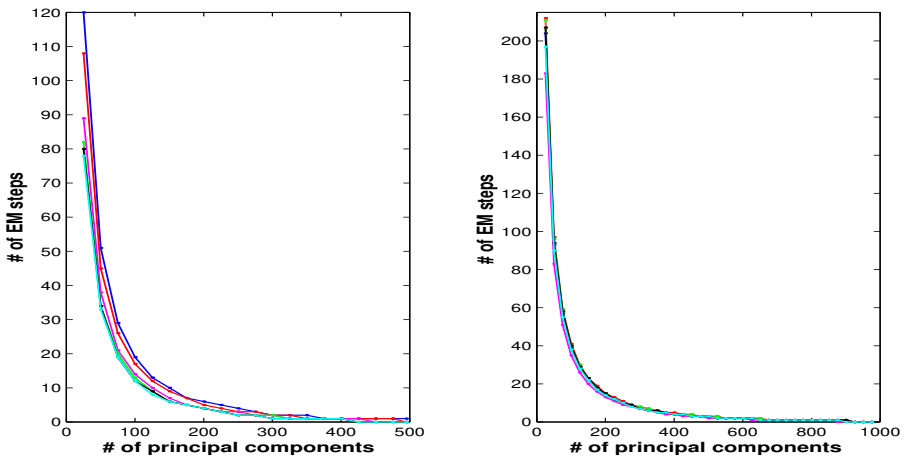

Figure 3: Number of EM steps (in dependence on number of extracted principal components) for which the number of floating points operations computed by Matlab's `flops` function was less than direct diagonalization of the Gram matrix $\mathbf{K}$. Individual plots represent the dependence for different widths L of the Gaussian kernel used in experiments reported in Table 1. *Left*: results using 500 training data points *Right*: results using 1000 training data points.

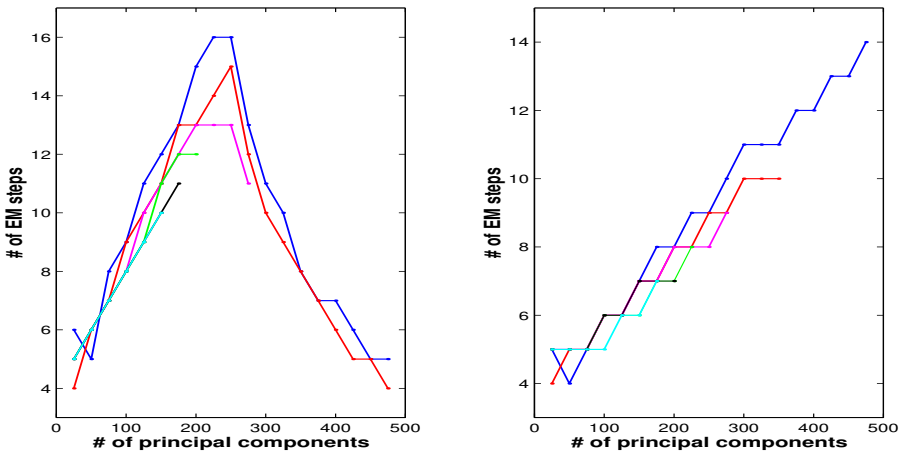

Figure 4: Number of EM steps (in dependence on number of extracted principal components) for which the number of floating points operations was less than using Matlab's `eigs` iterative method with default convergence tolerance $1e^{-10}$. Individual plots represent the dependence for different widths L of the Gaussian kernel used in experiments reported in Table 1. *Left*: results using 500 training data points *Right*: results using 1000 training data points.

main principal components [23]. Therefore, we have to deal with the trade off between noise reduction and the associated signal information loss. From Table 2 we can also see that the reduction of the number of used eigenvectors to 495 in the case of the clean Mackey-Glass leads to a significant decrease of the overall performance (NRMSE 0.014) compared to the results in Table 1 where the best performance was achieved using 750 eigenvectors (NRMSE 0.008).

In next step, we compared KPCR using the EMKPCA algorithm with the KRR method. The same prediction task, training-testing data pairs and selection of width L parameter as described above was used. The regularization parameter ξ in KRR was estimated by cross-validation using 20% of the training data partitions for validation set. In fact, to find the value of ξ, we did the cross-validation in two steps. First the order of ξ was estimated and then the finer structure of the values in range ± 1 order was taken to estimate a "optimal" value of ξ. Table 3 summarize the achieved results. We compare the behavior of KRR both with and without additional bias term, whose solution is given by equation (20) and equation (22), respectively. We can see that assuming an extra, unpenalized bias term we can improve the performance of KRR on this data set. It is also worth noting that there is no significant difference between KRR with bias and the KPCR method. However, in [23], for a wide range of the L parameter we reported smaller variance of KPCR over individual runs for some of the experimental settings. Moreover, in the case of KPCR we averaged the results on different training parts using a fixed number of principal components for each run. This is hardly optimal and we may hypothesize that by using the appropriate model selection criteria for the selection of the "best" eigenvectors we can further improve the performance of the KPCR algorithm (see e.g. [10, 22] and references therein).

On this data set we also experimentally compared KRR with a bias term (22) with "centralized" KRR defined by (23). Using the same strategy for estimation of a ξ penalization term by cross-validation we observed the maximum difference between NRMSE on test set for individual training sets of order $10e^{-6}$. Effectively this leads to the same averaged NRMSE of both methods as displayed for KRR with bias term (22) in Table 3.

Now, let us return to the KPCR model again. We observed that in the case of noisy Mackey-Glass time series we could significantly reduce the number of principal components used in KPCR. Moreover, we observed that using a smaller input data set the subspace defined by these principal components can be sufficiently precisely recovered (evaluated in terms of the performance of KPCR). In such a case we can highly profit from the computational and storage advantages of the EM for KPCA algorithm. However, in the case of the clean Mackey-Glass time series the situation seems to be rather more complicated. Using the 5 different training data sets we observed that on average the best performance of KPCR was achieved using the 750 main principal components. Observing the feature space eigen-spectra corresponding to individual mapped training data sets we found that on average 99% of the variance can be recovered

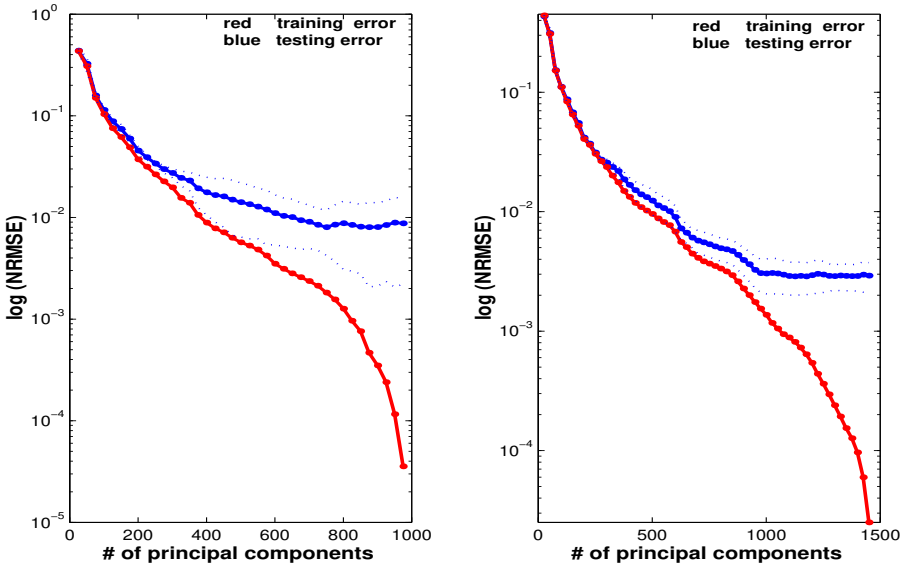

Figure 5: Dependence of the training and testing error of KPCR based on the number of principal components used. Standard errors on test set are presented by dotted lines. *Left*: the average results on five 1000 data points training sets. *Right*: the average results on seven 1500 data points training sets. The data partitions were constructed by "sliding" over a training set consisting of 3000 data points using the step 500 (left) and 250 (right), respectively.

by the first 234±17 principal components. This indicates a rather fast decay of the eigenvalues. In fact, the first 750 principal components are on average covering more then 99.999% of the variance. This effectively means that there also exists principal components corresponding to the small eigenvalues with not negligible importance to the final regression model. In Figure 5 (left) we depict the prediction error in terms of NRMSE in dependence on the number of principal components used. Using 250 principal components (on average covering more then 99% of the variance) will result with 4.24 times increased NRMSE on test set comparing the case when 750 principal components were used. We hypothesize, that there exist a "break point" when another increase of the input data points will not lead to the increase of the "optimal" number of principal components. However, our simulation with the training data sets of size 1500 (Figure 5 (right)) indicate that such a data representation is still not enough (the number of "optimal" principal components was increased to 1125 with corresponding improvement of the performance - 0.003 NRMSE on test set). Thus, it remains the topic of the further simulations to find the optimal subspace of feature space $\mathcal{F}$ representing Mackey-Glass time series.

Method	*n/s=0.0%*	*n/s=11%*	*n/s=22%*
KPCR_{1000}	0.014 (0.005)	0.280 (0.003)	0.414 (0.010)
KPCR_{500}	0.017 (0.009)	0.282 (0.005)	0.414 (0.008)

Table 2: The comparison of the approximation errors (NRMSE) of the KPCR method using all 1000 training data points (KPCR_{1000}) to estimate eigenvectors and eigenvalues with the KPCR method where the first half (500) of the training points was used KPCR_{500}. In the later case, the rest of the training points was projected onto the estimated eigenvectors. The values represent an average of 5 simulations. Corresponding standard deviation is presented in parentheses. *n/s* represents the ratio between the standard deviation of the added Gaussian noise and the underlying time-series. We used the first 495, 125 and 75 nonlinear principal components corresponding to the case of *n/s=0.0%*, *n/s=11%* and *n/s=22%*, respectively.

Method	*n/s=0.0%*		*n/s=11%*		*n/s=22%*	
	500	1000	500	1000	500	1000
KPCR *with EMKPCA*	0.038 (0.025)	0.008 (0.004)	0.308 (0.028)	0.281 (0.003)	0.442 (0.033)	0.413 (0.008)
KRR *with bias eq. (22)*	0.038 (0.024)	0.007 (0.003)	0.312 (0.032)	0.279 (0.010)	0.446 (0.036)	0.404 (0.006)
KRR *eq. (20)*	0.057 (0.039)	0.0102 (0.004)	0.343 (0.034)	0.294 (0.011)	0.468 (0.036)	0.423 (0.018)

Table 3: The comparison of the approximation errors (NRMSE) of prediction for 2 different sizes of Mackey-Glass training set. The values represent an average of 6 simulations in the case of 500 training points and 5 simulations in the case of 1000 training points, respectively. Corresponding standard deviation is presented in parentheses. *n/s* represents the ratio between the standard deviation of the added Gaussian noise and the underlying time-series. For KPCR computed on 500 training points we used the first 495, 100 and 50 nonlinear principal components corresponding to the case of *n/s=0.0%*, *n/s=11%* and *n/s=22%*, respectively. For KPCR computed on 1000 training points we used the first 750, 125 and 75 nonlinear principal components.

7.2 Human Signal Detection Performance Monitoring

On this data set we again investigated the behavior of the KPCR model with the principal components extracted by the EM approach. On this real world data set we do not have any information about the type of the noise. It is well known that in the case of the additive Gaussian noise the best approximation to the regression is given by the quadratic cost function. However, if the noise is uniform alike the Vapnik ϵ-insensitive cost function can lead to more robust estimate [38]. For that reason we also used SVR with the ϵ-insensitive cost function whose solution we found by using *SVMTorch* [2] algorithm.

First, we randomly selected two subjects from all eight subject data set. Based on the results of our former study we have used the first 90% of the principal components. We trained the models on 50% of the ERPs and tested on the remaining data. Using 20% of the training data set as a validation set, we estimated the regularization ξ and ϵ parameters for KRR and SVR models. The described results, for each setting of the parameters, are an average of 10 runs each on a different partition of training and testing data. To be consistent with the previous results reported in [37] the validity of the models was measured in terms of normalized mean squared error (NMSE) and in terms of the proportion of data for which PF1 was correctly predicted with 10% tolerance (test proportion correct (TPC)), i.e ± 0.1 in our case (the PF1 was linearly normalized into the range $\langle 0, 1 \rangle$).

In Figures 6 and 7 we depicted two examples of the dependence of the training and testing error of the KPCR on the number of EM iterations. We used subject A (592 ERPs) and two different initializations of the $\mathbf{\Gamma}$ matrix. We run the experiments for a wide range of Gaussian kernel width parameter L (x-axis). From the upper graphs we can see that using only 5 EM steps a slight overfitting occurs for the width on which minimum testing error was achieved. On the second example (lower graphs) the overfitting appears for bigger values of L. However, this overfitting is in both cases insignificant and from the graphs we can also conjecture that on average 10 EM steps give good results. Similar results were achieved for subject B (776 ERPs) - Figure 8.

Table 4 summarizes the results achieved on the subjects A and B using KPCR, KRR with an extra bias term (eq. (22)) and SVR methods. We displayed the results for kernel width L on which the minimum NMSE on the test set was achieved. This "optimal" width was different for the individual methods. We can see a slightly worse (appr. 2% on NMSE) performance of the SVR method comparing to the KPCR and KRR methods which assume a Gaussian type of noise in the regression model.

In the next experiments we constructed the regression models on all eight subjects. We split the overall data set (5594 ERPs) into three different training (2765 ERPs) and testing (2829 ERPs) data pairs. Again, 20% of the training data set was used for cross-validation in the case of KRR and SVR, respectively. Based on our former studies [21, 23] we used a Gaussian kernel of width

Method	NMSE		TPC	
	A	B	A	B
KPCR *with EMKPCA*	**0.1169**	**0.1746**	**90.44**	**84.54**
	(0.0228)	(0.0207)	(0.02)	(0.02)
KRR *with bias eq. (22)*	**0.1178**	**0.1750**	**90.74**	**84.66**
	(0.0232)	(0.0199)	(0.01)	(0.01)
SVR *with SVMTorch*	**0.1197**	**0.1772**	**90.91**	**84.28**
	(0.0241)	(0.0211)	(0.01)	(0.02)

Table 4: The comparison of the NMSE and TPC prediction errors for subjects A and B. The values represent an average of 10 different simulations and corresponding standard deviation is presented in parentheses.

$L = 6000$. In Figure 9, for one of the training-testing pair, we depict the training and testing error for a different number of selected eigenvectors. In fact, we estimated the first 2650 eigenvectors using 30 EM steps and then constructed KPCR models by removing the eigenvectors corresponding to the smallest eigenvalues. We can see that using more then 2525 principal components we did not improve the performance. Moreover comparing the left and right graphs we can see that further increase of the number of principal components used in KPCR model indicate a possible overfitting effect. On the other hand, reduction of the number of selected principal components to the first 2000 will lead to approximately 8.5% degradation of the performance in terms on NMSE. It is interesting to point out that this choice of 2000 principal components represents only $3.8e^{-6}$% reduction of the described variance compare to the case when 2600 principal components is used.

In Figure 10, for 3 different testing training pairs, we depicted the dependence of the training and testing error of KPCR on the number of EM steps used for estimation of the first 2600 principal components. The results suggests that approximately 30 EM steps are enough to achieve good performance of KPCR on this data set.

Table 5 summarizes the performance of the KPCR with EMKPCA, KRR with a extra bias term (eq. (22)) and SVR. We can see a slightly better performance achieved with the KPCR and KRR models in comparison to SVR (appr. 4% on NMSE). Together with results in Table 4 this suggests that on this data set a Gaussian type of noise is more likely; i.e. the regression models with quadratic cost function are preferable.

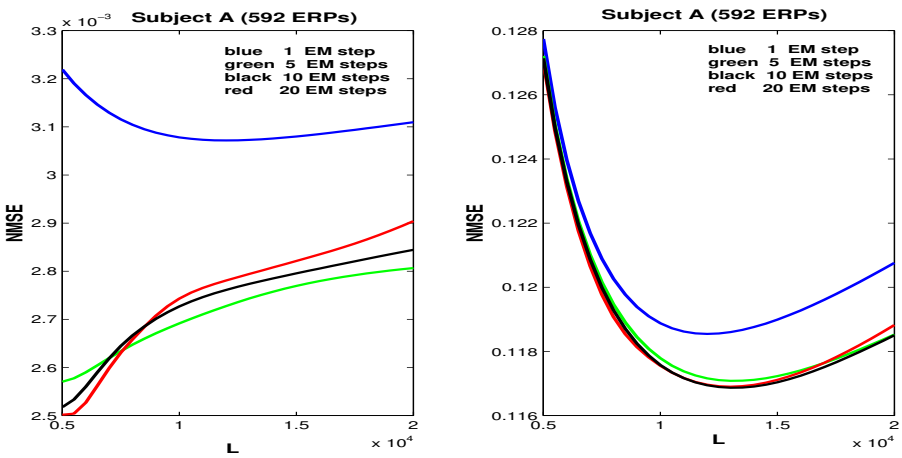

Figure 6: Subject A ($\mathbf{\Gamma}$ matrix initialization #1). Dependence of the KPCR performance based on the number of EM steps used to extract 90% of the non-linear principal components. Individual lines represents the results achieved for a particular number of EM steps using different width (L) parameter of the Gaussian kernel.

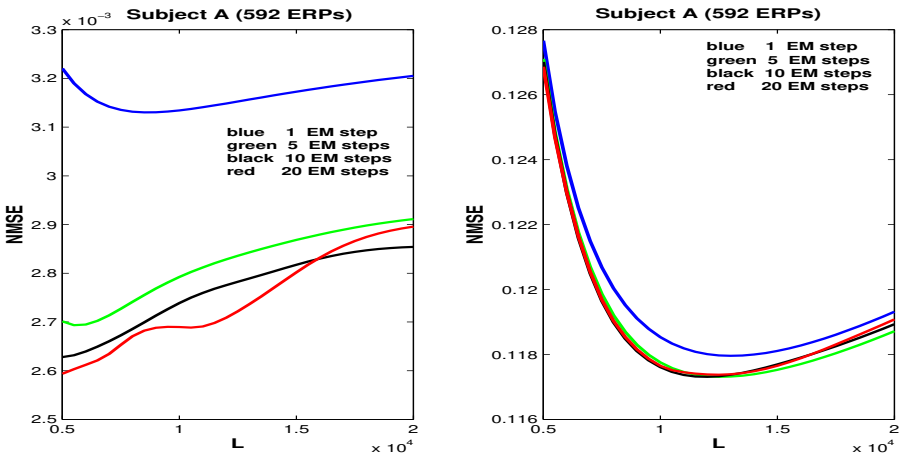

Figure 7: Subject A ($\mathbf{\Gamma}$ matrix initialization #2). Dependence of the KPCR performance based on the number of EM steps used to extract 90% of the non-linear principal components. Individual lines represents the results achieved for a particular number of EM steps using different width (L) parameter of the Gaussian kernel.

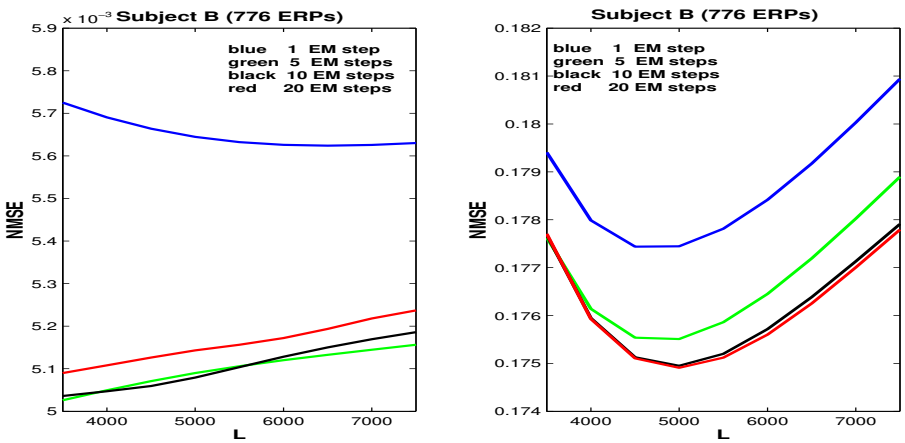

Figure 8: Subject B. Dependence of the KPCR performance based on the number of EM steps used to extract 90% of the non-linear principal components. Individual lines represents the results achieved for a particular number of EM steps using different width (L) parameter of the Gaussian kernel.

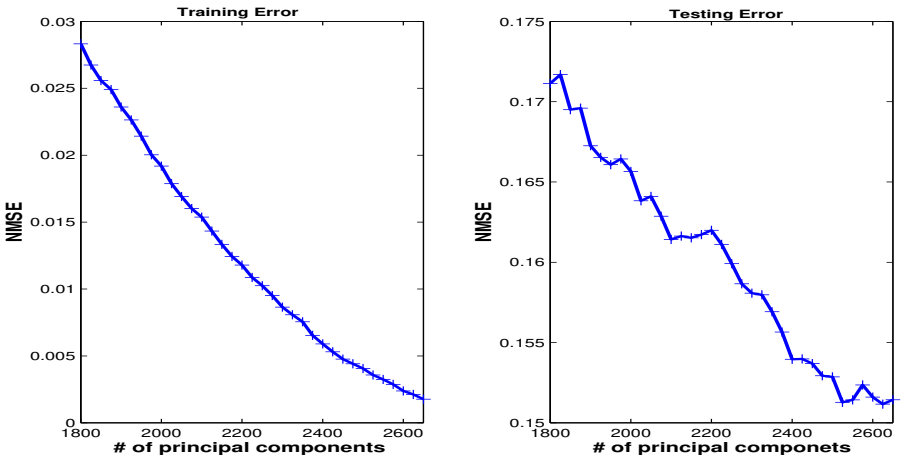

Figure 9: All subjects. Dependence of the KPCR performance in terms of NMSE based on the number of used principal components extracted by EM approach to KPCA. 30 EM steps were used.

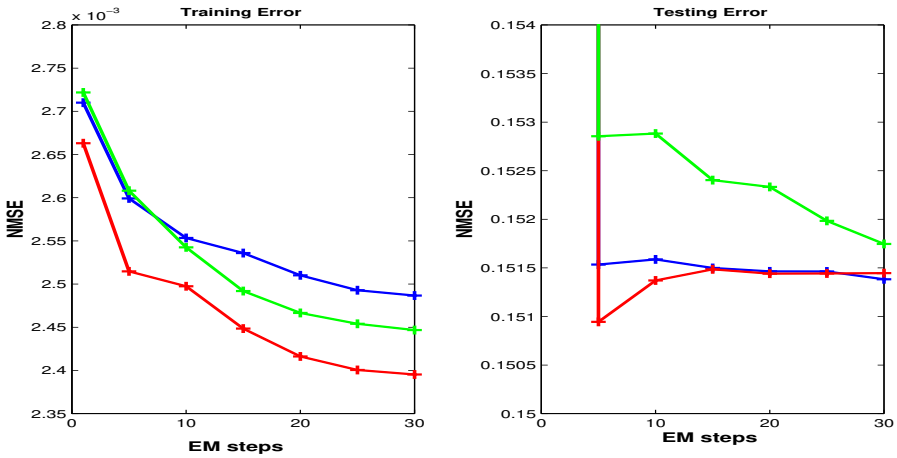

Figure 10: All subjects. Dependence of the KPCR performance in terms of NMSE based on the number of EM steps used to extract 2600 main non-linear principal components. Individual lines represents the results on different training/testing data pairs.

Method	Train NMSE	Test NMSE	TPC
KPCR *with EMKPCA*	**0.0025**	**0.1543**	**83.28**
KRR *with bias eq. (22)*	**0.0039**	**0.1546**	**83.50**
SVR *with SVMTorch*	**0.0109**	**0.1611**	**82.76**

Table 5: The comparison of the NMSE and TPC prediction errors for the model based on all subjects ERPs. The values represent an average of 3 different simulations.

8 Discussion and Conclusions

On two data sets we demonstrated the comparable performance of KPCR with EMKPCA algorithm for kernel principal components extraction in comparison to KRR and SVR. EMKPCA is computationally more attractive when extraction of $p \ll n$ main principal components is required. Moreover, the method can be used without storing of the whole $(n \times n)$ Gram matrix; i.e. with memory requirements equal $\mathcal{O}(p^2) + \mathcal{O}((p+1)n)$. This can be advantageous when dealing with large data sets. The good performance of KPCR with EMKPCA on both data sets also suggests that in the case when the number of "optimal" principal components is more than thousand EMKPCA extracts those principal components sufficiently precisely for desired KPCR. This is a quite important in the situation where we want to apply any model selection technique. Due to the fact that, on some data sets, principal components corresponding to

small eigenvalues can have significant correlation with the dependent variable, we would like to have available the principal components corresponding to the eigenvalues covering almost the whole variance of the training data. This fact was pointed out in [10, 11] and also stressed in the Section 5. On the data sets used in our parallel study [22] we observed that using either CIC or cross-validation model selection techniques, some of the "small" eigenvalues principal components entered a final model improving the performance.

The connection between KRR and KPCR; i.e. regularized linear regression techniques in a feature space was given. Using the parallel of those techniques in input space we have shown how the smaller covariance/variance of the coefficients estimates can be achieved. This is a quite important fact, because as it was shown in [31] we would like to construct the "*flattest*" linear regression models in feature space which together with the smoothness properties of the chosen kernel K would lead to the smooth final non-linear regression estimates. Moreover, on both data sets we observed that penalization of the bias term in the case of KRR degrades the performance. We described two possible ways to include an unpenalized bias term into KRR model and have shown the same improvement of both approaches. The centralization of the KRR model leads to the computational complexity of $\mathcal{O}(n^3)$ while the addition of an extra bias term increases the complexity to $\mathcal{O}((n+1)^3)$. On the other hand, the centralization of KRR model leads to additional computation of the "centralized" training and testing $\mathbf{K}$ matrices (Section 3) and in practical situations we also need to have a representative training set to correctly estimate centralized testing data.

The ϵ-insensitive SVR provides a sparse kernel representation. The selection of a reduced input data set for kernel principal components estimation will lead to a sparse representation of KPCR. In fact in [21] and also in Section 8 on some data sets we have shown that estimation of kernel principal components from a partial input data set did not significantly influence the final performance of KPCR. A possibility to significantly reduce the number of used kernel principal components for the next classification tasks was also reported in [27, 26, 17]. This may potentially lead to a significant reduction of the training set used for the estimation of the main principal components and effectively allows us to deal with larger data sets. We conjecture that in the case of the classification this results are more intuitive, as usually the data structure can be revealed by a few main eigenvectors. This fact was also stressed in [42] where the lower bound for the selection of the eigenvalues was proposed. On the other hand, some of the presented results also suggest that in regression tasks one should be more careful about significant reduction of a number of used principal components based only on criterion derived from a proportion of variance described by the main principal components.

Acknowledgments

We would like to thank Mark Girolami for helpful discussions during the preparation of the manuscript. ERPs data were obtained under a grant from the US Navy Office of Naval Research (PE60115N), monitored by Joel Davis and Harold Hawkins. R. Rosipal is funded by a research grant for the project "Objective Measures of Depth of Anaesthesia"; University of Paisley and Glasgow Western Infirmary NHS trust, and is partially supported by Slovak Grant Agency for Science (grants No. 2/5088/00 and No. 00/5305/468). L. Trejo is supported in part by the U.S. National Aeronautics and Space Administration, Aerospace Operations Systems Program, Psychological and Physiological Stressors and Factors Project.

References

[1] N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404, 1950.

[2] R. Collobert and S. Bengio. Support Vector Machines for Large-Scale Regression Problems. Technical Report 00-17, IDIAP, August 2000.

[3] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines.* Cambridge University Press, 2000.

[4] A.P. Dempster, N.M. Laird, and D.R. Rubin. Maximum likelihood for incomplete data via the EM algorithm. *J. R. Statist. Soc. B*, 39:1–38, 1977.

[5] T. Evgeniou, M. Pontil, and T. Poggio. Regularization Networks and Support Vector Machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

[6] I.E. Frank and J.H. Friedman. A Statistical View of Some Chemometrics Regression Tools. *Technometrics*, 35(2):109–147, 1993.

[7] M. Gibss and D. J. .C. MacKay. Efficient implementation of Gaussian processes. Technical report, Department of Physics, Cavendish Laboratory, Cambridge University, UK, 1997.

[8] F. Girosi. An equivalence between sparse approximation and Support Vector Machines. *Neural Computation*, 10(6):1455–1480, 1998.

[9] F. Girosi, M. Jones, and T. Poggio. Priors, stabilizers and basis functions: From regularization to radial, tensor and additive splines. Technical Report A.I. Memo No. 1430, MIT, 1993.

[10] I. T. Jolliffe. *Principal Component Analysis.* Springer-Verlag, New York, 1986.

[11] I.T. Jolliffe. A Note on the Use of Principal Components in Regression. *Applied Statistics*, 31:300–302, 1982.

[12] G. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation of stochastic processes and smoothing by splines. *Annals of Mathematical Statistics*, 41:495–502, 1970.

[13] G. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *J. Math. Anal. Applic.*, 33:82–95, 1971.

[14] M. Koska, R. Rosipal, A. König, and L. J. Trejo. Estimation of human signal detection performance from ERPs using feed-forward network model. In *Computer Intensive Methods in Control and Signal Processing, The Curse of Dimensionality.* Birkhauser, Boston, 1997.

[15] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London*, A209:415–446, 1909.

[16] S. Mika. Nichtlineare Signalverarbeitung in Feature-Räumen. Master's thesis, Technische Universität Berlin, 1998.

[17] P. Moerland. *Mixture Models for Unsupervised and Supervised Learning.* PhD thesis, École Polytechnique Fédérale de Lausanne, Computer Science Department, 2000.

[18] D. C. Montgomery and E.A. Peck. *Introduction to Linear Regression Analysis*. John Wiley & Sons, II. edition, 1992.

[19] R.J. Radke. A Matlab Implementation of the Implicitly Restarted Arnoldi Method for Solving Large-Scale Eigenvalue Problems . Technical Report TR96-07, Dept. of Comp. and Applied Math, Rice University, Houston,Texas, 1996.

[20] R. Rosipal and M. Girolami. An Expectation-Maximization Approach to Nonlinear Component Analysis. to appear *Neural Computation*, 2001.

[21] R. Rosipal, M. Girolami, and J.L. Trejo. Kernel PCA for Feature Extraction of Event-Related Potentials for Human Signal Detection Performance. In *Proceedings of ANNIMAB-1 Conference*, pages 321–326, Götegorg, Sweden, 2000. Springer.

[22] R. Rosipal, M. Girolami, and L. J. Trejo. On Kernel Principal Component Regression with Covariance Inflation Criterion for Model Selection. Technical report, University of Paisley, 2000.

[23] R. Rosipal, M. Girolami, L. J. Trejo, and A. Cichocki. Kernel PCA for Feature Extraction and De-Noising in Non-Linear Regression. to appear *Neural Computing & Applications* , 2001.

[24] S. Roweis and Z. Ghahramani. A unifying Review of Linear Gaussian Models. *Neural Computation*, 11:305–345, 1999.

[25] C. Saunders, A. Gammerman, and V. Vovk. Ridge Regression Learning Algorithm in Dual Variables. In *Proc. of the 15th International Conference on Machine Learning*, 1998.

[26] B. Schölkopf, S.Mika, C.J.C.Burges, P.Knirsch, K.R.Müller, G.Rätsch, and A.J.Smola. Input Space vs. Feature Space in Kernel-Based Methods. *IEEE Transactions on Neural Networks* , 10(5):1000–1017, 1999.

[27] B. Schölkopf, A. J. Smola, and K. R. Müller. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, 10:1299–1319, 1998.

[28] L. Sirovich. Turbulence and the dynamics of coherent structures. *Quarterly of Applied Mathematics*, 45:561–590, 1987.

[29] A. J. Smola and B. Schölkopf. A Tutorial on Support Vector Regression. Technical Report NC2-TR-1998-030, NeuroColt2 Technical Report Series, 1998.

[30] A.J. Smola and B. Schölkopf. Sparse Greedy Matrix Approximation for Machine Learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann, 2000.

[31] A.J. Smola, B. Schölkopf, and K. R. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11:637–649, 1998.

[32] J.A.K. Suykens, L. Lukas, and J. Vandewalle. Sparse approximation using least squares support vector machines. In *IEEE International Symposium on Circuits and Systems ISCAS'2000*, 2000.

[33] R. Tibshirani and K. Knight. The covariance inflation criterion for adaptive model selection. *J. R. Statist. Soc. B*, 61(3):529–546, 1999.

[34] M.E. Tipping and C.M. Bishop. Probabilistic principal component analysis. *J. R. Statist. Soc. B*, 61:611–622, 1999.

[35] G. Ferrari Trecate, C.K.I. Williams, and M. Opper. Finite-dimensional approximation of Gaussian processes. In M.S. Kearns, S.A. Solla and D.A. Cohn, editor, *Advances in Neural Information Processing Systems 11*, pages 218–224. MIT Press, 1999.

[36] L. J. Trejo, A. F. Kramer, and J. A. Arnold. Event-related Potentials as Indices of Display-monitoring Performance. *Biological Psychology*, 40:33–71, 1995.

[37] L. J. Trejo and M. J. Shensa. Feature Extraction of ERPs Using Wavelets: An Application to Human Performance Monitoring. *Brain and Language*, 66:89–107, 1999.

[38] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1998.

[39] G. Wahba. *Splines Models of Observational Data*, volume 59 of *Series in Applied Mathematics*. SIAM, Philadelphia, 1990.

[40] J.H. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, 1965.

[41] C.K.I. Williams. Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In M.I. Jordan, editor, *Learning and Inference in Graphical Models*. Kluwer, 1998.

[42] C.K.I. Williams and M. Seeger. The Effect of the Input Density Distribution on Kernel-based Classifiers. In *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann, 2000.

Appendix A

We would like to stress several facts concerning EMKPCA. First from the definition of our probabilistic PCA model it is clear that the latent variables $\mathbf{y}$ have identity covariance matrix. Thus at the convergence the projection of the observed data to the found p dimensional subspace will lead to the whitening of the projected data. As we noted in Section 4 the maximum-likelihood estimation $\mathbf{Q}_{ML}$ for the probabilistic PCA model can be express in the form consisting from the principal eigenvectors of the data covariance matrix, corresponding eigenvalues diagonal matrix and arbitrary rotation matrix $\mathbf{Q}_{ML} = \mathbf{V}\mathbf{\Lambda}^{1/2}\mathbf{R}$ [34]. Thus to avoid different scaling in the individual eigendirections we need to find $\mathbf{\Lambda}$ matrix and to use the $\mathbf{Q}_{ML}\mathbf{\Lambda}^{-1/2}$ matrix; i.e. changing the projection (13) to the

$$\beta(\mathbf{x}) := \mathbf{\Lambda}^{1/2}\mathbf{R}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\Phi(\mathbf{x}) = \mathbf{\Lambda}^{1/2}\mathbf{R}(\mathbf{\Gamma}^T\mathbf{K}\mathbf{\Gamma})^{-1}\mathbf{\Gamma}^T\mathbf{K}^{test}. \qquad (24)$$

Now, let us show how the matrix $\mathbf{\Lambda}$ can be recovered. From the fact that we defined $\mathbf{Q} = \mathbf{\Phi}^T\mathbf{\Gamma}$ and from (7) we can write

$$\{\mathbf{\Phi}^T\mathbf{\Gamma}\}_{ML} = \mathbf{\Phi}^T\mathbf{\Upsilon}\mathbf{\Lambda}^{1/2}\mathbf{R}$$

where $\mathbf{\Upsilon}$ now represents the matrix with columns created by estimated eigenvectors $\{\boldsymbol{\alpha}^k\}_{k=1}^p$ of the Gram matrix $\mathbf{K}$. Assuming positive definiteness of the

$\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T$ matrix (see remark in Section 4) we can see that at the convergence we also have the relation

$$\mathbf{\Gamma}_{ML} = \mathbf{\Upsilon}\mathbf{\Lambda}^{1/2}\mathbf{R} \,.$$

It was pointed out in [26] and [16] that the diagonalization of the $\mathbf{K}^2$ instead of $\mathbf{K}$ leads to the same eigenvectors and squared eigenvalues. Further the $n\mathbf{K}^2$ can be seen as the covariance matrix of *the empirical kernel map* Φ_{emp} defined for a given set of points $\{\mathbf{x}_i\}_{i=1}^n$ as

$$\Phi_{emp} : R^N \to R^n;$$

$$\mathbf{x} \to K(.,\mathbf{x})|_{\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}} = (K(\mathbf{x}_1,\mathbf{x}),\ldots,K(\mathbf{x}_n,\mathbf{x})).$$

This fact was recently also used in [17] where similar EMKPCA algorithm was proposed. That simply means that we can assume the probabilistic PCA model of the form $\mathbf{K}_{emp} = \mathbf{\Gamma}\mathbf{Y}$ where the columns of $\mathbf{K}_{emp}$ now represents centralized empirical kernel maps $\{\Phi_{emp}(\mathbf{x}_i)\}_{i=1}^n$[9]. In the next step we can take the orthonormal basis created by orthogonalization[10] the columns of $\mathbf{\Gamma}$ and project the observed data to the p-dimensional subspace defined by this orthogonal matrix $\mathbf{\Gamma}_{orth}$. By applying standard PCA on the covariance matrix of the projected data $\mathbf{Y} = \mathbf{\Gamma}_{orth}^T\mathbf{K}_{emp}$ we can recover the desired squared eigenvalues[11] of the Gram matrix $\mathbf{K}$. Now, the projection of the test point $\mathbf{x}$ can by done using either (24) or by projection of the centralized empirical mapping $\Phi_{emp}(\mathbf{x})$ by $\mathbf{\Gamma}_{orth}^T\mathbf{V}_Y$ where the $\mathbf{V}_Y$ is the matrix of the eigenvectors found by diagonalization of the covariance matrix of $\mathbf{Y}$.

At the end of this appendix we would like to stress one interesting fact. If we can allow the whitening of the projected data in the found p-dimensional subspace we can simply reduce the computational demands of the projection of the training data points by realizing that the rows of orthonormal matrix $\mathbf{\Gamma}_{orth}$ are in fact desired projected data (or latent variables $\mathbf{y}$). This can be seen from matrix form of our probabilistic PCA model in feature space $\mathcal{F}$

$$\mathbf{\Phi}^T = \mathbf{Q}\mathbf{Y} = \mathbf{\Phi}^T\mathbf{\Gamma}\mathbf{Y}$$

which can be rewritten after multiplication of the both sides by $\mathbf{\Phi}$ as

$$\mathbf{K} = \mathbf{K}\mathbf{\Gamma}\mathbf{Y} \,.$$

Providing the orthonormalization of the $\mathbf{\Gamma}$ matrix and again assuming the positive definiteness of the $\mathbf{K} = \mathbf{\Phi}\mathbf{\Phi}^T$ matrix this leads to

$$\mathbf{\Gamma}_{orth}^T = \mathbf{Y} \,.$$

[9] Let us remind, that in practice we diagonalize "centered" matrix $\tilde{\mathbf{K}}$ (eq. (10)). This simply means that for $\mathbf{K}_{emp}$ we have to take $\tilde{\mathbf{K}}$ with centered columns and implicitly assume the empirical map $\tilde{\Phi}_{emp}$ defined by $\mathbf{x} \to \tilde{K}(.,\mathbf{x})|_{\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}} = (\tilde{K}(\mathbf{x}_1,\mathbf{x}),\ldots,\tilde{K}(\mathbf{x}_n,\mathbf{x}))$.

[10] Orthogonalization will not change p-dimensional subspace defined by the columns of $\mathbf{\Gamma}$.

[11] The relation between these extracted eigenvalues $\tilde{\lambda}_k$ and the eigenvalues λ_k of the covariance matrix $\hat{\mathbf{C}}$ (5) is $\lambda_k = \sqrt{\frac{\tilde{\lambda}_k}{n}}$ [16].

Effectively, it means to use the projection[12]

$$(\mathbf{\Gamma}_{orth}^T \mathbf{K} \mathbf{\Gamma}_{orth})^{-1} \mathbf{\Gamma}_{orth}^T \mathbf{K} = \mathbf{\Gamma}_{orth}^T \ .$$

However, to project the new testing point we need to use (13).

Appendix B

First assume the E-step; i.e equation (11) $\mathbf{Y} = (\mathbf{\Gamma}^T \mathbf{K} \mathbf{\Gamma})^{-1} \mathbf{\Gamma}^T \mathbf{K}$. In first step we have to allocate and compute the $(n \times p)$ $\mathbf{\Gamma}$ and $(p \times p)$ $\mathbf{\Gamma}^T \mathbf{K} \mathbf{\Gamma}$ matrices. The computation of the later matrix does not require storing of the $(n \times n)$ Gram matrix $\mathbf{K}$ because the procedure can be done by the elements of $\mathbf{K}$. In next step we can compute the right hand side $(p \times n)$ $\mathbf{\Gamma}^T \mathbf{K}$ matrix. However this will increase the memory requirements only by allocating one n dimensional vector into which we need to temporarily store the results of multiplication of one particular row of $\mathbf{\Gamma}^T$ matrix with the columns of $\mathbf{K}$. It is clear that this procedure can significantly slow down the algorithm. So, if we have enough memory we can do the procedure for a couple of rows of $\mathbf{\Gamma}$ matrix at the same time. Moreover, if we can allocate additional $(n \times p)$ matrix the whole algorithm can be significantly faster as we will need to compute $\mathbf{\Gamma}^T \mathbf{K}$ only one time. In next step we need to compute the $\mathbf{Y}$ matrix, however this will not increase the memory requirements because now we can overwrite the $\mathbf{\Gamma}^T \mathbf{K}$ matrix. Up till now we assumed we are dealing with "centralized" $\mathbf{K}$ matrix. As we noticed in Section 3 the centralization is given by $\tilde{\mathbf{K}} = \mathbf{K} - \mathbf{1}_n \mathbf{K} - \mathbf{K} \mathbf{1}_n + \mathbf{1}_n \mathbf{K} \mathbf{1}_n$ where $\mathbf{1}_n$ is a $(n \times n)$ matrix of $1/n$ elements. More detailed look will reveal that the individual columns $\{[\mathbf{1}_n \mathbf{K}]^j\}_{i=j}^n$ of the $\mathbf{1}_n \mathbf{K}$ matrix consist of the same numbers $\frac{1}{n} \sum_{i=1}^n K_{ij}$ and that the $\mathbf{K} \mathbf{1}_n$ is just the transpose of $\mathbf{1}_n \mathbf{K}$. Similar we can see that the elements of $\mathbf{1}_n \mathbf{K} \mathbf{1}_n$ are $\frac{1}{n^2} \sum_{i,j=1}^n K_{ij}$. To speed up the centralization procedure this $n + 1$ values can be computed in advanced and stored during execution of the EM algorithm. To compute the new $\mathbf{\Gamma}$ matrix in M-step $\mathbf{\Gamma}^{new} = \mathbf{Y}^{\mathrm{T}} (\mathbf{Y} \mathbf{Y}^{\mathrm{T}})^{-1}$ we can use the same approach as described for E-step. We can see that again we need to allocate the $(n \times p)$ and $(p \times p)$ matrices plus an n-dimensional vector.

Summarizing the both steps we can see that the memory requirements can be reduced to the $\mathcal{O}(p^2) + \mathcal{O}((p+1)n)$, however, as we discussed above, this will be done at the expense of the speed the algorithm. We conjecture that a good compromise can be achieved by allocating extra $(p \times n)$ space for storing of the $\mathbf{\Gamma}^T \mathbf{K}$ matrix.

[12] The equality can by simply proved by realizing that $\mathbf{C}^T \mathbf{C} = \mathbf{\Gamma}^T \mathbf{K} \mathbf{\Gamma} = \mathbf{R}^T \mathbf{\Lambda} \mathbf{R}$ and that due to the fact that rank of $\mathbf{\Gamma}$ is p, the $(p \times p)$ matrix used for orthogonalization of $\mathbf{\Gamma}$ (Gram-Schmidt orthogonalization) will have full rank p.