KERNEL PRINCIPAL COMPONENT ANALYSIS (KPCA) FOR THE DE-NOISING OF COMMUNICATION SIGNALS

*GRIGORIOS S. **KOUTSOGIANNIS*,** JOHN J. SORAGHAN[+],*
SIGNAL PROCESSING DIVISION, DEPARTMENT OF EEE,
UNIVERSITY OF STRATHCLYDE, GLASGOW, G1 1XW,UK
e-mail: grigorios@spd.eee.strath.ac.uk*
[+] j.soraghan@eee.strath.ac.uk

ABSTRACT

This paper is concerned with the problem of de-noising for non-linear signals. Principal Component Analysis (PCA) cannot be applied to non-linear signals however it is known that using kernel functions, a non-linear signal can be transformed into a linear signal in a higher dimensional space. In that feature space, a linear algorithm can be applied to a non-linear problem. It is proposed that using the principal components extracted from this feature space, the signal can be de-noised in its input space.

1.0 INTRODUCTION

Principal Component Analysis (PCA) is a powerful technique for reducing the dimensionality of a dataset. PCA is an orthogonal basis transformation which when applied to a set of n correlated variables transforms them into a new set of d uncorrelated variables with $d \leq n$ [1]. Choosing a fewer number of variables than originally existed, it is possible to remove some of the noise that exists in a noisy signal. This occurs because each new variable represents a part of the variation of the signal. When PCA is applied to non-linear signals, it cannot adequately capture the structure of the signal.

Scholkopf et al. have introduced a non-linear version of PCA using kernel functions [2]. The main idea behind this algorithm is to transform the non-linear input space into a high dimensional feature space where linear algorithms can be applied. The transformation of the signal is done using kernel functions. The kernel induced feature space has the advantage that up to infinite dimensions can be used efficiently. This is achieved using implicit mapping. The signal is not actually mapped into the feature space but instead an inner product is used. This technique has been successfully used in classification, image recognition and text categorization [3].

In this paper a Kernel-PCA (KPCA) algorithm is developed and applied to DQPSK modulated signals. The signals are passed through an AWGN channel, and KPCA is used to de-noise the signal. A comparison with linear PCA is also given.

In Section 2, the Kernel-PCA algorithm is presented. In Section 3, the effect of applying KPCA on noisy signals is demonstrated. Additionally, a comparison between linear PCA and KPCA is illustrated. Finally Section 4 concludes the paper.

2.0 KERNEL PCA THEORY

2.1 Kernel Induced Spaces

The first step in kernel-induced space is to map the data into another space. Thus for an input space X, the new *feature space*, is $F=\{\varphi(\mathbf{x}): \mathbf{x} \in X\}$ where φ is the mapping process. A feature map can simplify the classification task as seen in figure 2.1. It is an example of feature mapping from a two-dimensional input space to a two dimensional feature space. Normally the dimensions of the feature space are higher than the input space. The data cannot be separated linearly in input space however it is linearly separable in feature space.

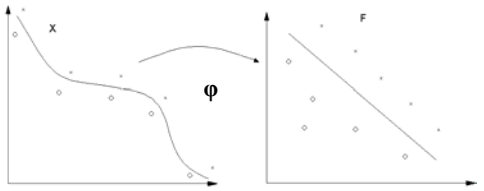

Figure 2.1 From non-linear input space to linear feature space

Although the approach of mapping the data into a feature space solves the non-linearity problem, it introduces two additional problems. The first is that the feature space is a high-dimensional space, which results in high computational demands. The second is the generalization theory problem or curse of dimensionality, which is overfitting in high-dimensional spaces [4]. To solve the computational problems, kernel functions are used. A kernel is a function K, such that for all $\mathbf{x}, \mathbf{z} \in X$, with φ being the mapping from X to the inner product feature space F:

$$K(\mathbf{x},\mathbf{z}) = <\phi(\mathbf{x}).\phi(\mathbf{z})> \qquad (1)$$

The idea of kernels generalizes the standard inner product in the input space [2]. This inner product provides an example of a kernel by making the feature map the identity such as:

$$K(\mathbf{x},\mathbf{z}) = <\mathbf{x}.\mathbf{z}> \qquad (2)$$

Assume for example the decision function for linear classifiers $f(x) = <\mathbf{w}.\mathbf{x}> + b$, where $\mathbf{w}$ and b are the

parameters that control the function. The weights **w** can be written as a linear combination of the N_x training points,

$$\mathbf{w} = \sum_{i=1}^{N_x} a_i y_i \mathbf{x}_i \qquad (3)$$
$$a_i \geq 0$$

where α_i are positive coefficients proportional to the number of times misclassification of $\mathbf{x}_i$ has caused the weight to be updated, and y_i is the classification [4]. Substituting equation 2 in equation 3, the decision function becomes

$$f(\mathbf{x}) = \sum_{i=1}^{N_x} a_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \qquad (4)$$

From equation 4, it is obvious that there is no need to know the underlying feature map to process the data, because the data exists only in the inner product (equation 2). The use of kernels makes it possible to map the data implicitly into a feature space, overcoming the computational problems [2]. Substituting all occurrences of dot products with a priori chosen kernel function transforms the data into the feature space.

In order to ensure that the kernel will be efficient to transform the data it must have some properties. Mercer's theorem [6] is used to provide the answer of when a function K (**x**, **z**) is a kernel. First of all it must be a symmetric function. Because of that there will be a matrix **V** such that **K=V**Λ**V`**, where Λ is a diagonal matrix containing the eigenvalues λ_t of **K**, with corresponding eigenvectors $\mathbf{v}_t$ the columns of **V**. Assume that an eigenvalue is negative. The norm of a point **x** is given by,

$$\|\mathbf{x}\|^2 = <\mathbf{x}\mathbf{x}> = \mathbf{v}_s'\mathbf{V}\sqrt{\Lambda}\sqrt{\Lambda}\mathbf{V}'\mathbf{v}_s' = \mathbf{v}_s'\mathbf{K}\mathbf{v}_s = \lambda_s \qquad (5)$$

The eigenvalues must therefore be non-negative because otherwise the norm of a point **x** in space will be negative, contradicting the geometry of that space [6]. So a function K is a kernel function if and only if the matrix $\mathbf{K} = K(\mathbf{x}_i, \mathbf{x}_j)$ is positive semi-definite.

The most commonly used kernel functions [1] are:

- Polynomials $k(x,y)=(x.y)^d$
- Radial Basis Functions $k(x,y)=exp(-||x-y||^2/2\sigma^2)$
- Sigmoid Kernels $k(x,y)=tanh(k(x.y)+\Theta)$

or combinations of these.

From equation 2 it is obvious that the resulting feature space will have a number of dimensions equal to the number of observations. Although this seems difficult to process, it is noted that the algorithm does not work in the full feature space but a small linear subspace of it [2]. This subspace is chosen without requiring the knowledge of the mapping into feature space, which simplifies the process.

2.2 Kernel PCA Algorithm

Assume a given set of M points x_i, with i=1,…,M, $x_i \in R^N$. Initially, map the data into the feature space using one of the kernel functions.

$$\Phi : R^N \rightarrow F$$
$$\mathbf{x} \rightarrow \mathbf{X} \qquad (6)$$

The size of the resulting square matrix will be equal to the number of the points of the dataset. The next step is to center the data in the feature space. Finally the eigenvalues have to be calculated. Using the covariance matrix in F,

$$C = \frac{1}{M}\sum_{i=1}^{M} \Phi(x_j)\Phi(x_j)^T \qquad (7)$$

find the eigenvalues λ, as specified in equation 5. Additionally, they must be arranged in descending order so that the first λ gives the highest variance. The eigenvectors of the data in feature space $\mathbf{V} \in F$ ($\neq 0$) are found by satisfying,

$$\lambda \mathbf{V} = C\mathbf{V} \qquad (8)$$

V will be of the form $[\alpha_{i1},\alpha_{i2},\ldots,\alpha_{iM}]$. Note here, that the k^{th} corresponding vector has to be normalized by the eigenvalue, using $1=\lambda^k<V^k.V^k>$, in order to obtain the optimum principal component extractor in the sense of the shortest weight vector. After the eigenvectors are obtained, the kernel principal components have to be extracted. This is done by projecting the points of the signal into each eigenvector as seen in equation 9.

$$PC_k\Phi(\mathbf{x}) = (\mathbf{V}^k.\Phi(\mathbf{x})) = \sum_{i=1}^{M} a_i^k <\Phi(\mathbf{x}_i).\Phi(\mathbf{x})> \qquad (9)$$

The number of kernel principal components extracted is important due to the fact that if they are too few not all the structure of the data will be represented. On the other hand if a large number of kernel principal component is extracted then more noise will be encapsulated making the de-noising more difficult.

Due to the fact that a non-linear map was used to move into the feature space it cannot be said that there is a one-to-one correspondence between the points of the feature space and the points of the input space. Therefore it cannot be said that each point in the span of the mapped input data is necessarily the image of some input pattern. Thus, to move the points from feature space into input space, a "pre-image" of these points is required. In some cases, an exact pre-image is not possible (Gaussian Kernels) so an approximate pre-image is computed as follows [7],

$$z_{n+1} = \frac{\sum_{i=1}^{M} a_i \exp(-\|\mathbf{x}_i - z_n\|^2/(2\sigma^2))\mathbf{x}_i}{\sum_{i=1}^{M} a_i \exp(-\|\mathbf{x}_i - z_n\|^2/(2\sigma^2))} \qquad (10)$$

Using the iteration seen in equation 10, the points of the feature space are moved back into the input space. The result is to place each point to the center of the appropriate cluster and this allows

de-noising to be accomplished. The amount of error depends on the initial values chosen for z. In the cases studied it was not possible to pre-image correctly. Using equation 10 all the points were centered on a particular point near the initial value.

In de-noising, the noisy points themselves can be used as initial values. Thus equation 10 can be re-written as equation 11, where each point is used as an initial value and is run through all the other points. For a number of iterations N<M, the last iteration N moves the reconstructed point **x'** to the center of its constellation:

$$\mathbf{x}'_m = \frac{\sum_{i=1}^{N} a_i \exp(-\| \mathbf{x}_m - \mathbf{x}_i \|^2 / (2\sigma^2))\mathbf{x}_i}{\sum_{i=1}^{N} a_i \exp(-\| \mathbf{x}_m - \mathbf{x}_i \|^2 / (2\sigma^2))} \qquad (11)$$

Now, all the points are moved to the centers of the constellations achieving apart from the classification, the de-noising of the dataset as well.

3.0 SIMULATIONS

To generate a modulated signal a pseudorandom process was used as an input. 8100 points were fed to a DQPSK modulator, which was passed through an AWGN channel. Without noise, the data only exists in position [0 1], [1 0], [-1 0], and [0 –1] as seen in Figure 3.1(a). Additionally, in figure 3.1(b) the effect of noise on the signal is illustrated. The difficulty of this case is that some points have moved into the region of other constellation making it impossible to use the distance as a classification tool.

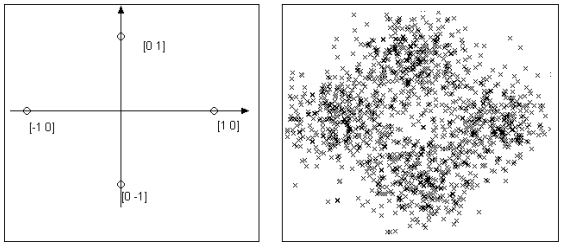

Figure 3.1(a) Clean DQPSK Figure 3.1 (b) Noisy DQPSK E_s/N_o 5dB

To map the dataset into the feature space a Gaussian kernel was used, with variance 0.1. The first set of simulations examined the reconstruction and de-noising using the first four kernel Principal Components (PC) under different noise levels. The dataset is partitioned into smaller sets and the KPCA is applied. The pre-image is then formed using equation 11. Choosing the number of kernel PCs to reconstruct the signal is tricky as an increase in the number of PCs captures more structure of the signal. On the other hand more PCs mean more noise included.

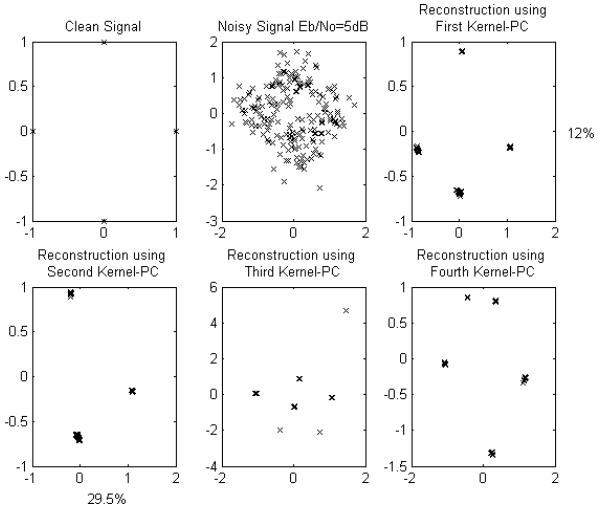

Figure 3.2 Reconstruction at E_s/N_o=5dB

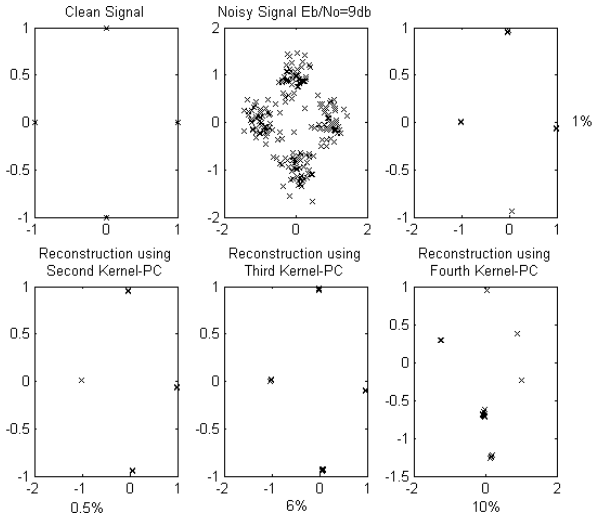

Figure 3.3 Reconstruction at E_s/N_o=9dB

In Figure 3.2 and 3.3, two cases of Es/N_o 5dB and 9dB respectively are demonstrated, along with the reconstruction error for each kernel PC. As the number of kernel PC's used increases the reconstruction error increases. For optimum reconstruction up to two kernel-PCs are required. In figure 3.2 it can be seen that with one kernel-PC all four constellations are obtained where using the second kernel-PC only three constellations appear. The third and fourth kernel-PC cannot center the points into their constellation. In figure 3.3 where noise level is E_s/N_o=9dB, the second kernel-PC yields the best reconstruction error. In figure 3.4 a comparison of the reconstruction errors using different kernel-PCs is provided. The channel had an E_s/N_o from 0dB to 20dB. It can be seen that choosing to reconstruct with the second kernel-PC offers the minimum error.

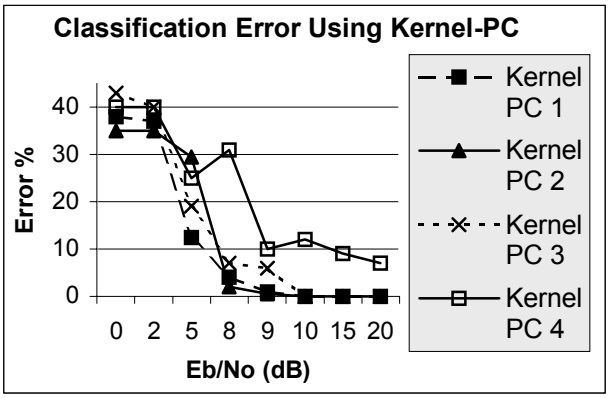

Figure 3.4 Comparison of error using different kernel principal components

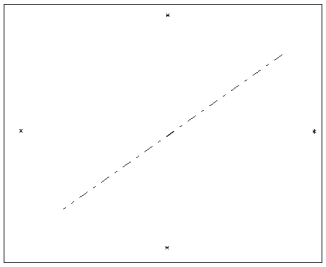

Figure 3.5(a) Linear PCA on clean signal

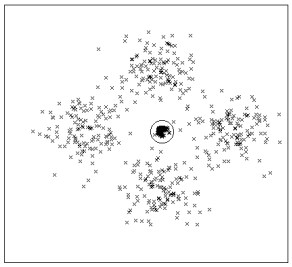

Figure 3.5(b) Linear PCA on noisy signal

The reconstruction error can be affected by the variance of the Gaussian function used to map the signal into the feature space. In the previous experiments, the variance of the Gaussian kernel was set to 0.1. Table 1 shows the different reconstruction errors for the first and the second kernel-PC for σ = 0.05, 0.07, 0.1 and 0.2

Variance	Kernel-PC 1	Kernel-PC 2
0.05	1.65%	2.5%
0.07	1.68%	1.9%
0.1	2%	3%
0.2	6%	10.75%

Table 1 Reconstruction error over different variance at E_s/N_o= 8db

If instead of kernel-PCA, linear PCA was used it is obvious that it would be impossible to capture the structure of the data as it is non-linear. Applying linear PCA in a noiseless DQPSK signal results in obtaining a line, which cuts through the middle of the data. Additionally, in the noisy signal, applying linear PCA results in all the reconstructed points to appear in the center of the noisy signal as illustrated in figure 3.5(a) and 3.5(b) respectively.

4.0 CONCLUSIONS

In this paper a non-linear PCA algorithm was presented for de-noising communication signals. The drawback of the initial reconstruction process is that it cannot perform satisfactory when the points are not in order. In a communication signal the points appear randomly, therefore it was required to re-form the pre-image generation algorithm. The algorithm was tested to DQPSK modulated signals. The signal was passed through a range of noise levels from E_s/N_o=0dB up to 20 dB. The fewer kernel principal components used, the smaller the classification error was. The reconstruction errors achieved using different kernel-PCs were given. Additionally, the performance of the algorithm varied with different variances for the kernel Gaussian function.

In order to improve the reconstruction error, it is proposed to consider a more appropriate kernel function for the mapping process. Additionally, this could be achieved by devising a better iteration for the reconstruction.

It is noted that kernel PCA performed better in a real life case than linear PCA. Kernel PCA can extract up to M number of principal components were M is the number of the training set. This means that the information for the structure of the data is broken into more pieces, allowing better de-noising. Finally, it is appropriate for non-linear signals as linear PCA fails to operate completely as it cannot capture the nonlinearity of the signal.

ACKNOWLEDGEMENT

The authors would like to thank QinetiQ, UK and especially Dr. Anthony Brown for his valuable comments to this work.

5.0 REFERENCES

[1] C. Chatfield, A. J. Collins, *"Introduction to Multivariate Analysis"*, Chapman and Hall, 1980

[2] B. Scholkopf, A. Smola, K.R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem", *Max Planck Institut fur biologishe Kybernetik,* December 1996

[3] Nello Cristianini, John Shawe-Taylor, *"An introduction to Support Vector Machines and other kernel-based learning methods"*, Cambridge University Press, 2000

[4] Nello Cristianini, "Tutorial on Support Vector and Kernel machines", 18th *International Conference on Machine Learning*, Williams College, USA, 2001

[5] V. Vapnik, *The Nature of Statistical Learning Theory,* New York, Springer-Verlag, 1995

[6] "Generalization performance of regularization networks and SVM via entropy numbers and compact operators", *NEUROCOLT,* Tech. Report 19,1998

[7] B. Scholkopf, S. Mika, C.J Burges, Knirsch P., K.R. Muller, G. Ratsch, B. Smola, "Input space versus feature space in kernel-based methods", *IEEE Transactions on neural networks,* Vol. 100, No.5, Sep' 1999