Kernel Principal Component Regression with Application to Nonlinear Prediction

Dachapak, Chooleewan
Department of Electrical and Electronic Systems Engineering, Graduate School of Information Science and Electrical Engineering, Kyushu University : Graduate Student

Kanae, Shunshoku
Department of Electrical and Electronic Systems Engineering, Faculty of Information Science and Electrical Engineering, Kyushu University, Kyushu University

Yang, Zi-Jiang
Department of Electrical and Electronic Systems Engineering, Faculty of Information Science and Electrical Engineering, Kyushu University, Kyushu University

Wada, Kiyoshi
Department of Electrical and Electronic Systems Engineering, Faculty of Information Science and Electrical Engineering, Kyushu University, Kyushu University

https://doi.org/10.15017/1515812

出版情報：九州大学大学院システム情報科学紀要. 8 (1), pp.19-23, 2003-03-26. 九州大学大学院システム情報科学研究院
バージョン：
権利関係：

九州大学大学院
システム情報科学紀要
第8巻 第1号 平成15年3月

Research Reports on Information Science and
Electrical Engineering of Kyushu University
Vol.8, No.1, March 2003

Kernel Principal Component Regression with Application to Nonlinear Prediction

Chooleewan DACHAPAK*, Shunshoku KANAE**, Zi-Jiang YANG** and Kiyoshi WADA**

(Received December 20, 2002)

Abstract: In this study, Kernel Principal Component Analysis (KPCA) is applied as feature selection in a high-dimensional feature space which is nonlinearly related to an input space. By using Mercer Kernels, we can compute principal components in a high dimensional feature space. Then, the extracted features by KPCA method are employed as a new kind of regressors in an ordinary least square regression in the feature space which is Reproducing Kernel Hilbert Space (RKHS). In the experiment, KPCR method was applied to predict the Mackey-Glass time series.

Keywords: Kernel Principal Component Analysis, Kernel Principal Component Regression

1. Introduction

In the present study, we are interested in the feature selection method as the appropriate preprocessing step for an ordinary least squares regression, since the suitable feature selection can increase the overall performance of algorithms [1]. Kernel Principal Component Analysis is based on the computation of the standard linear PCA in the feature space mapped from input space by some nonlinear function [2]. We have applied KPCA method for feature selection in a high dimensional or possibly infinite feature F (with dimension $M \leq \infty$). This allows us to extract the nonlinear principal components up to the number of data points n ($n \leq M$). We compute a dot product in the feature space by means of kernel functions in input space, i.e. $k(\boldsymbol{x}, \boldsymbol{y}) = (\Phi(\boldsymbol{x}).\Phi(\boldsymbol{y}))$, and we never need the mapped patterns $\Phi(\boldsymbol{x})$ explicitly. It is often the case that a small number of Kernel Principal Component is sufficient to account for most of the structure in the data. Also, the Kernel principal components are uncorrelated each other, and it is possible that KPCA can decrease the effect of noise [2]. Then, the features extracted by KPCA method are employed as a regressors of the ordinary least squares regression. This means it performs a linear regression in the feature space F.

2. Kernel Principal Component Analysis

First, mapping the data nonlinearly into the feature space F by the nonlinear function as following,

$$\Phi : \boldsymbol{R}^N \rightarrow F, \quad \boldsymbol{x} \mapsto \boldsymbol{X}. \tag{1}$$

Note that the feature space F could have an arbitrarily large, possibly infinite, dimensionality [3]. Assume that our data mapped into feature space is centered, $\sum_{k=1}^{n} \Phi(\boldsymbol{x}_k) = 0$. Then, in F space, a covariance matrix can be expressed as

$$\bar{C} = \frac{1}{n} \sum_{j=1}^{n} \Phi(\boldsymbol{x}_j) \Phi(\boldsymbol{x}_j)^T. \tag{2}$$

Define $(n \times M)$ matrix $\boldsymbol{\Phi}$,

$$\boldsymbol{\Phi}^T \triangleq \left[\Phi(\boldsymbol{x}_1)\ \Phi(\boldsymbol{x}_2)\ \ldots\ \Phi(\boldsymbol{x}_n) \right]. \tag{3}$$

So $(M \times M)$ matrix $\bar{C}$ can be rewritten as

$$\bar{C} = \frac{1}{n} \boldsymbol{\Phi}^T \boldsymbol{\Phi}. \tag{4}$$

Let $(n \times n)$ matrix K denote $K = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$. Noting $n\bar{C}$ and K's nonzero eigenvalues are the same. Let $\lambda_k, \boldsymbol{V}^k$ be $\bar{C}$'s k^{th}eigenvalues and eigenvectors respectively, where $k = 1, 2, ..., M$.

$$n\lambda \boldsymbol{V}^k = n\bar{C}\boldsymbol{V}^k = \boldsymbol{\Phi}^T \boldsymbol{\Phi} \boldsymbol{V}^k. \tag{5}$$

Pre-multiply $\boldsymbol{\Phi}$,

$$n\lambda \boldsymbol{\Phi} \boldsymbol{V}^k = \boldsymbol{\Phi}\boldsymbol{\Phi}^T \boldsymbol{\Phi} \boldsymbol{V}^k = K \boldsymbol{\Phi} \boldsymbol{V}^k. \tag{6}$$

Let

$$\boldsymbol{\gamma}^k = \boldsymbol{\Phi} \boldsymbol{V}^k, \tag{7}$$

then,

$$K\boldsymbol{\gamma}^k = \tilde{\lambda} \boldsymbol{\gamma}^k. \tag{8}$$

Note that $\tilde{\lambda} = n\lambda$. Moreover from,

$$\boldsymbol{\Phi}^T \boldsymbol{\gamma} = \boldsymbol{\Phi}^T \boldsymbol{\Phi} \boldsymbol{V} = \tilde{\lambda} \boldsymbol{V}, \tag{9}$$

We obtain $\boldsymbol{V}^k = \frac{1}{\tilde{\lambda}} \boldsymbol{\Phi}^T \boldsymbol{\gamma}^k$. Since it is required that

* Department of Electrical and Electronic Systems Engineering, Graduate Student

** Department of Electrical and Electronic Systems Engineering

eigenvector $\boldsymbol{V}$ in F space must be normalized. We obtain the normalized eigenvector as following form,

$$\boldsymbol{V}^k = \frac{1}{\sqrt{\tilde{\lambda}}}\boldsymbol{\Phi}^T\boldsymbol{\gamma}^k. \tag{10}$$

Let $\boldsymbol{\alpha}^k = \frac{1}{\sqrt{\tilde{\lambda}}}\boldsymbol{\gamma}^k$. Then,

$$\boldsymbol{V}^k = \boldsymbol{\Phi}^T\boldsymbol{\alpha}^k. \tag{11}$$

Finally, project $\Phi(\boldsymbol{x})$ onto the k^{th} nonlinear principal component by the dot product between $\boldsymbol{V}^k$ and $\Phi(\boldsymbol{x})$ as below,

$$\boldsymbol{V}^k \cdot \Phi(\boldsymbol{x}) = \boldsymbol{\alpha}^{k^T}\boldsymbol{\Phi}\Phi(\boldsymbol{x}). \tag{12}$$

The above $1^{st}, ..., M^{th}$ nonlinear principal components, $\boldsymbol{\Phi}V$, are employed as a new regressors for the regression model in the feature space. We used only first p components in order to reduce the noise effect since the components whose variances are very small can be assumed as the variance from noise.

3. Kernel Principal Component Regression

Consider the standard regression model in the feature space

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{\xi} + \boldsymbol{\epsilon}, \tag{13}$$

where $\boldsymbol{y}$ is a vector of n observations, $\boldsymbol{\Phi}$ is an $(n \times M)$ matrix of regressors whose i^{th} row is the vector $\Phi(\boldsymbol{x}_i)$ of the mapped $\boldsymbol{x}_i$ observation into $M \leq \infty$ dimensional feature space F, $\boldsymbol{\xi}$ is a vector of regression coefficients and $\boldsymbol{\epsilon}$ is a vector of error terms. We assume that regressors are centered. Now, let $(M \times M)$ matrix V denote

$$V = [\,\boldsymbol{V}^1\ \boldsymbol{V}^2\ \ldots\ \boldsymbol{V}^M\,] \tag{14}$$

then,

$$\boldsymbol{\Phi}^T\boldsymbol{\Phi}V = V\Lambda. \tag{15}$$

Noting Λ is a diagonal matrix of $\{\tilde{\lambda}_1, \tilde{\lambda}_2, \ldots, \tilde{\lambda}_M\}$. V is an orthogonal matrix so the linear regression model can be expressed in the term of new regressors as

$$\boldsymbol{y} = \boldsymbol{\Phi}VV^T\boldsymbol{\xi} + \boldsymbol{\epsilon} = \boldsymbol{B}\boldsymbol{w} + \boldsymbol{\epsilon}, \tag{16}$$

where $\boldsymbol{B} = \boldsymbol{\Phi}V$, $\boldsymbol{w} = V^T\boldsymbol{\xi}$.

Here we can see that the new regressors B are transformed in the term of Kernel Principal Components which are the extracted features we are interested in. And

$$\boldsymbol{B}^T\boldsymbol{B} = V^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}V = V^TV\Lambda = \Lambda \tag{17}$$

So the least squares estimator of coefficient $\boldsymbol{w}$ can be expressed as

$$\hat{\boldsymbol{w}} = (\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{y} = \Lambda^{-1}\boldsymbol{B}^T\boldsymbol{y}. \tag{18}$$

From above, we can see that it is difficult to find $\boldsymbol{B}$ directly. The easier way to find $\boldsymbol{B}$ by using kernel function will be shown.

Let y_k be the k^{th}component of $\boldsymbol{y}$ and ϵ_k be the k^{th}component of $\boldsymbol{\epsilon}$, from the linear regression model, we obtain

$$y_k = \Phi(\boldsymbol{x}_k)^T\boldsymbol{\xi} + \epsilon_k = \boldsymbol{\Phi}(\boldsymbol{x}_k)^TV\boldsymbol{w} + \epsilon_k. \tag{19}$$

Here, consider $V^T\Phi(\boldsymbol{x}_k)$, from the definition of V, we can see that

$$V^T\Phi(\boldsymbol{x}_k) = \begin{bmatrix} \boldsymbol{V}^{1^T}\Phi(\boldsymbol{x}_k) \\ \boldsymbol{V}^{2^T}\Phi(\boldsymbol{x}_k) \\ \vdots \\ \boldsymbol{V}^{M^T}\Phi(\boldsymbol{x}_k) \end{bmatrix} = \begin{bmatrix} \beta_1(\boldsymbol{x}_k) \\ \beta_2(\boldsymbol{x}_k) \\ \vdots \\ \beta_M(\boldsymbol{x}_k) \end{bmatrix}. \tag{20}$$

Note that $\beta_i(\boldsymbol{x}_k) = \boldsymbol{V}^{i^T}\Phi(\boldsymbol{x}_k)$, because $\boldsymbol{V}^i$ and $\boldsymbol{\alpha}^i$ have the relationship as

$$\boldsymbol{V}^i = \boldsymbol{\Phi}^T\boldsymbol{\alpha}^i \tag{21}$$

so $\beta_i(\boldsymbol{x}_k)$ can be represented by

$$\begin{aligned} \beta_i(\boldsymbol{x}_k) &= \boldsymbol{\alpha}^{i^T}\boldsymbol{\Phi}\Phi(\boldsymbol{x}_k) \\ &= \boldsymbol{\alpha}^{i^T}\begin{bmatrix} \Phi(\boldsymbol{x}_1)^T\Phi(\boldsymbol{x}_k) \\ \Phi(\boldsymbol{x}_2)^T\Phi(\boldsymbol{x}_k) \\ \vdots \\ \Phi(\boldsymbol{x}_n)^T\Phi(\boldsymbol{x}_k) \end{bmatrix} \\ &= \boldsymbol{\alpha}^{i^T}\begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_k) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_k) \\ \vdots \\ k(\boldsymbol{x}_n, \boldsymbol{x}_k) \end{bmatrix}. \end{aligned} \tag{22}$$

We compute the dot products $\Phi(\boldsymbol{x}_i)^T\Phi(\boldsymbol{x}_k)$ in the feature space by means of kernel functions in input space $k(\boldsymbol{x}_i, \boldsymbol{x}_k) = \Phi(\boldsymbol{x}_i)^T\Phi(\boldsymbol{x}_k)$. Moreover, we do not need to map $\Phi(\boldsymbol{x})$ explicitly throughout the study. Then, define $\boldsymbol{k}(\boldsymbol{x}_k)$ as following,

$$\boldsymbol{k}(\boldsymbol{x}_k) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_k) \\ k(\boldsymbol{x}_2, \boldsymbol{x}_k) \\ \vdots \\ k(\boldsymbol{x}_n, \boldsymbol{x}_k) \end{bmatrix} \tag{23}$$

$\beta_i(\boldsymbol{x}_k)$ becomes

$$\beta_i(\boldsymbol{x}_k) = \boldsymbol{\alpha}^{i^T}\boldsymbol{k}(\boldsymbol{x}_k). \tag{24}$$

So

$$V^T\Phi(\boldsymbol{x}_k) = \begin{bmatrix} \beta_1(\boldsymbol{x}_k) \\ \beta_2(\boldsymbol{x}_k) \\ \vdots \\ \beta_n(\boldsymbol{x}_k) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\alpha}^{1^T} \\ \boldsymbol{\alpha}^{2^T} \\ \vdots \\ \boldsymbol{\alpha}^{n^T} \end{bmatrix} \boldsymbol{k}(\boldsymbol{x}_k) = A^T\boldsymbol{k}(\boldsymbol{x}_k). \quad (25)$$

Note that

$$A = \begin{bmatrix} \boldsymbol{\alpha}^1 & \boldsymbol{\alpha}^2 & \dots & \boldsymbol{\alpha}^n \end{bmatrix}. \quad (26)$$

We can rewrite (19) as

$$y_k = \boldsymbol{k}(\boldsymbol{x}_k)^T A\boldsymbol{w} + \epsilon_k = \boldsymbol{k}(\boldsymbol{x}_k)^T\boldsymbol{\theta} + \epsilon_k \quad (27)$$

where

$$\boldsymbol{\theta} = A\boldsymbol{w} = AV^T\boldsymbol{\xi}. \quad (28)$$

From the definition of B,

$$B = \boldsymbol{\Phi}V \quad (29)$$

$$B = \begin{bmatrix} \Phi(\boldsymbol{x}_1)^T \\ \Phi(\boldsymbol{x}_2)^T \\ \vdots \\ \Phi(\boldsymbol{x}_n)^T \end{bmatrix} V = \begin{bmatrix} \Phi(\boldsymbol{x}_1)^T V \\ \Phi(\boldsymbol{x}_2)^T V \\ \vdots \\ \Phi(\boldsymbol{x}_n)^T V \end{bmatrix}$$

$$B = \begin{bmatrix} \boldsymbol{k}(\boldsymbol{x}_1)^T \\ \boldsymbol{k}(\boldsymbol{x}_2)^T \\ \vdots \\ \boldsymbol{k}(\boldsymbol{x}_n)^T \end{bmatrix} A, \quad (30)$$

where,

$$\begin{bmatrix} \boldsymbol{k}(\boldsymbol{x}_1)^T \\ \boldsymbol{k}(\boldsymbol{x}_2)^T \\ \vdots \\ \boldsymbol{k}(\boldsymbol{x}_n)^T \end{bmatrix} = \begin{bmatrix} k(\boldsymbol{x}_1,\boldsymbol{x}_1) & k(\boldsymbol{x}_2,\boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_n,\boldsymbol{x}_1) \\ k(\boldsymbol{x}_1,\boldsymbol{x}_2) & k(\boldsymbol{x}_2,\boldsymbol{x}_2) & \cdots & k(\boldsymbol{x}_n,\boldsymbol{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_1,\boldsymbol{x}_n) & k(\boldsymbol{x}_2,\boldsymbol{x}_n) & \cdots & k(\boldsymbol{x}_n,\boldsymbol{x}_n) \end{bmatrix}. \quad (31)$$

We can rewrite B as

$$B = KA. \quad (32)$$

So we can formulate the KPCR model as following expression,

$$\boldsymbol{y} = B\boldsymbol{w} + \boldsymbol{\epsilon} = KA\boldsymbol{w} + \boldsymbol{\epsilon} = K\boldsymbol{\theta} + \boldsymbol{\epsilon}. \quad (33)$$

We prefer (32) to (29) since we can compute B by means of kernel functions in input space without mapping $\Phi(\boldsymbol{x})$ explicitly. As we mentioned above, we employed the first p nonlinear principal components to create a KPCR model in order to decrease the effect of noise and eliminate large variances of the estimate due to multicollinearities. In this case the first p nonlinear principal components mean the first p column vectors of A.

4. Experiment

In this present study, we employed Gaussian kernel function, $k(\boldsymbol{x},\boldsymbol{y}) = e^{-(\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{L})}$, where L determines the width of the Gaussian function. We utilized the noisy Chaotic Mackey-Glass Time-Series applying to KPCR method which the regressors are extracted by Kernel PCA. This KPCR method was 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 over 3000 training samples in steps of 500 samples. So we got six partitions. The simulations were repeated for the width L from range $(0.2\hat{\sigma}, 20\hat{\sigma})$, where $\hat{\sigma}$ is the variance of the overall clean training set, using the step size 0.01. A fixed test set of size 500 data points was employed through out the experiments. The performance of the regression models to predict a clean Mackey-Glass time series was evaluated in terms of the normalized root mean squared error (NRMSE). The best result on the test set averaged over all individual runs are summarized in **Table 1**.

Table 1 The comparison of the approximation errors (NRMSE) for prediction of Mackey-Glass training set. N/S represents the ratio between the standard deviation of the added Gaussian noise and the underlying time-series.

Method	$N/S = 0.0$	$N/S = 0.11$	$N/S = 0.22$
KPCR	0.038	0.307	0.443

5. Conclusions

The Kernel PCA method has been employed for the feature extraction and noise reduction in the preprocessing step for regression problem. With KPCR method, there are some problems occured when we remove some Kernel principal components with small eigenvalues in order to reduce the noise present because those Kernel principal components may have a significant contribution for the prediction.

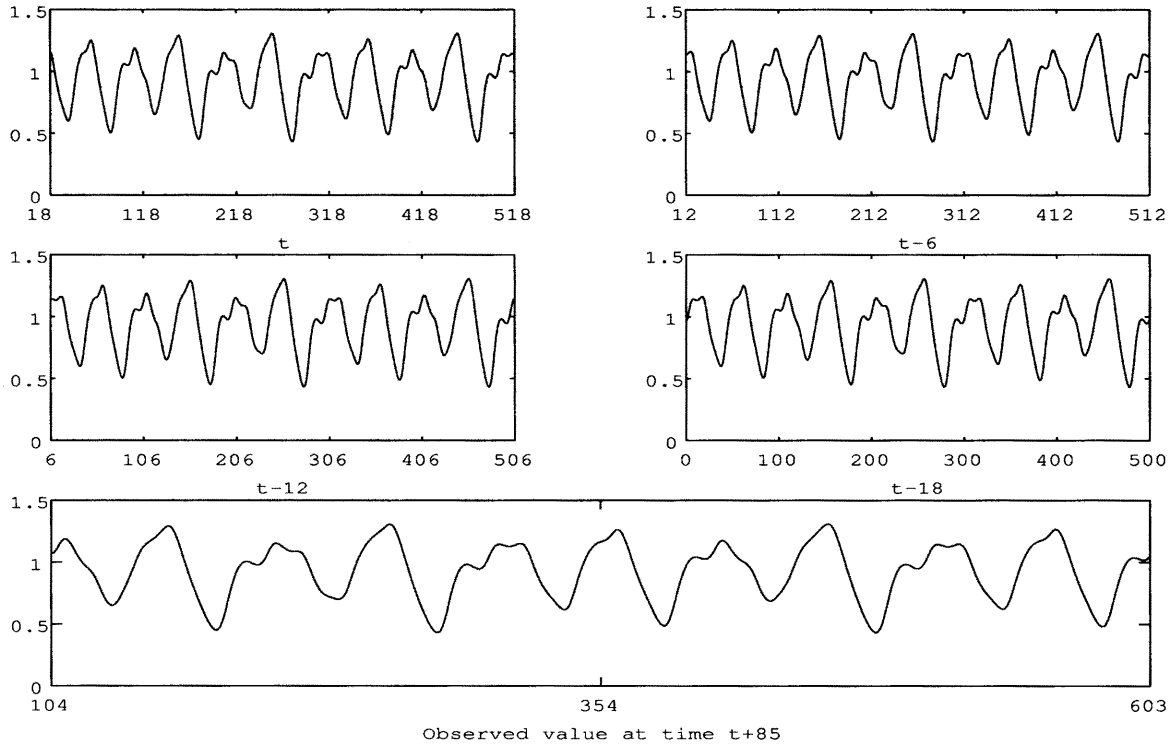

Fig.1 Training data set: 500 samples from 3000 samples of Mackey-Glass time series. First 4 figures show the inputs at time $t, t-6, t-12, t-18$ from $t = 19$ to 518. They are employed as the data set x which are trained by KPCR method to predict the value at time $t+85$ (The bottom shows the observed value at time t+85).

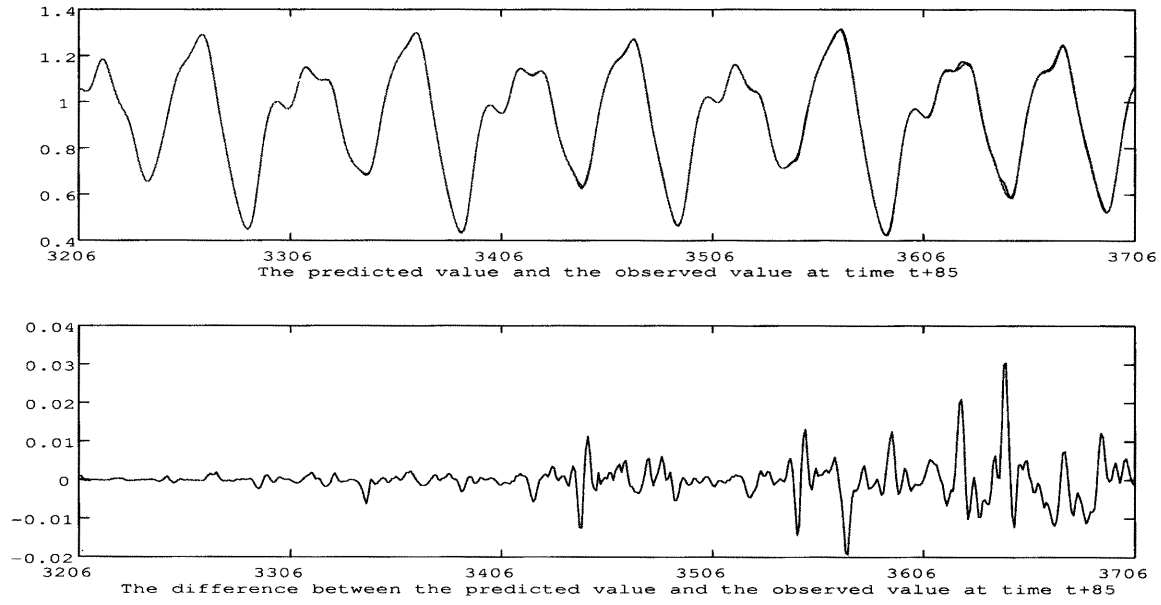

Fig.2 Testing data set: the above shows the predicted value and the observed value at time t+85 (t=3122 to 3621). And the bottom shows the different value between the predicted value and the observed value.

References

1) R. Rosipal, M. Girolami, L.J. Trejo, and A. Cichocki. Kernel PCA for Feature Extraction and De-noising in Non-linear Regression. *Neural Computing & Applications*, 10(3), 2001.
2) B. Schölkopf, C.J.C. Burges, and A.J. Smola. Advances in Kernel Methods Support Vector Learning. *The MIT Press, Cambridge, Massachusetts, London England*, 1998.
3) B. Schölkopf, A.J. Smola, and K.R. Muller. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation*, 10:1299–1319, 1998.
4) R. Rosipal and L.J. Trejo. Kernel Partial Least Squares Regression in Reproducing Kernel Hilbert Space. *Journal of Machine Learning Research*, pages 97–123, December 2(2001).
5) B. Schölkopf. Support Vector Learning. *PhD thesis, Universitat Berlin, Berlin, Germany*, 1997.
6) M.G. Genton. Classes of Kernels for Machine Learning : A Statistics Perspective. *Journal of Machine Learning Research*, pages 299–312, 2(2001).
7) I.T. Jolliffe. Principal Component Analysis. *Springer-Verlag, New York*, 1986.
8) H.D. Vinod and A. Ullah. Recent Advances in Regression Methods. *Marcel Dekker,INC.*, 1981.