# Kernel Partial Least Squares Based On Least Squares Support Vector Machine Primal Dual Optimization Problems

HuiGuo LingWang HePing_Liu

Information Engineering School, University of Science and Technology Beijing 100083,China

Email: paul_gh@sina.com linda_gh@sina.com liuheping@sina.com

## Abstract

Partial Least Squares (PLS) and its kernel version (KPLS) have become competitive regression approaches. KPLS performs as well as or better than support vector regression (SVR) for moderately sized problems with the advantages of simple implementation, less training cost, and easier tuning of parameters. As a result, we present a simple and straightforward least square support vector machine formulation to the problem of kernel Partial Least Squares (KPLS). In the paper a least squares support vector machine style deduction is given with a primal-dual optimization problem formulation for the kernel partial least squares. Finally, the model is illustrated on some examples. This shows that the method proposed is effective and superior.
**Keyword**: Kernel methods, least squares-support vector machine , KPLS, RKHS

## I. Introduction

Over the last years one can see many learning algorithms being transferred to a kernel Representation[1]. The benefit lies in the fact that nonlinearity can be allowed and be avoided to solve a nonlinear optimization problem. Support vector machines (SVMs) as originally introduced by Vapnik within the area of statistical learning theory and structural risk minimization have been proven working successfully on many applications of nonlinear classification and function estimation[3]. The problems are formulated as convex optimization problems, usually quadratic programs, for which the dual problem is solved. Least Squares Support Vector Machines (LS-SVMs)[2] are reformulations to standard SVMs which lead to solving linear systems for classification tasks as well as regression. Within the models and the formulation one makes use of the kernel trick which is based on the Mercer theorem related to positive definite kernels[4]. One can plug in any positive definite kernel for a support vector machine classifier or regressor with as typical choices linear, polynomial and RBF kernels.

The work on SVMs has also stimulated the research on kernel-based learning methods in general in recent years. The conceptual idea of generalizing an existing linear technique to a nonlinear version by applying the kernel trick has become an area of active research. In this paper we focus on least squares regression models in the kernel context. By means of a nonlinear map into a Reproducing Kernel Hilbert Space (RKHS)[7,13] the data are projected to a high-dimensional space. Kernel methods typically operate in this RKHS. The high-dimensionality which cause problems with proper parameter estimation is circumvented by the kernel trick, which brings the dimensionality to

* Supported by key discipline construction program of Beijing Municipal commission of education

the number of training instances n and at the same time allows excellent performance in classification and regression tasks. Yet, for large datasets this dimensionality in n means a serious bottleneck. Therefore downsizing the system in dimensions to size $m \leq n$ is needed.

A nonlinear kernel function is used to map the data into a feature space in which linear partial least squares regression (PLS) techniques can not be performed. This principle of using nonlinear kernel functions to construct nonlinear variants of linear techniques is commonly used in the field of machine learning (*Schölkopf* and *Smola*, 2002)[1]. Using a least squares support vector machine (LS-SVM) approach instead of just the kernel trick with application of Mercer's theorem, an appropriate form of regularization can be incorporated within KPLS.

Primal-dual optimization formulations of this regularized KCCA have been proposed by Suykens et al. (2002). The resulting problem to be solved in the dual space is a generalized eigenvalue problem that corresponds to the formulation of a least squares problem in the primal space involving a feature map.

As a result, this paper shows an extension of LS-SVM formulations to the area of unsupervised learning. The LS-SVM approach is closely related to regularization networks, Gaussian processes, kernel ridge regression and reproducing kernel Hilbert spaces (RKHS)[5,6,9,13]. The formulation is in the style of LS-SVMs, in the sense that one starts from a constrained optimization problem in primal weight space with incorporation of a regularization term and one solves the dual problem after application of the kernel trick. The nonlinear version of the formulation yields a solution which is equivalent to kernel PLS. On the other hand, the LS-SVM formulations are closer related to standard SVMs with explicit primal-dual interpretations from the viewpoint of optimization theory.

This paper is organized as follows. In Section II we present some minimal background on kernel methods in relation to reproducing kernel Hilbert spaces. In Section III we deal with KPLS in its primal-dual optimization formulation and formulate its sparse kernel version. In Section IV we illustrate the sparse algorithm on a large data set application. We conclude the paper in Section V.

## II. Kernel Partial Least Squares Regression in RKHS

### A. *Partial Least Squares*

Partial Least Squares (PLS)[9,14] is a multivariate technique that delivers an optimal basis in x-space for y onto x regression. Reduction to a certain subset of the basis introduces a bias, but reduces the variance. In general, PLS is based on a maximization of the covariance between successive linear combinations in x and y space, $(v, x)$ and $(w, y)$, where coefficient vectors: v and w are normed to unity and constrained to be orthogonal in x space:

$$\max_{v,w} \operatorname{cov}(v^T x, w^T y) = v^T C_{xy} w \qquad (1)$$

subject to $\|v\| = 1 = \|w\|$ and $v^T v = I_p$, with $C_{xy} = X^T Y$ the sample covariance matrix. Solutions can be obtained by using Lagrange multipliers, which leads to solving the following system[8]

$$C_{xy} w = \lambda v \qquad (2)$$

$$C_{yx} v = \lambda w \qquad (3)$$

As a least squares cost function, PLS turns out to be a sum of the least squares formulation of each of the above methods:

$$J(v, w) = \sum_{i=1}^{n} \left\| x_i - v v^T x_i \right\|^2 + \left\| v^T x_i - w^T y_i \right\|^2 + \left\| y_i - w w^T y_i \right\|^2 \qquad (4)$$

subject to the constraints.

## *B. Kernel Partial Least Squares*

Consider a general setting of the linear PLS algorithm to model the relation between two data sets. Denote by $x \in X \subset R^n$ a $N$ -dimensional vector of variables in the first block of data and similarly $y \in Y \subset R^n$ denotes a vector of variables from the second set. Observing n data samples from each block of variables, PLS decomposes the $n \times N$ matrix of zero mean variables X and the $n \times M$ matrix of zero mean variables Y into the form[11]

$$\begin{aligned} X &= TP^T + F \\ Y &= UQ^T + G \end{aligned} \tag{5}$$

Where T, U are $n \times p$ matrices of the extracted p score vectors and P, Q, F and G are the matrices of residuals. The PLS method, which in its classical form is based on the nonlinear iterative partial least squares (NIPALS) algorithm[8], finds weight vectors w; c such that

$$[\mathrm{cov}(t,u)]^2 = [\mathrm{cov}(Xw, Yc)]^2 = \max_{|r|=|s|=1} [\mathrm{cov}(x_r, Y_s)]^2$$

Where $\mathrm{cov}(t,u) = t^T u/n$ denotes the sample covariance between the score vectors t and u. It can be shown that the weight vector w also corresponds to the first eigenvector of the following eigenvalue problem

$$X^T Y Y^T X w = \lambda w \tag{6}$$

The X-scores t are then given as

$$t = Xw \tag{7}$$

The kernel PLS method is based on mapping the original input data into a high-dimensional feature space F. In this case the vectors w and c cannot be usually computed. Alternatively, the score vectors t can be directly estimated as the first eigenvector of the following eigenvalue problem[10,11]

$$X X^T Y Y^T t = \lambda t \tag{8}$$

The Y-scores t are estimated as

$$u = Y Y^T t \tag{9}$$

Now, consider a nonlinear transformation of x into a feature space $F$ Denote $\Phi$ as the $(n \times s)$ matrix of mapped X-space data $\phi(x)$ into an S-dimensional feature space $F$ . Instead of an explicit mapping of the data, property (2) can be used resulting in

$$k = \phi \phi^T \tag{10}$$

Where K represents the $(n \times n)$ kernel Gram matrix of the cross dot products. $k(\cdot,\cdot)$ is a selected kernel function. Similarly, consider a mapping of the second set of variables y into a feature space $F_1$ and denote by $\varphi$ the $(n \times s_1)$ matrix of mapped Y-space data $\varphi(y)$ into an S1-dimensional feature space $F_1$. Define the $(n \times n)$ kernel Gram matrix $k_1$

$$k_1 = \varphi \varphi^T \tag{11}$$

Using this notation the estimates of t and u can be reformulated into its nonlinear kernel variant

$$\begin{aligned} k k_1 t &= \lambda t \\ u &= k_1 t \end{aligned} \tag{12}$$

Similar to linear PLS, a zero mean nonlinear kernel PLS model is assumed. To centralize the mapped data in a feature space F the following procedure must be Applied[8]

$$k \leftarrow (I_n - \frac{1}{n} I_n I_n^T) k (I_n - \frac{1}{n} I_n I_n^T) \tag{13}$$

Where $I_n$ is an n-dimensional identity matrix and $I_n$ represents a $(n \times 1)$ vector with elements equal to one. The same is true for $k_1$.

## III. An LS-SVM Approach To Kernel PLS

A more principled approach can be taken by starting from the PLS least squares formulation (4). This primal cost criterion aims at optimizing the coefficient vectors $v$ and $w$, searching simultaneously for the maximal projection of a data point in x space, maximal covariation with the corresponding projection of the point in y space, and maximal projection of a data point in y space. Since the coefficients could become arbitrarily large, these are typically constrained or at least regularized. By simplifying expression (4) we obtain $(\mathrm{v}^{\mathrm{T}}\mathrm{x}, \mathrm{w}^{\mathrm{T}}\mathrm{y})$ and by adding (soft) regularization, we arrive at the following primal form problem:

$$\max J_{PLS}(v,w,e,r) = \gamma \sum_{i=1}^{n} e_i r_i - \frac{1}{2} v^T v - \frac{1}{2} w^T w \qquad (14)$$

such that $\mathrm{e_i} = \mathrm{w}^{\mathrm{T}}\phi(\mathrm{x_i})$ and $\mathrm{r_i} = \mathrm{v}^{\mathrm{T}}\phi(\mathrm{y_i})$, $\mathrm{k} = 1, \cdots, \mathrm{n}$. Introducing $\alpha_i$, $\beta_i$ as Lagrange multiplier parameters, the Lagrangian is written as:

$$L(v,w,e,r,\alpha,\beta) = \gamma \sum_{i=1}^{n} e_i r_i - \frac{1}{2} v^T v - \frac{1}{2} w^T w - \sum_{i=1}^{n} \alpha_i (e_i - v^T \phi_1(x_i)) - \sum_{i=1}^{n} \beta_i (r_i - w^T(\phi(y_i)) \quad (15)$$

A given optimization problem has a corresponding dual formulation. The number of constraints in the original problem becomes the number of variables in the dual problem. One optimizes the Lagrangian subject to the following optimality conditions:

$$\frac{\partial \mathrm{L}}{\partial \mathrm{v}} = 0 \rightarrow \mathrm{v} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \alpha_{\mathrm{i}} \phi(\mathrm{x_i}) \qquad (16)$$

$$\frac{\partial \mathrm{L}}{\partial \mathrm{w}} = 0 \rightarrow \mathrm{w} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \beta_{\mathrm{i}} \phi(\mathrm{y_i}) \qquad (17)$$

$$\frac{\partial L}{\partial e_i} = 0 \rightarrow \alpha_i = \gamma r_i \quad i = 1, \cdots, n \qquad (18)$$

$$\frac{\partial L}{\partial r_i} = 0 \rightarrow \beta_i = \gamma e_i \quad i = 1, \cdots, n \qquad (19)$$

$$\frac{\partial L}{\partial \alpha_i} = 0 \rightarrow e_i = v^T \phi(x_i) \quad i = 1, \cdots, n \qquad (20)$$

$$\frac{\partial L}{\partial \beta_i} = 0 \rightarrow r_i = w^T \phi(y_i) \quad i = 1, \cdots, n \qquad (21)$$

By elimination of the variables $e, r, v, w$, and defining $\lambda = \frac{1}{\gamma}$, we can simplify the dual problem further:

$$\begin{bmatrix} 0 & \Omega_{c,2} \\ \Omega_{c,1} & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \lambda \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \qquad (22)$$

One has the following elements for the centered kernel matrix:

$$\Omega_{c,1_{i,j}} = \phi(x_i)^T \phi(x_j)$$
$$\Omega_{c,2_{i,j}} = \phi(y_i)^T \phi(y_j) \qquad (23)$$

are the elements of the centered Gram matrices for $i, j = 1, \cdots, n$. As such it only remains to choose a reduced set of feature vectors to induce the sparse kernel expansion. This gives analogously rise to an expression but with the KPLS based eigenvectors instead:

$$f(x) = (\omega^{(s)})^T \phi(x) + b$$
$$= \sum_{j=1}^{s} \omega_j \left( \sum_{i=1}^{m} \alpha_{i,j}^{(m)} k(x_{i,} x) \right) + b$$

$$= \sum_{i=1}^{m} \alpha_i k(x_i, x) + b \qquad (24)$$

where $s$ is the number of retained principal components and $\alpha_i = \sum_{j=1}^{s} \omega_j \alpha_{ij}^{(m)}$

## IV. Experiments Results

### *A. Mackey-Glass chaotic time series*

Our first experiment is with the Mackey-Glass chaotic time series. It is often used in practice as a benchmark set because of its nonlinear chaotic characteristics. Chaotic time series do not converge or diverge in time and their trajectories are highly sensitive to initial conditions. This time series may be generated by numerical integration of a time-delay differential equation:

$$\frac{dx(t)}{dt} = -bx(t) + \frac{ax(t-t_d)}{1+x^{10}(t-t_d)} \quad for \quad t_d > \tau \qquad (25)$$

Where $a = 0.2$, $b = 0.1$. For $\tau >$ 16.8 the dynamics become chaotic. We therefore conduct our tests using two values for $\tau$, corresponding to weakly chaotic behavior at $\tau$ = 17 and a more difficult case at $\tau$ = 30. Eq.(16) is numerically integrated using the Euler method and uniformly distributed initial conditions $x_0 \in [0.1,2]$ and $x_t = 0$ for $t < 0$.

The goal of this task is to use known values of the time series up to the point $x = t$ to predict the value at some point in the future $x = t + \tau$. The training data partitions were constructed by moving a "sliding windows" over the 3000 training samples in steps of 500 samples. This window had two size-500 samples and 1000 samples, respectively. In Figure 1, we get mean squared error based on the number of principal components used. We can see that mean squared error of primal-dual optimization KPLS is lower than the mean squared error of KPLS.

Here, MSE= $\sqrt{\frac{1}{k}\sum_{i=1}^{k}(x_i - \hat{x}_i)^2}$ , $x_i$ is true output, $\hat{x}_i$ is estimate output, k is the number of the test data.

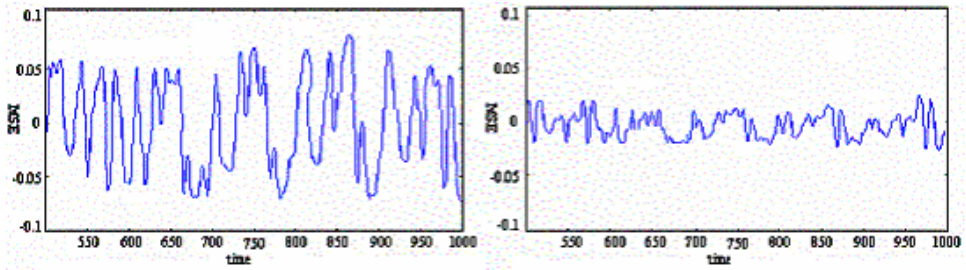


Figure 1: The MSE based on the number of principal components used. Left: The average results of KPLS. Right: The average results of primal-dual optimization KPLS

### *B. Sinc function approximation*

To demonstrate some characteristics of this kind of kernel framework methods, we applied a large scale data set to the sinc function for the noisy case (Gaussian noise with standard.0.5). The sinc(x) function is defined as

$$f(x) = sinc(x) = \frac{sin|x|}{|x|}$$

Given N=45222 training data point. Here we use the kernel $\exp(-\|x_i - x_j\|^2 / \sigma^2)$. The values of $\delta = 16.2, \gamma = 10$ are determined from the cross-validation. After cross-validation the best model was evaluated on the independent test set. Training and evaluation time per model is typically of the order of minutes for a modest number of components. The corresponding output values were centralized. We picked at random a training set of size $n = 33000$ and a test set size $t = 12222$.

In Fig.2 a typical picture of the first three components qualitatively show a good correlation with the targets. We can see the first three components extracted by primal-dual optimization kernel PLS. In the next step we added the white Gaussian noise with standard deviation 0.2 to the outputs.

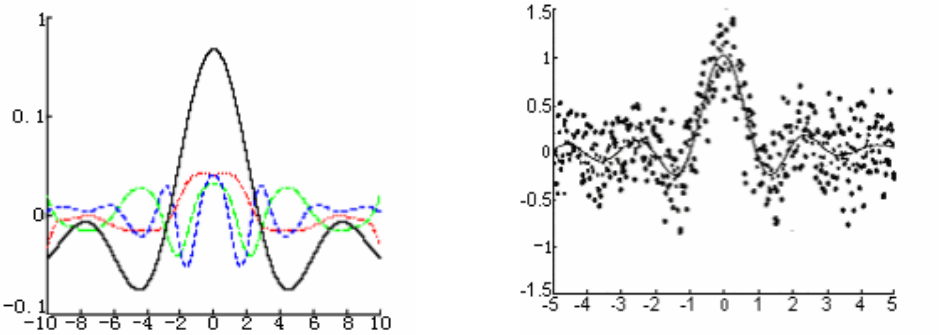

Figure2:(left) First three components extracted by primal-dual optimization kernel PLS. (right) primal-dual optimization kernel PLS on noisy sinc(x) function. Sinc(x) function is shown as a solid line.

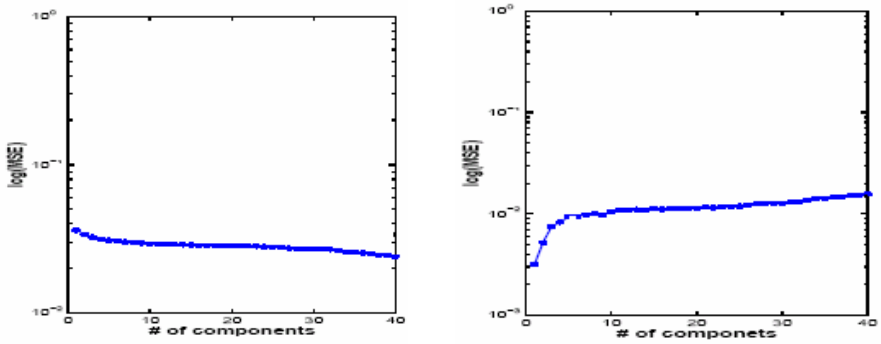


Figure 3 Dependence of the training and testing error of primal-dual optimization kernel PLS on the number of extracted components. Error is evaluated in terms of mean squared error (MSE). (left) training set .(right) testing set.

The results obtained on training and testing parts of the data are depicted in Figure 3. We can see mean squared error of training and testing set of primal-dual optimization kernel PLS. The figure shows that the MSE of testing set increases as the components gradually augment. So the primal dual optimization kernel PLS method fits more precisely noisy training data by the appropriate selection of the components.

## V. Conclusion

The use of a mapping to a high dimensional feature space leads to the kernel PLS. A new least squares support vector machine style formulation has been given to KPLS. It operates in primal space and has an important advantage: a small number of regression coefficients. In various example, we have shown that the new formulation is effectively capable of dealing with some datasets.

## References

[1] *B.Schölkopf* and A.Smola, Learning with kernels. MIT Press, 2002.

[2] J. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, Least Squares Support Vector Machines. World Scientific, Singapore, 2002.

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

[4] G. Baudat and F. Anouar, "Generalized discriminant analysis using a kernel approach," *Neural Comput.*, vol. 12, 2000. pp. 2385–2404.

[5] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller, "Fisher discriminant analysis with kernels," in Neural Networks for Signal Processing *IX*, Y.H. Hu, J. Larsen, E. Wilson, and S. Douglas, Eds. Piscataway, NJ: IEEE Press, 1999, pp. 41–48.

[6] C. Williams, C. Rasmussen, A. Schwaighofer, and V. Tresp, "Observations on the Nystrom

method for Gaussian processes," Institute for Adaptive and Neural Computation, Division of Informatics, University of Edinburgh, Tech. Rep., 2002.
[7] S. Fine and K. Scheinberg, "Efficient SVM training using low-rank kernel representations," *Journal of Machine Learning Research*, vol. 2, no. 2, 2002, pp.243–264
[8] J.A. Wegelin. A survey of Partial Least Squares (PLS) methods, with emphasis on the two-block case. Technical report, Department of Statistics, University of washington, Seattle, 2000.
[9] H. Wold, "Estimation of principal components and related models by iterative least squares," in *Multivariate Analysis, ed. P.R. Krishnaiah*. New York: Academic Press, 1966,pp. 91–420
[10] A. Äoskuldsson. PLS Regression Methods. Journal of Chemometrics,1998, 2:211-228.
[11] S.Äannar, F. Lindgren, P. Geladi, and S. Wold. A PLS kernel algorithm for data sets with many variables and fewer objects. Chemometrics and Intelligent Laboratory Systems, 1994. 8:111-125.
[12] Bennett and M. J. Embrechts, Advances in Learning Theory: Methods,Models and Applications, J.A.K. Suykens, G. *Horvat*h, S. Basu, C.Micchell*i*, J. Vandewalle *(*Eds.*)*. NATO Science Series III: Computer & Systems Sciences,IOS Press Amsterdam, 2003, vol.190, pp. 227–250.
[13] C. K. I. Williams and C. E. Rasmussen, "Gaussian processes for regression," in Advances in Neural Information Processing Systems, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds. Cambridge, MA:MIT Press, 1996, vol. 8, pp. 514–520.
[14] E.C. Malthouse, A.C. Tamhane, and R.S.H. Mah. Nonlinear partial least squares. Computers in Chemical Engineering, 1997, 21(8):875-890.

Hui Guo: Ph.D. candidate in the Institute of Information Engineering at Beijing the university of Science and Technology. His research interests include machine learning and data mining.

Ling Wang: Ph.D. candidate in the Institute of Information Engineering at Beiing university of Science and Technology. Her research interests include Artificial intelligent, machine learning, and data mining.

HePing_Liu: Professor at Beijing University of Science and Technology. His research interests include the artificial intelligent control and machine learning.