Kernel Component Analysis Using an Epsilon-Insensitive Robust Loss Function

Carlos Alzate and Johan A. K. Suykens, *Senior Member, IEEE*

***Abstract*—Kernel principal component analysis (PCA) is a technique to perform feature extraction in a high-dimensional feature space, which is nonlinearly related to the original input space. The kernel PCA formulation corresponds to an eigendecomposition of the kernel matrix: eigenvectors with large eigenvalues correspond to the principal components in the feature space. Starting from the least squares support vector machine (LS-SVM) formulation to kernel PCA, we extend it to a generalized form of kernel component analysis (KCA) with a general underlying loss function made explicit. For classical kernel PCA, the underlying loss function is L_2. In this generalized form, one can plug in also other loss functions. In the context of robust statistics, it is known that the L_2 loss function is not robust because its influence function is not bounded. Therefore, outliers can skew the solution from the desired one. Another issue with kernel PCA is the lack of sparseness: the principal components are dense expansions in terms of kernel functions. In this paper, we introduce robustness and sparseness into kernel component analysis by using an epsilon-insensitive robust loss function. We propose two different algorithms. The first method solves a set of nonlinear equations with kernel PCA as starting points. The second method uses a simplified iterative weighting procedure that leads to solving a sequence of generalized eigenvalue problems. Simulations with toy and real-life data show improvements in terms of robustness together with a sparse representation.**

***Index Terms*—Epsilon-insensitive loss function, kernel principal component analysis (PCA), least squares support vector machines (LS-SVM), loss function, robustness, sparseness.**

I. Introduction

PRINCIPAL component analysis (PCA) is a powerful unsupervised learning technique widely used for feature extraction, denoising, and compression [1]. It is an orthogonal basis transformation where the new basis is found by an eigendecomposition of the covariance matrix C of a normalized data set. PCA constructs the rank k subspace approximation to the training data that is optimal in a least squares sense, where $k < d$ and d is the input dimensionality. Although PCA is a powerful technique, it cannot detect nonlinear structures in the input data [2]. Kernel PCA, as a natural nonlinear generalization of PCA, first maps the input data into some feature space $\mathcal{F}$ via a nonlinear feature map φ induced by a kernel and then performs linear PCA on the mapped data. The projections onto the subspace spanned by the eigenvectors now lie in $\mathcal{F}$ but in many applications we need the corresponding patterns in the input space. In order to recover the extracted features or the denoised patterns, we have to map from $\mathcal{F}$ back to the input space (also called the preimage problem [3]–[6]). In [7], it was shown that kernel PCA can be derived as the dual problem to a primal optimization problem formulated in a kernel-induced feature space. With this formulation, the underlying loss function associated to classical kernel PCA was shown to be L_2. In the context of robust statistics, it is known that the L_2 loss function is not robust because its influence function is not bounded [8], [9], hence outliers can arbitrarily deviate the solution from the desired one. Even though the approach based on influence functions was developed mainly for regression, an explicit formulation for the PCA influence function was derived in [10], showing that it is unbounded and confirming its sensitivity to outliers and therefore its nonrobust nature. In [11] a statistical physics approach to PCA was studied in order to obtain robust principal components. The resulting formulation led to a bounded influence function as it was demonstrated in [12]. Another issue with kernel PCA is the lack of sparseness because the principal components are expressed in terms of a dense expansion of kernels associated with every training data point. Some approaches have been developed to introduce sparseness into kernel PCA; see, e.g., [4] and [13].

In this paper, we derive a generalized formulation of kernel component analysis (KCA) where the loss function associated to kernel PCA is explicit and can easily be changed. By incorporating an epsilon-insensitive zone into a robust loss function, it is possible to impose robustness and sparseness into kernel PCA. Two different algorithms are proposed. The first one solves a set of nonlinear equations using, e.g., a sequential quadratic programming (SQP) method with classical kernel PCA eigenvalues/vectors as starting points. The second proposed algorithm is a simplified weighting scheme that results in solving a sequence of generalized eigenvalue problems. With this scheme, every data point in the training set has an associated weight, which determines its importance in the component calculation. We also propose a model selection method on validation data to find the kernel parameters and the number of components.

Manuscript received August 10, 2007; revised January 21, 2008; accepted February 29, 2008. First published July 18, 2008; current version published September 4, 2008. This work was supported by the Research Council K.U.Leuven under Grants GOA-Mefisto 666 and GOA-Ambiorics, and several Ph.D./Postdoctoral and Fellow Grants, the Flemish Government FWO under Ph.D./Postdoctoral Grants, Projects G.0240.99, G.0211.05, G.0407.02, G.0197.02, G.0080.01, G.0141.03, G.0491.03, G.0120.03, G.0452.04, G.0499.04, G.0226.06, G.0302.07, ICCoS, ANMMM;AWI;IWT:PhD grants, GBOU (McKnow) Soft4s, the Belgian Federal Government (Belgian Federal Science Policy Office: IUAP V-22; PODO-II (CP/01/40), the European Union (FP5-Quprodis, ERNSI, Eureka 2063-Impact;Eureka 2419-FLiTE) and Contracts Research/Agreements (ISMC/IPCOS, Data4s, TML, Elia, LMS, IPCOS, Mastercard). The scientific responsibility is assumed by its authors.

The authors are with the Department of Electrical Engineering ESAT-SCD-SISTA, Katholieke Universiteit Leuven, B-3001 Leuven, Belgium (e-mail: carlos.alzate@esat.kuleuven.be; johan.suykens@esat.kuleuven.be).

Digital Object Identifier 10.1109/TNN.2008.2000443

1045-9227/$25.00 © 2008 IEEE

This tuning method looks for parameters that maximize a criterion based on the proposed loss function. A heuristic to obtain good choices of the proposed loss function parameters is also proposed.

This paper is organized as follows. Section II describes the links between least squares support vector machines (LS-SVMs) and kernel PCA. In Section III, we propose two approaches to extend the kernel PCA formulation to other loss functions. Section IV discusses robustness and sparseness in relation to the chosen loss function. In Section V, we review three preimage algorithms that can be combined with the new method. In Section VI, we propose two algorithms to find components with a general loss function. Section VII describes a model selection method. In Section VIII, we report the empirical results, and in Section IX, we give conclusions.

II. LS-SVMs and Kernel PCA

LS-SVM formulations to different problems were discussed by Suykens *et al.* [14]. It emphasizes primal-dual interpretations for a class of kernel machines in terms of constrained optimization problems. LS-SVMs have been studied with respect to kernel Fisher discriminant analysis, kernel ridge regression, kernel PCA, kernel canonical correlation analysis, kernel partial least squares, spectral clustering, recurrent networks, control, and others [7], [14], [15]. In this section, we discuss LS-SVM formulations to kernel PCA, which are relevant for the sequel of this paper.

A. Classical PCA Formulation

In the classical PCA formulation, one considers a given set of zero mean and normalized data $\{x_i\}_{i=1}^N$ with $x_i \in \mathbb{R}^d$. The objective is to find projected variables $w^T x$ with maximal variance

$$\max_w \mathrm{Var}(w^T x) \simeq \frac{1}{N}\sum_{i=1}^N (w^T x_i)^2 = w^T C w \qquad (1)$$

such that $w^T w = 1$, where $C = (1/N)\sum_{i=1}^N x_i x_i^T$ is the sample covariance matrix. The Lagrangian for this constrained optimization problem is given by

$$\mathcal{L}(w;\lambda) = \frac{1}{2} w^T C w - \lambda(w^T w - 1) \qquad (2)$$

where λ is a Lagrange multiplier. From the conditions for optimality $\partial\mathcal{L}/\partial w = 0$ and $\partial\mathcal{L}/\partial\lambda = 0$, one obtains the eigenvalue problem

$$Cw = \lambda w. \qquad (3)$$

The eigenvector w corresponding to the largest eigenvalue defines the direction in which the projected variables have maximal variance. Subsequent eigenvectors and eigenvalues are usually found by deflating the data matrix or by adding orthogonality constraints with respect to the previously found eigenvectors [1].

B. LS-SVM Approach to Kernel PCA

1) Unconstrained Formulation: Consider the following unconstrained optimization problem:

$$\min_w J(w) = \frac{\gamma}{2N}\sum_{i=1}^N \left(w^T(\varphi(x_i) - \hat{\mu}_\varphi)\right)^2 - \frac{1}{2} w^T w \qquad (4)$$

where γ is a positive regularization constant, $\hat{\mu}_\varphi = (1/N)\sum_{j=1}^N \varphi(x_j)$, and $\varphi(\cdot) : \mathbb{R}^d \to \mathbb{R}^{n_h}$ is the mapping to a high-dimensional feature space $\mathcal{F}$ of dimension n_h (which can be infinite dimensional).

Proposition 1: Each eigenvector $w^\star$ with corresponding eigenvalue ρ of the sample covariance matrix in the feature space $C_\varphi = (1/N)\sum_{j=1}^N (\varphi(x_j) - \hat{\mu}_\varphi)(\varphi(x_j) - \hat{\mu}_\varphi)^T$ is a stationary point of $J(w)$ given that $\gamma = 1/\rho$. The objective function $J(w)$ evaluated at any eigenvector $w^\star$ equals zero.

Proof: The optimization problem in (4) can be written as

$$\min_w J(w) = \frac{\gamma}{2} w^T C_\varphi w - \frac{1}{2} w^T w = \frac{1}{2} w^T(\gamma C_\varphi - I_h) w.$$

Using the first-order optimality condition $\partial J(w)/\partial w = 0$ leads to

$$C_\varphi w^\star = \rho w^\star \qquad (5)$$

which is an eigenvalue problem of C_φ with $\rho = 1/\gamma$ and corresponding eigenvector $w^\star$ such that $(w^\star)^T w^\star = 1$. The objective function becomes

$$\begin{aligned} J(w^\star) &= \frac{\gamma}{2}(w^\star)^T \rho w^\star - \frac{1}{2}(w^\star)^T w^\star \\ &= \left(\frac{\gamma\rho}{2} - \frac{1}{2}\right)(w^\star)^T w^\star \\ &= 0. \end{aligned}$$

■

Remark 1: Note that solving (5) becomes intractable for high-dimensional feature spaces and the size of the sample covariance matrix C_φ may become too large.

2) Constrained Formulation: Now consider the following constrained optimization problem obtained by introducing additional variables e_i into (4):

$$\begin{aligned} \min_{w,e} J_p(w,e) &= \frac{\gamma}{2N}\sum_{i=1}^N e_i^2 - \frac{1}{2} w^T w \\ \text{such that} \quad & e_i = w^T(\varphi(x_i) - \hat{\mu}_\varphi), \qquad i = 1,\ldots,N. \end{aligned} \qquad (6)$$

Lemma 1: Given a positive definite kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ with $K(x,z) = \varphi(x)^T\varphi(z)$ and a regularization constant $\gamma \in \mathbb{R}^+$, the Karush–Kuhn–Tucker (KKT) conditions of the Lagrangian of (6) are satisfied by the eigenvectors $\alpha^\star$ of the following dual eigenvalue problem:

$$\Omega_c \alpha = \lambda\alpha \qquad (7)$$

where Ω_c denotes the centered kernel matrix with ijth entry: $\Omega_{c,ij} = K(x_i,x_j) - (1/N)\sum_{r=1}^N K(x_i,x_r) -$

$(1/N)\sum_{r=1}^{N} K(x_j, x_r) + (1/N^2)\sum_{r=1}^{N}\sum_{s=1}^{N} K(x_r, x_s)$ and $\lambda = N/\gamma$. The dual objective function $\mathcal{L}(\alpha) = (1/2)\alpha^T(\Omega_c - \lambda I_N)\alpha$ evaluated at any eigenvector $\alpha^\star$ equals zero.

Proof: Consider the Lagrangian of the problem (6)

$$\mathcal{L}(w, e_i; \alpha) = \frac{\gamma}{2N}\sum_{i=1}^{N} e_i^2 - \frac{1}{2}w^T w - \sum_{i=1}^{N}\alpha_i\left(e_i - w^T\left(\varphi(x_i) - \hat{\mu}_\varphi\right)\right)$$

with KKT optimality conditions

$$\begin{cases} \dfrac{\partial \mathcal{L}}{\partial w} = 0 \rightarrow w = \sum_{i=1}^{N}\alpha_i\left(\varphi(x_i) - \hat{\mu}_\varphi\right) \\ \dfrac{\partial \mathcal{L}}{\partial e_i} = 0 \rightarrow e_i = \frac{N}{\gamma}\alpha_i, \qquad i = 1, \ldots, N \\ \dfrac{\partial \mathcal{L}}{\partial \alpha_i} = 0 \rightarrow e_i = w^T\left(\varphi(x_i) - \hat{\mu}_\varphi\right), \qquad i = 1, \ldots, N. \end{cases}$$

Eliminating w and e_i leads to

$$\frac{N}{\gamma}\alpha_i = \sum_{l=1}^{N}\alpha_l\left(\varphi(x_l) - \hat{\mu}_\varphi\right)^T\left(\varphi(x_i) - \hat{\mu}_\varphi\right), \qquad i = 1, \ldots, N.$$

By making use of the positive–definite kernel function

$$\begin{aligned} &\left(\varphi(x_i) - \hat{\mu}_\varphi\right)^T\left(\varphi(x_j) - \hat{\mu}_\varphi\right) \\ &\quad = K(x_i, x_j) - \frac{1}{N}\sum_{r=1}^{N} K(x_i, x_r) - \frac{1}{N}\sum_{r=1}^{N} K(x_j, x_r) \\ &\qquad + \frac{1}{N^2}\sum_{r=1}^{N}\sum_{s=1}^{N} K(x_r, x_s) \\ &\quad = \Omega_{c,ij} \end{aligned}$$

and defining $\lambda = N/\gamma$, $\alpha = [\alpha_1, \ldots, \alpha_N]$ leads to the eigenvalue problem (7).

Expressing the Lagrangian in terms of the dual variables α leads to the dual objective function

$$\mathcal{L}(\alpha) = \frac{1}{2}\alpha^T(\Omega_c - \lambda I_N)\alpha.$$

Given that $\alpha^\star$ is an eigenvector of Ω_c with eigenvalue λ, the dual objective function becomes $\mathcal{L}(\alpha^\star) = 1/2(\lambda - \lambda) = 0$. ■

Note that every eigenvalue/eigenvector pair of Ω_c is a stationary point of (6) with objective function value equal to zero. Therefore, the principal components can be seen as a pool of candidate solutions. The component selection problem from this pool can be formulated as

$$\max_{j \in \{1, \ldots, N\}} \frac{\gamma_j}{2N}\sum_{i=1}^{N}\left(e_i^{(j)}\right)^2 = \frac{N}{2\gamma_j}\sum_{i=1}^{N}\left(\alpha_i^{(j)}\right)^2 = \frac{\lambda_j}{2} \quad (8)$$

which is equivalent to sorting the components in decreasing order with respect to the eigenvalue λ_j. This differs from classical PCA where the objective function value at the optima is the eigenvalue and reflects the amount of variance captured by the projection onto the corresponding eigenvector.

Remark 2: Note that changing the minimization operator to maximization in (6) leads to the same set of stationary points (7).

Remark 3: Note that the dual solutions (7) correspond to kernel PCA as proposed by Schölkopf *et al.* [2], where Suykens *et al.* [7] showed that a similar form of (6) is the underlying optimization problem to which kernel PCA is the dual. Equation (7) is regularized by restricting the eigenvectors to the unit ball while regularization is explicitly present on the other hand in the primal problem in (6) through the regularization term.

Remark 4: Considering a bias term in (6) leads to centering the kernel matrix Ω in the feature space such that the mapped data have zero mean; see [7]. The bias term becomes

$$b = -\frac{1}{N}1_N^T\Omega\alpha. \quad (9)$$

Remark 5: The projections of a data point x onto an eigenvector (also called the score variables) become

$$\begin{aligned} z(x) &= w^T(\varphi(x) - \hat{\mu}_\varphi) \\ &= \sum_{l=1}^{N}\alpha_l\left(K(x_l, x) - \frac{1}{N}\sum_{r=1}^{N} K(x_r, x)\right. \\ &\qquad \left. - \frac{1}{N}\sum_{r=1}^{N} K(x_r, x_l) + \frac{1}{N^2}\sum_{r=1}^{N}\sum_{s=1}^{N} K(x_r, x_s)\right). \end{aligned} \quad (10)$$

III. Extension to Other Loss Functions

A. Set of Nonlinear Equations

In (6), the kernel PCA uses the L_2 loss function. By extending the formulation now to a general loss function $L : \mathbb{R} \rightarrow \mathbb{R}$ and introducing a bias term b, the following problem can be formulated:

$$\min_{w, e_i, b} J_p(w, e_i, b) = \frac{\gamma}{2}\sum_{i=1}^{N} L(e_i) - \frac{1}{2}w^T w \quad \text{such that} \quad e_i = w^T\varphi(x_i) + b, \qquad i = 1, \ldots, N. \quad (11)$$

Lemma 2: Given a positive–definite kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ with $K(x, z) = \varphi(x)^T\varphi(z) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$, a differentiable loss function $L : \mathbb{R} \rightarrow \mathbb{R}$, and positive regularization constant γ, the stationary solutions to the problem (11) can be obtained as the solution to the following set of nonlinear equations with unknowns $\alpha_i, i = 1, \ldots, N$ (dual problem) and b:

$$\begin{cases} \alpha_i - \frac{\gamma}{2}L'\left(\sum_{l=1}^{N}\alpha_l K(x_l, x_i) + b\right) = 0, \qquad i = 1, \ldots, N \\ \sum_{i=1}^{N} L'\left(\sum_{l=1}^{N}\alpha_l K(x_l, x_i) + b\right) = 0. \end{cases} \quad (12)$$

Proof: Consider the Lagrangian of the problem (6)

$$\mathcal{L}(w, e_i, b; \alpha) = \frac{\gamma}{2}\sum_{i=1}^{N} L(e_i) - \frac{1}{2}w^T w - \sum_{i=1}^{N}\alpha_i\left(e_i - w^T\varphi(x_i) - b\right) \quad (13)$$

with KKT optimality conditions

$$\begin{cases} \dfrac{\partial \mathcal{L}}{\partial w} = 0 \rightarrow w = \sum\limits_{i=1}^{N} \alpha_i \varphi(x_i) \\ \dfrac{\partial \mathcal{L}}{\partial e_i} = 0 \rightarrow \alpha_i = \frac{\gamma}{2} L'(e_i), \qquad i = 1, \ldots N \\ \dfrac{\partial \mathcal{L}}{\partial b} = 0 \rightarrow \sum\limits_{i=1}^{N} \alpha_i = 0 \\ \dfrac{\partial \mathcal{L}}{\partial \alpha_i} = 0 \rightarrow e_i = w^T \varphi(x_i) + b, \qquad i = 1, \ldots N. \end{cases} \tag{14}$$

Eliminating w, e_i leads to

$$\begin{cases} \frac{\gamma}{2} L' \left(\sum\limits_{l=1}^{N} \alpha_l \varphi(x_l)^T \varphi(x_i) + b \right) = \alpha_i, \qquad i = 1, \ldots, N, \\ \sum\limits_{i=1}^{N} L' \left(\sum\limits_{l=1}^{N} \alpha_l \varphi(x_l)^T \varphi(x_i) + b \right) = 0 \end{cases}$$

and applying the definition of a positive–definite kernel function gives the set of nonlinear equations (12).

The score variables become

$$z(x) = w^T \varphi(x) + b = \sum_{l=1}^{N} \alpha_l K(x_l, x) + b. \tag{15}$$

■

Remark 6: Note that, in general, (11) is a nonconvex problem, therefore the KKT optimality conditions are necessary but not sufficient.

Remark 7: Note that (12) leads to a different centering in the feature space, because the bias term is also affected by the loss function. This is important for robustness because the bias term is related to the estimation of the mean in the feature space as shown in the L_2 case.

Remark 8: Consider the minimization of the L_2 loss of the residuals

$$U^\star = \arg\min_{U} \sum_{i=1}^{N} r_i^2 \tag{16}$$

where $r_i = \|\varphi(x_i) - P_U \varphi(x_i)\|_2$, U denotes an orthogonal subspace, and $P_U \varphi(x_i)$ is the projection of the ith mapped data point onto U. Since $P_U \varphi(x_i)$ is an orthogonal projection, one can write

$$\|\varphi(x_i) - P_U \varphi(x_i)\|_2^2 = \|\varphi(x_i)\|_2^2 - \|P_U \varphi(x_i)\|_2^2$$

which leads to

$$U^\star = \arg\max_{U} \sum_{i=1}^{N} \|P_U \varphi(x_i)\|_2^2.$$

This shows that minimizing the sum of the L_2 loss of the residuals is equivalent to maximizing the sum of the L_2 loss of the projections. However, in the case of a general loss function L, this equivalence no longer holds.

Remark 9: A general loss function can be applied to the projections e but the loss function applied to the primal vector w should be L_2 in order to exploit the equivalence between a dot product in the feature space and a kernel function evaluation.

B. A Simplified Weighting Scheme

Another approach to extend kernel PCA to general loss functions is to consider a loss function of the form $L(e_i) = v_i^{(k)} e_i^2$ where $v_i^{(k)}$, $i = 1, \ldots, N$, are user-defined weights and k is the iteration step. A two-step weighting scheme was proposed in [16] to impose robustness in LS-SVM regression. In the first step, an unweighted LS-SVM is calculated. The second stage consists of associating weighting values to the error variables obtained in the first step. This scheme can also be employed in KCA as a simplified technique to obtain robust and sparse solutions. Introducing weighting factors $v_i^{(k)}$ in the formulation leads to the following primal optimization problem:

$$\min_{w, e_i} J_p(w, e_i) = \gamma \frac{1}{2} \sum_{i=1}^{N} v_i^{(k)} e_i^2 - \frac{1}{2} w^T w$$
$$\text{such that} \quad e_i = w^T \left(\varphi(x_i) - \hat{\mu}_\varphi \right), \qquad i = 1, \ldots, N. \tag{17}$$

A simple method is to make a two-step approach. In general, it can be more steps as well. A precise convergence analysis is needed in that case. However, here we are interested in a simple way to cheaply incorporate robustness and sparseness to kernel PCA. In the first step, an unweighted kernel PCA solution is obtained ($v_i^{(1)} = 1$, $i = 1, \ldots, N$). Then, the weights are calculated in the second step in such a way that $v_i^{(2)} e_i^2 = L(e_i)$. The advantage of this approach is that one extends kernel PCA to general loss functions by solving a sequence of generalized eigenvalue problems which can be solved efficiently, while in (12), one needs to solve a set of nonlinear equations which may lead to local optima and slow convergence.

The first weighted principal component is the solution to (17). The second weighted component is the solution to (17) with an additional constraint. This constraint imposes orthogonality with respect to the score variables associated to the previously found weighted component. For the third weighted component, two constraints are added to ensure orthogonality with respect to the score variables for the second and first weighted principal components. This iterative process is repeated until all required weighted components are obtained. Hence, the mth weighted component is the solution to the following primal problem with a number of $m - 1$ orthogonality constraints:

$$\min_{w, e_i} J_p(w, e_i) = \gamma \frac{1}{2} \sum_{i=1}^{N} v_i^{(k)} e_i^2 - \frac{1}{2} w^T w$$
$$\text{such that} \quad \begin{cases} e_i = w^T \left(\varphi(x_i) - \hat{\mu}_\varphi \right), \qquad i = 1, \ldots, N \\ \sum\limits_{i=1}^{N} e_i e_i^{(1)} = 0 \\ \sum\limits_{i=1}^{N} e_i e_i^{(2)} = 0 \\ \vdots \\ \sum\limits_{i=1}^{N} e_i e_i^{(m-1)} = 0 \end{cases} \tag{18}$$

where $k = 1, \ldots, N_{\text{iter}}$ is the iteration step, $e_i^{(l)}, l = 1, \ldots, m-1$, are the score variables corresponding to the lth previously

found weighted principal component, which are assumed to be given.

Lemma 3: Given a positive–definite kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ with $K(x,z) = \varphi(x)^T\varphi(z)$, score variables $e_i^{(1)}, e_i^{(2)}, \ldots, e_i^{(m-1)} \in \mathbb{R}$, $i = 1, \ldots, N$, and positive weighting factors $v_i^{(k)} \in \mathbb{R}^+$, $i = 1, \ldots, N$, the solution to the problem (18) is given by the following generalized eigenvalue problem:

$$\left[\begin{array}{c|c} \Omega_c^V & 0_{N\times(m-1)} \\ \hline 0_{N\times(m-1)}^T & 0_{(m-1)\times(m-1)} \end{array}\right]\left[\begin{array}{c}\alpha\\ \hline \beta\end{array}\right] = \lambda \left[\begin{array}{c|c} I_N & E \\ \hline E^T & D \end{array}\right]\left[\begin{array}{c}\alpha\\ \hline \beta\end{array}\right] \tag{19}$$

where $V^{(k)} = \operatorname{diag}(v_1^{(k)}, \ldots, v_N^{(k)})$, $\Omega_c^V = V^{(k)}\Omega_c$, $0_{p\times q}$ is the $p \times q$ zero matrix, $\alpha = [\alpha_1; \ldots; \alpha_N]$, $\beta = [\beta_1; \beta_2; \ldots; \beta_{m-1}]$, $\lambda = 1/\gamma$, I_N is the $N \times N$ identity matrix, and E and D are defined as follows:

$$E = \begin{bmatrix} \frac{e_1^{(1)}}{v_1^{(k)}} & \frac{e_1^{(2)}}{v_1^{(k)}} & \cdots & \frac{e_1^{(m-1)}}{v_1^{(k)}} \\ \frac{e_2^{(1)}}{v_2^{(k)}} & \frac{e_2^{(2)}}{v_2^{(k)}} & \cdots & \frac{e_2^{(m-1)}}{v_2^{(k)}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{e_N^{(1)}}{v_N^{(k)}} & \frac{e_N^{(2)}}{v_N^{(k)}} & \cdots & \frac{e_N^{(m-1)}}{v_N^{(k)}} \end{bmatrix}$$

$$D = \begin{bmatrix} \sum_i \frac{\left(e_i^{(1)}\right)^2}{v_i^{(k)}} & \sum_i \frac{e_i^{(1)}e_i^{(2)}}{v_i^{(k)}} & \cdots & \sum_i \frac{e_i^{(1)}e_i^{(m-1)}}{v_i^{(k)}} \\ \sum_i \frac{e_i^{(1)}e_i^{(2)}}{v_i^{(k)}} & \sum_i \frac{\left(e_i^{(2)}\right)^2}{v_i^{(k)}} & \cdots & \sum_i \frac{e_i^{(2)}e_i^{(m-1)}}{v_i^{(k)}} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_i \frac{e_i^{(1)}e_i^{(m-1)}}{v_i^{(k)}} & \sum_i \frac{e_i^{(2)}e_i^{(m-1)}}{v_i^{(k)}} & \cdots & \sum_i \frac{\left(e_i^{(m-1)}\right)^2}{v_i^{(k)}} \end{bmatrix}.$$

Proof: Consider the Lagrangian of the problem (18)

$$\begin{aligned}\mathcal{L}(w, e_i; \alpha_i, \beta_j) =& \gamma\frac{1}{2}\sum_{i=1}^N v_i^{(k)} e_i^2 - \frac{1}{2}w^T w \\ &- \sum_{i=1}^N \alpha_i \left(e_i - w^T\left(\varphi(x_i) - \hat{\mu}_\varphi\right)\right) \\ &- \beta_1 \sum_{i=1}^N e_i e_i^{(1)} - \beta_2 \sum_{i=1}^N e_i e_i^{(2)} - \ldots \\ &- \beta_{i-1} \sum_{i=1}^N e_i e_i^{(m-1)}\end{aligned} \tag{20}$$

with KKT optimality conditions

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 \to w = \sum_{i=1}^N \alpha_i \left(\varphi(x_i) - \hat{\mu}_\varphi\right) \\ \frac{\partial \mathcal{L}}{\partial e_i} = 0 \to e_i = \frac{1}{\gamma v_i^{(k)}}\left(\alpha_i + \beta_1 e_i^{(1)} + \ldots + \beta_{i-1} e_i^{(m-1)}\right), \quad i = 1, \ldots, N \\ \frac{\partial \mathcal{L}}{\partial \alpha_i} = 0 \to e_i = w^T\left(\varphi(x_i) - \hat{\mu}_\varphi\right), \qquad i = 1, \ldots, N \\ \frac{\partial \mathcal{L}}{\partial \beta_1} = 0 \to \sum_{i=1}^N e_i e_i^{(1)} = 0 \\ \frac{\partial \mathcal{L}}{\partial \beta_2} = 0 \to \sum_{i=1}^N e_i e_i^{(2)} = 0 \\ \quad \vdots \\ \frac{\partial \mathcal{L}}{\partial \beta_{i-1}} = 0 \to \sum_{i=1}^N e_i e_i^{(m-1)} = 0. \end{cases}$$

Using $\partial\mathcal{L}/\partial w = 0$, $\partial\mathcal{L}/\partial e_i = 0$, and $\partial\mathcal{L}/\partial\alpha_i = 0$ leads to

$$\begin{aligned}&\frac{1}{\gamma}\left(\alpha_i + \beta_1 e_i^{(1)} + \beta_2 e_i^{(2)} + \ldots + \beta_{m-1} e_i^{(m-1)}\right) \\ &\quad = v_i \sum_{j=1}^N \alpha_j \left(\varphi(x_j) - \hat{\mu}_\varphi\right)^T \left(\varphi(x_i) - \hat{\mu}_\varphi\right), \qquad i = 1, \ldots, N.\end{aligned}$$

Applying the property of a positive–definite kernel

$$\begin{aligned}&\frac{1}{\gamma}\left(\alpha_i + \beta_1 e_i^{(1)} + \beta_2 e_i^{(2)} + \ldots + \beta_{m-1} e_i^{(m-1)}\right) \\ &\qquad = v_i^{(k)} \sum_{j=1}^N \alpha_j \Omega_c(x_j, x_i), \qquad i = 1, \ldots, N.\end{aligned} \tag{21}$$

Substituting $\partial\mathcal{L}/\partial\alpha_i = 0$ into the orthogonality constraints leads to the following set of equations:

$$\begin{cases} \sum_{j=1}^N \frac{\alpha_j e_j^{(1)}}{v_j^{(k)}} + \beta_1 \sum_{j=1}^N \frac{\left(e_j^{(1)}\right)^2}{v_j^{(k)}} + \ldots \\ \quad + \beta_{m-1} \sum_{j=1}^N \frac{e_j^{(1)} e_j^{(m-1)}}{v_j^{(k)}} = 0 \\ \sum_{j=1}^N \frac{\alpha_j e_j^{(2)}}{v_j^{(k)}} + \beta_1 \sum_{j=1}^N \frac{e_j^{(1)} e_j^{(2)}}{v_j^{(k)}} + \ldots \\ \quad + \beta_{m-1} \sum_{j=1}^N \frac{e_j^{(1)} e_j^{(m-1)}}{v_j^{(k)}} = 0 \\ \quad \vdots \\ \sum_{j=1}^N \frac{\alpha_j e_j^{(m-1)}}{v_j^{(k)}} + \beta_1 \sum_{j=1}^N \frac{e_j^{(1)} e_j^{(m-1)}}{v_j^{(k)}} + \ldots \\ \quad + \beta_{m-1} \sum_{j=1}^N \frac{\left(e_j^{(m-1)}\right)^2}{v_j^{(k)}} = 0. \end{cases} \tag{22}$$

The sets of (21) and (22) give the generalized eigenvalue problem (19). ■

Lemma 4: The matrix

$$M = \left[\begin{array}{c|c} \Omega_c^V & 0_{N\times(m-1)} \\ \hline 0_{N\times(m-1)}^T & 0_{(m-1)\times(m-1)} \end{array}\right]$$

is positive semidefinite.

Proof: The matrix Ω_c^V is the product of two positive–definite matrices $V^{(k)}$ and Ω_c. Thus, by [17, Th. 7.6.3], Ω_c^V is also positive definite.

The eigendecomposition of M

$$\left[\begin{array}{c|c} \Omega_c^V & 0_{N\times(m-1)} \\ \hline 0_{N\times(m-1)}^T & 0_{(m-1)\times(m-1)} \end{array}\right]\left[\frac{x_1}{x_2}\right] = \lambda\left[\frac{x_1}{x_2}\right]$$

is equal to

$$\left[\frac{\Omega_c^V x_1}{0_{(m-1)\times 1}}\right] = \lambda\left[\frac{x_1}{x_2}\right].$$

Hence, the nonzero eigenvalues of M are identical to the eigenvalues of Ω_c^V. ∎

Corollary 1: The generalized eigenvalue problem in (19) has real eigenvalues.

Proof: Let (19) be written as $Mp = \lambda Bp$

$$B = \left[\begin{array}{c|c} I_N & E \\ \hline E^T & D \end{array}\right] \quad p = \left[\frac{\alpha}{\beta}\right]$$

with $p \in \mathbb{C}^{N+i-1}$.

Then, premultiplying by the complex conjugate p^* leads to

$$p^*Mp = \lambda p^*Bp$$

and the generalized eigenvalues become

$$\lambda = \frac{p^*Mp}{p^*Bp}.$$

Since M is positive semidefinite (Corollary 4), and B is Hermitian, the generalized eigenvalues become the ratio of two real numbers. ∎

Remark 10: Note that λ in (19) can be negative. However, only positive λ values are considered.

IV. Robustness and Sparseness

A. Robustness in PCA and Kernel PCA

It is known that classical PCA is heavily influenced by outlying observations. A single outlier can by itself determine an entire component. The subspace approximation to the training data that PCA constructs is optimal only in a least squares sense [1]. A small deviation from the assumptions produces a gross error in the principal component estimation. Although several approaches to robustify PCA have been developed [11], [18]–[21], that is not the case for kernel PCA.

In this section, we tackle the robustness problem in kernel PCA using the approach based on influence functions introduced by Hampel *et al.* [22]. However, an explicit form of the influence function is not given.

Let T_N be a target statistic based on the training data $\{x_i\}_{i=1}^N$ assuming that T_N can be expressed as a functional, say $T(\bar{F}_N)$, where $\bar{F}_N$ indicates the empirical distribution function of the data. The influence function of $T(\bar{F}_N)$ against an outlier ξ is defined as

$$IF(\xi;\bar{F}_N,T) = \lim_{t\to 0}\frac{T\left[(1-t)\bar{F}_N + t\delta_\xi\right] - T(\bar{F}_N)}{t} \quad (23)$$

where δ_ξ is the pointmass 1 distribution at ξ. If the influence function is bounded, then the influence of the functional $T(\bar{F}_N)$ by any ξ should have a reduced effect.

The ith principal component $\alpha^{(i)}$ can be expressed as a functional $\alpha^{(i)}(\bar{F}_N)$ and according to [1], [10], and [12] its influence function becomes

$$IF\left(\xi;\bar{F}_N,\alpha^{(i)}\right) = -\beta_1(\xi)\sum_{j\neq i}^{d}\beta_j(\xi)(\lambda_j-\lambda_i)^{-1}\alpha^{(j)} \quad (24)$$

where $\beta_j(\xi) = (\alpha^{(j)})^T(\xi-\hat{\mu})$, λ_j is the jth eigenvalue, $\hat{\mu}$ is the sample mean of the data, and $\alpha^{(j)}$ is the jth principal component. Note that (24) depends on all the components and is unbounded which confirms the nonrobust nature of PCA.

For *M-estimators* [8], the influence function of the functional $T(\bar{F}_N)$ is defined as

$$IF(\xi;\bar{F}_N,T) = \frac{L'(\xi)}{\int L'(\xi)d\bar{F}_N}. \quad (25)$$

In other words, the influence function of an *M-estimator* is proportional to the first derivative of the loss function (also called the score function). Several loss functions with bounded score functions have been proposed. Well-known examples are the Huber, Hampel, Welsch, and Tukey loss functions. Kernel PCA has an underlying L_2 loss function with an unbounded score function (Fig. 1). Hence, kernel PCA is not robust against outlying data points. Note also that kernel PCA is linear PCA in a kernel-induced feature space, therefore outliers in the feature space have a large influence. Christmann *et al.* showed [23] that the kernel and the score function should be bounded in convex risk minimization methods to achieve robustness.

B. Sparseness

As can be seen in (10), the score variables are expressed in terms of kernel expansions in which every training point contributes. These expansions are typically dense (nonsparse). Different methods have been proposed to overcome the lack of sparseness in kernel PCA. Smola *et al.* [13] proposed a method for obtaining sparse components employing an L_1 penalty on the expansion coefficients α_i. Schölkopf *et al.* [4] introduced a compact representation for the expansions leading to a reduced set method.

Because the loss function is explicit in our formulation, we can introduce an epsilon-insensitive zone into the robust loss function and achieve sparseness and robustness at the same time due to the conditions for optimality in (14). The epsilon-insensitive zone was introduced by Vapnik [24] to impose sparseness in support vector regression. We propose a robust loss function with bounded influence function and tunable parameters ϵ and

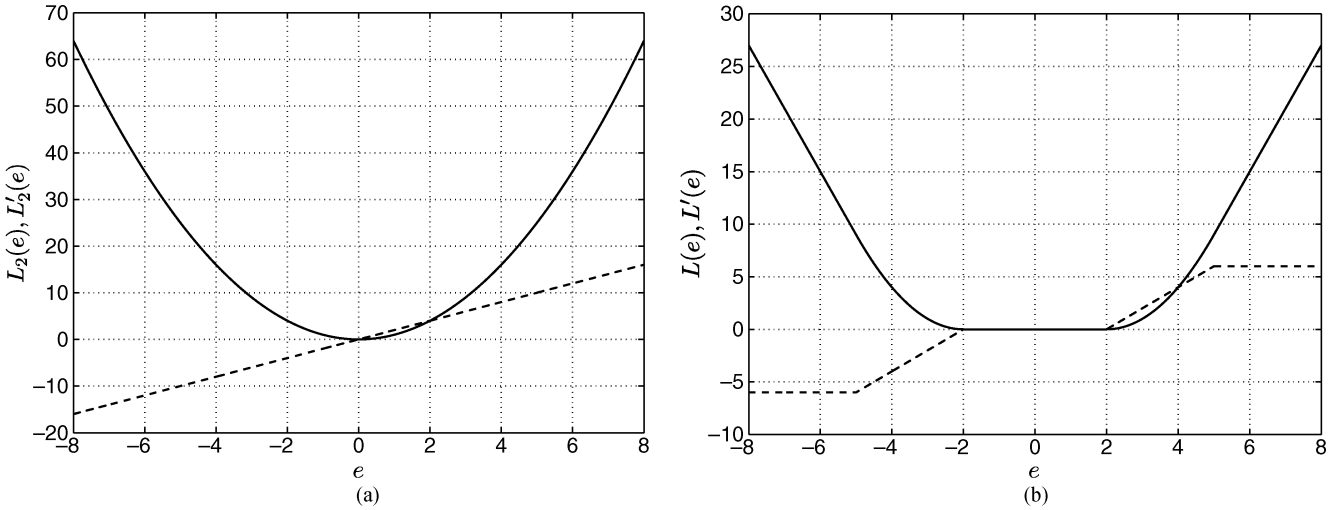

Fig. 1. (a) L_2 loss function and its influence function. (b) Proposed loss function with $\epsilon = 2$, $h = 5$, and its influence function. Note that the $L'(e)$ is bounded which is important for robustness.

h for sparseness and robustness, respectively. This loss function is a combination of the Huber loss and the epsilon-insensitive loss function for support vector regression introduced by Vapnik [24]. The proposed loss function (Fig. 1) is defined as

$$L(e) = \begin{cases} 0, & \text{if } |e| \leq \epsilon \\ e^2 - 2|e| + \epsilon^2, & \text{if } \epsilon < |e| < h \\ 2(h-\epsilon)|e| - h^2 + \epsilon^2, & \text{if } |e| \geq h \end{cases} \quad (26)$$

with $h > \epsilon$. Its score function becomes

$$\psi(e) = L'(e) = \begin{cases} 0, & \text{if } |e| \leq \epsilon \\ 2\text{sign}(e)\left(|e| - \epsilon\right), & \text{if } \epsilon < |e| < h \\ 2\text{sign}(e)(h-\epsilon), & \text{if } |e| \geq h. \end{cases} \quad (27)$$

Taking an epsilon zone in the loss function leads to sparseness because of the conditions for optimality (14). Note that setting ϵ to zero leads to the Huber loss function and $\epsilon > 0$, $h \to \epsilon$ approaches to the Vapnik epsilon-insensitive loss function.

V. The Preimage Problem

The projections onto the subspace spanned by the principal components lie in some feature space $\mathcal{F}$ but in most applications of kernel PCA we are interested in a reconstruction in the input space rather than in $\mathcal{F}$. This mapping from the feature space back to the input space is called the preimage problem. In this section, we summarize three preimage algorithms that will be used in the sequel.

- Preimage algorithm 1:
  An iterative nonlinear optimization method is proposed in [3] for Gaussian kernels, but it can suffer from numerical instabilities and local minima. Given a test point x, we want to recover $\hat{x}$ for which $\varphi(\hat{x}) \approx P_{N_c}\varphi(x)$, where $P_{N_c}\varphi(x)$ is the projection of $\varphi(x)$ onto the subspace spanned by the first N_c principal components. For Gaussian kernels, the approximate preimage $\hat{x}$ can be computed using the following iteration [3]:

$$\hat{x}^{(t)} = \frac{\sum_{i=1}^{N} \eta_i \exp\left(-\left\|\hat{x}^{(t-1)} - x_i\right\|^2 / \sigma^2\right) x_i}{\sum_{i=1}^{N} \eta_i \exp\left(-\left\|\hat{x}^{(t-1)} - x_i\right\|^2 / \sigma^2\right)} \quad (28)$$

  where $\eta_i = \sum_{j=1}^{N_c} \beta_j u_i^{(j)}$, β_j is the projection of $\varphi(x)$ onto the jth component $u^{(j)}$, and t is the iteration step.
- Preimage algorithm 2:
  Another approach is proposed in [6]. It is a modified version of (28) to remove iteratively the influence of outliers. Let $l_j^{(t)}$ be the certainty of $x_{(j)}$, where $x_{(j)}$ denotes the jth coordinate of the test point x. This certainty is calculated using the difference between $x_{(j)}$ and the corresponding reconstruction $\hat{x}_{(j)}^{(t-1)}$ as

$$l_j^{(t)} = \exp\left(-\left(x_{(j)} - \hat{x}_{(j)}^{(t-1)}\right)^2 / \left(2\sigma_j^2\right)\right), \quad j = 1, \ldots, d. \quad (29)$$

  The parameter σ_j is the standard deviation of the differences and is estimated using robust estimation [9]

$$\sigma_j = 1.4826\left(1 + 5/(N-1)\right) \underset{i}{\text{median}} \sqrt{\varepsilon_{ij}^2} \quad (30)$$

  where ε_{ij}^2 is the squared error between the jth dimension of the ith data point and its reconstruction. The approximate preimage $\hat{x}$ can be computed using the following iteration [6]:

$$\hat{x}^{(t)} = \frac{\sum_{i=1}^{N} \eta_i \exp\left(-\left\|g^{(t)} - x_i\right\|^2 / \sigma^2\right) x_i}{\sum_{i=1}^{N} \eta_i \exp\left(-\left\|g^{(t)} - x_i\right\|^2 / \sigma^2\right)} \quad (31)$$

where $g^{(t)}$ is given by

$$g^{(t)} = L^{(t)}x + \left(I_d - L^{(t)}\right)\hat{x}^{(t-1)} \tag{32}$$

for $t > 0$. The matrix I_d is the $d \times d$ identity matrix, d is the dimensionality of the input vectors, and $L^{(t)}$ is a $d \times d$ matrix defined as $L^{(t)} = \text{diag}(l_1^{(t)}, \ldots, l_d^{(t)})$.

- Preimage algorithm 3:
  A newer approach is proposed in [5]. This method is noniterative, involves only linear algebra, and uses the distances between the projection $P_{N_c}\varphi(x)$ and its nearest neighbors. The method can be summarized as follows: given the N_n neighbors $\{x_1, \ldots, x_{N_n}\}$ and $q = [q_1, \ldots, q_{N_n}]^T$ where q_l is the input space distance between $P_{N_c}\varphi(x)$ and x_l, $l = 1, \ldots, N_n$, first construct the matrix $\tilde{X} = [x_1, x_2, \ldots, x_{N_n}]$ and compute its singular value decomposition (SVD)

$$\tilde{X}M_c = GR \tag{33}$$

where M_c is the centering matrix, G is the matrix of left singular vectors, $R = \Lambda H^T = [r_1, \ldots, r_{N_n}]$, Λ is the diagonal matrix of singular values, H is the matrix of right singular vectors, and r_j is the projection of x_j onto G, $j = 1, \ldots, N_n$. The approximate preimage can be obtained as

$$\hat{r} = -\frac{1}{2}\Lambda^{-1}H^T(q - q_0) \tag{34}$$

where $q_0 = [\|r_1\|^2, \ldots, \|r_{N_n}\|^2]^T$. Transforming back to the original coordinate system in the input space leads to

$$\hat{x} = G\hat{r} + \bar{x} \tag{35}$$

where $\bar{x} = (1/N_n)\sum_{j=1}^{N_n} x_j$.

VI. Algorithms

We propose two algorithms to obtain the components with the Huber epsilon-insensitive loss function. The first algorithm solves (12) using an SQP approach [25] and smooth approximation of $L'(e)$. The second algorithm corresponds to an implementation of the weighted scheme (19).

A. SQP Algorithm

The Huber epsilon-insensitive influence function $L'(e)$ is not differentiable at $|e| = \epsilon$ and $|e| = h$, therefore to use a gradient-based approach such as SQP, a smooth approximation $L'_s(e)$ to $L'(e)$ is taken

$$L'_s(e) = \begin{cases} \frac{2}{s}\log\left(1 + \exp\left(s(e+h)\right)\right) - 2(h-\epsilon), \\ \qquad \text{if } e \le \frac{\epsilon - h}{2} - \epsilon \\ \frac{2}{s}\log\left(\frac{1+\exp(s(e-\epsilon))}{1+\exp(-s(e+\epsilon))}\right), \\ \qquad \text{if } \frac{\epsilon-h}{2} - \epsilon < |e| < \frac{h-\epsilon}{2} + \epsilon \\ -\frac{2}{s}\log\left(1 + \exp\left(-s(e-h)\right)\right) + 2(h-\epsilon), \\ \qquad \text{if } e \ge \frac{h-\epsilon}{2} + \epsilon \end{cases} \tag{36}$$

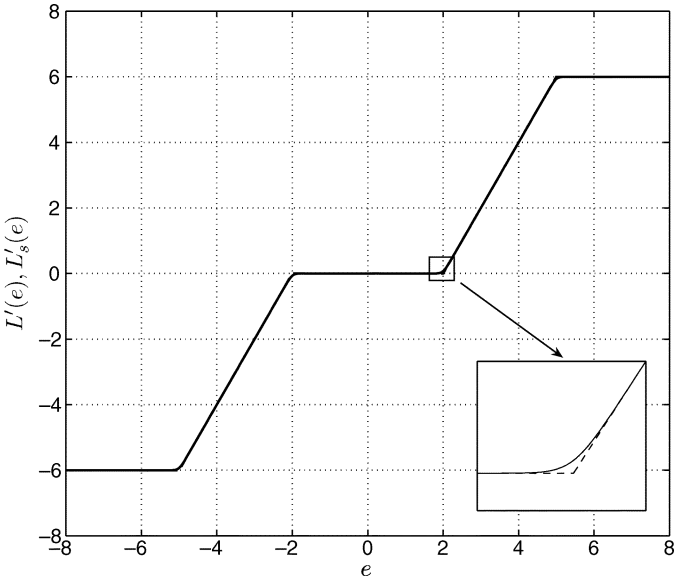

Fig. 2. Smooth approximation of the Huber epsilon-insensitive influence function. The dashed line corresponds to $L'(e)$ and the solid line to $L'_s(e)$.

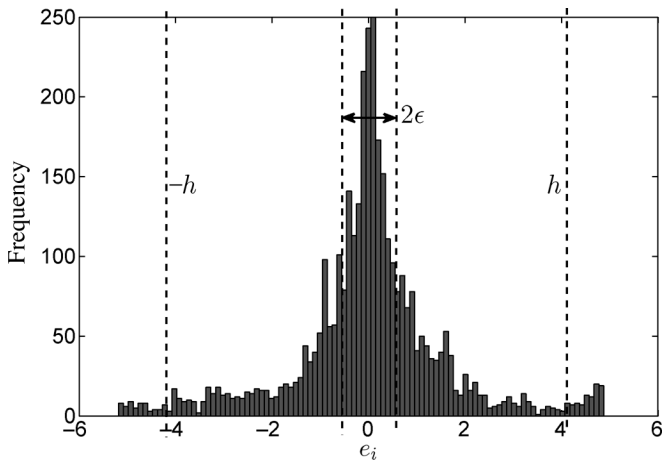

Fig. 3. Empirical distribution of the score variables associated to the kernel principal components. The degree of sparseness and robustness can be determined heuristically from this plot.

where s is a constant that controls the smoothness of the approximation. Fig. 2 shows $L'_s(e)$ and $L'(e)$ for $\epsilon = 2$, $h = 5$, and $s = 20$.

Replacing $L'(\cdot)$ by $L'_s(\cdot)$ in (12) leads to

$$f(\lambda, \alpha, b) = \begin{bmatrix} \lambda\alpha - \frac{1}{2}\tilde{L}'_s\left(z(\alpha, b)\right) \\ 1_N^T\tilde{L}'_s\left(z(\alpha, b)\right) \end{bmatrix} = 0 \tag{37}$$

where $f : \mathbb{R}^+ \times \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^{N+1}$, $\lambda = 1/\gamma$, $z(\alpha, b) = [z_1(\alpha, b); z_2(\alpha, b); \ldots; z_N(\alpha, b)]$, $\alpha = [\alpha_1; \alpha_2; \ldots; \alpha_N]$, $z_i(\alpha, b) = \Omega^{(i)}\alpha + b1_N$, $\Omega^{(i)}$ denotes the ith row of the kernel matrix Ω, and

$$\tilde{L}'_s\left(z(\alpha, b)\right) = \left[L'_s\left(z_1(\alpha, b)\right); \ldots; L'_s\left(z_N(\alpha, b)\right)\right].$$

The set of nonlinear equations (37) is usually solved via minimization of a sum of squares of the residuals. We impose two

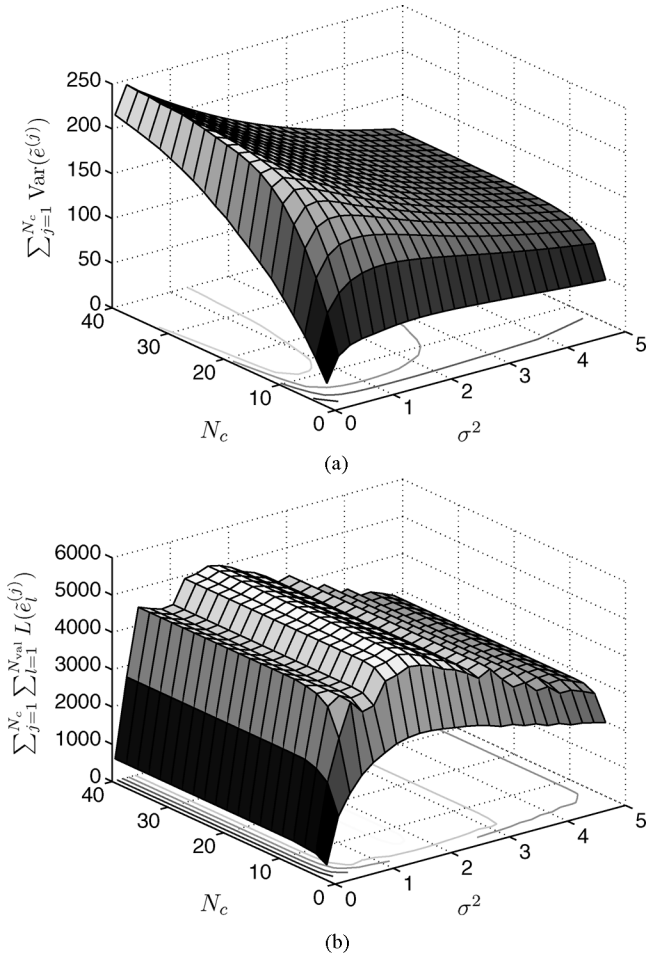

Fig. 4. Model selection surfaces on validation data. (a) Kernel PCA. (b) KCA with SQP algorithm. Note that the kernel PCA criterion is always increasing with respect to the number of components.

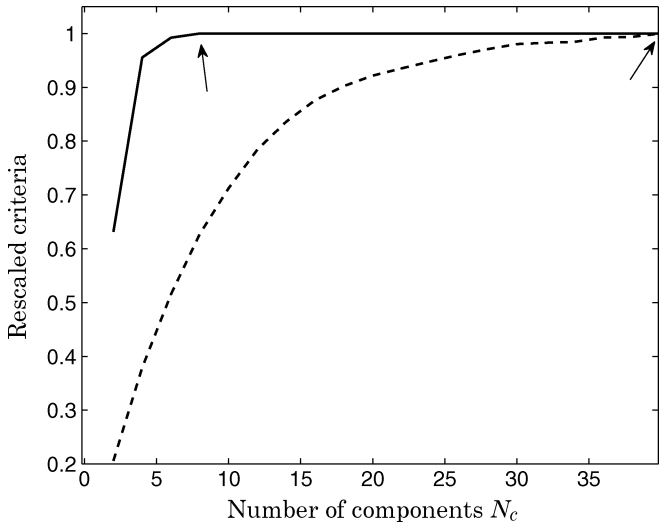

Fig. 5. Slice plot of the model selection surfaces for the best RBF kernel parameter σ^2. Dashed line: Average of the maximal variance on validation data (kernel PCA). Solid line: Average of the maximal $L(e_i^{\mathrm{val}})$ on validation data (KCA with SQP algorithm). The arrows show the selected number of components. The value of the proposed criterion saturates with less components compared to kernel PCA. Obtained parameters are $\sigma^2 = 0.45$ and $N_c = 40$ for kernel PCA and $\sigma^2 = 2$ and $N_c = 8$ for the SQP algorithm.

additional constraints $\alpha^T\alpha \geq 1$ and $\lambda \geq c$, where c is a small positive constant (e.g., $c = 10^{-6}$). The first constraint prevents the trivial solution $\alpha = 0$ and noninteresting solutions with small norm causing the score variables $z_i(\alpha, b)$ to lie in the range $-h < |z_i(\alpha, b)| < h$, $i = 1, \ldots, N$, therefore avoiding the bounded zone $|z_i(\alpha, b)| \geq h$ of the score function. The proposed nonlinear optimization problem is the following:

$$\min_{\lambda,\alpha,b} \quad g(\lambda,\alpha,b) = f(\lambda,\alpha,b)^T f(\lambda,\alpha,b) \quad \text{such that} \quad \begin{cases} \alpha^T\alpha \geq 1 \\ \lambda \geq c \end{cases} \tag{38}$$

where

$$g(\lambda,\alpha,b) = \lambda^2\alpha^T\alpha - \lambda\alpha^T\tilde{L}_s'(z(\alpha,b)) + \frac{1}{4}\left(\tilde{L}_s'(z(\alpha,b))\right)^T \tilde{L}_s'(z(\alpha,b)) + \left(1_N^T\tilde{L}_s'(z(\alpha,b))\right)^2.$$

All $N+1$ equations have to be satisfied at the local solution $(\lambda^\star, \alpha^\star, b^\star)$ rather than just minimizing the sum of the squares of the equations. This means that the value of $g(\lambda^\star, \alpha^\star, b^\star)$ should be zero in order for $(\lambda^\star, \alpha^\star, b^\star)$ to be a solution to the problem (37). In the same fashion as in the weighting scheme, additional constraints can be added to impose orthogonality with previously found components. Note that sparseness is achieved *componentwise* due to $\alpha_i = (\gamma/2)L'(e_i)$, which is one of the optimality conditions (14). Therefore, each component vector may have a different number of zero α_i values.

An SQP method approximates in every iteration the constrained optimization problem (38) as a quadratic programming problem of the form

$$\min_d \quad \frac{1}{2}d^T H^{(k)} d + \nabla g\left(\theta^{(k)}\right)^T d \quad \text{such that} \quad \begin{cases} \nabla c_1\left(\theta^{(k)}\right)^T d + c_1\left(\theta^{(k)}\right) \geq 0 \\ \nabla c_2\left(\theta^{(k)}\right)^T d + c_2\left(\theta^{(k)}\right) \geq 0 \end{cases} \tag{39}$$

where k denotes the iteration step, H is the Hessian matrix (or an approximation to it) of the Lagrangian of (38), $\theta = [\lambda; \alpha; b]$ is the vector of parameters, $c_1(\theta) = \alpha^T\alpha - 1$ is the first constraint, $c_2(\theta) = \lambda - c$ is the second constraint, $g(\theta) = g(\lambda; \alpha; b)$, and the gradient $\nabla g(\theta) \in \mathbb{R}^{N+2}$ is defined as

$$\nabla g(\theta) = \nabla g(\lambda,\alpha,b) = \begin{bmatrix} \nabla_\lambda g(\lambda,\alpha,b) \\ \nabla_\alpha g(\lambda,\alpha,b) \\ \nabla_b g(\lambda,\alpha,b) \end{bmatrix} \tag{40}$$

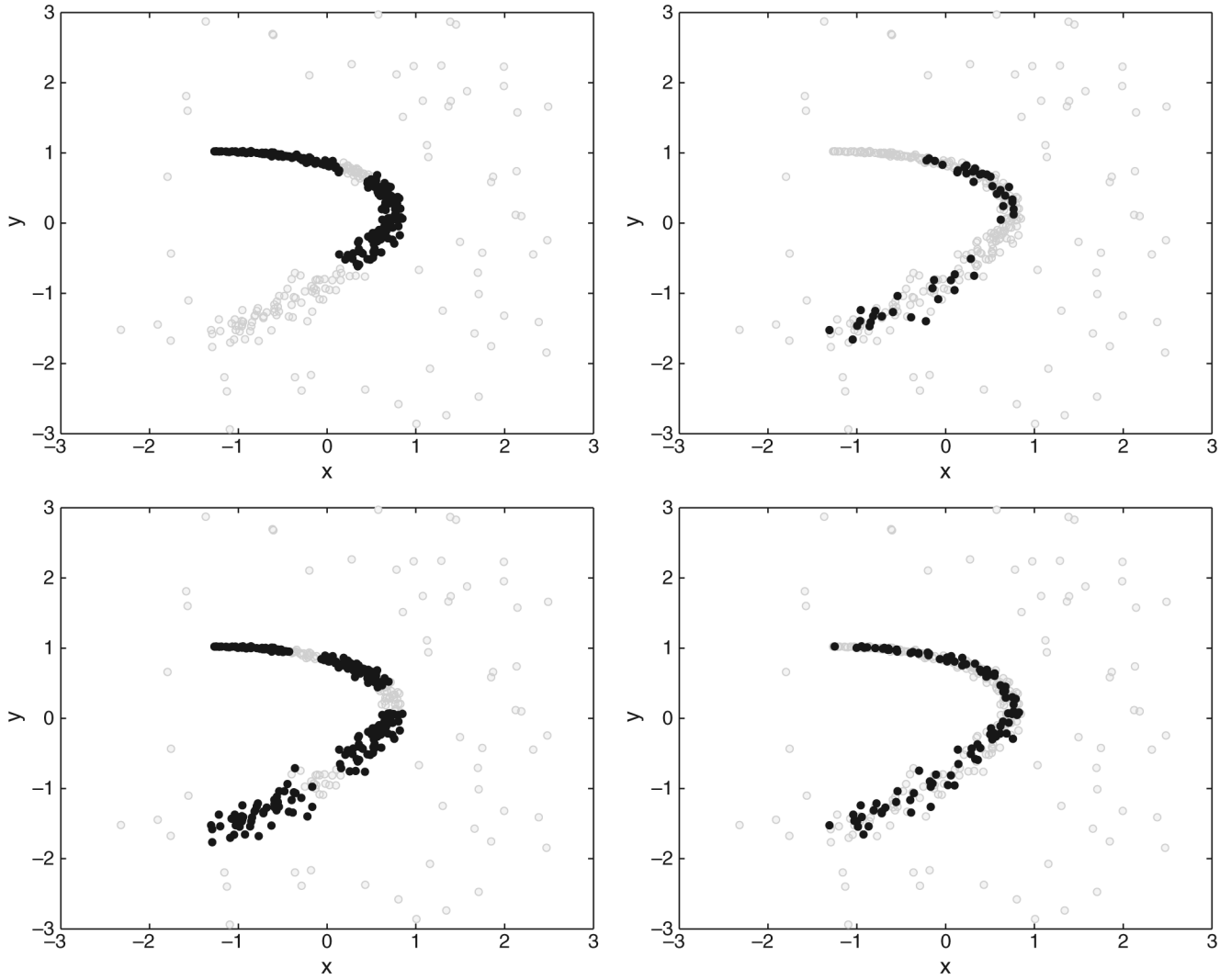

Fig. 6. Sparseness per component. The figure shows the support vectors (black points) for the first four components. Most of the outliers are not support vectors in any component, therefore they can be removed from the data set.

where

$$\begin{aligned}
\nabla_\lambda g(\lambda,\alpha,b) &= 2\lambda\alpha^T\alpha - \alpha^T\tilde{L}_s'\left(z(\alpha,b)\right)\\
\nabla_\alpha g(\lambda,\alpha,b) &= 2\lambda^2\alpha\\
&\quad - \lambda\left(\Omega\mathrm{diag}\left(\tilde{L}_s''\left(z(\alpha,b)\right)\right)\alpha + \tilde{L}_s'\left(z(\alpha,b)\right)\right)\\
&\quad + \frac{1}{2}\Omega\mathrm{diag}\left(\tilde{L}_s''\left(z(\alpha,b)\right)\right)\tilde{L}_s'\left(z(\alpha,b)\right)\\
&\quad + 2r_1\Omega\tilde{L}_s''\left(z(\alpha,b)\right)\\
\nabla_b g(\lambda,\alpha,b) &= -\lambda\alpha^T\tilde{L}_s''\left(z(\alpha,b)\right)\\
&\quad + \frac{1}{2}\left(\tilde{L}_s''\left(z(\alpha,b)\right)\right)^T\tilde{L}_s'\left(z(\alpha,b)\right) + 2r_1r_2
\end{aligned}$$

$r_1 = 1_N^T\tilde{L}_s'(z(\alpha,b))$, and $r_2 = 1_N^T\tilde{L}_s''(z(\alpha,b))$. The solution vector $d^{(k)}$ to the QP subproblem (39) can then be used to form a new iterate

$$\theta^{(k+1)} = \theta^{(k)} + \mu^{(k)}d^{(k)} \tag{41}$$

where $\mu^{(k)}$ is the step length parameter of the line search and can be obtained using a merit function.

A critical issue in nonlinear optimization is the selection of good starting points. In the proposed SQP approach, kernel PCA eigenvectors and eigenvalues are used as initial values for the nonlinear solver. Therefore, we are considering only the first local solution obtained through the proposed initialization. Given a set of N d-dimensional data points $\mathcal{X} = \{x_1,\ldots,x_N\}$, the algorithm gives a set of N_c component vectors $\mathcal{A} = \{\alpha^{(1)\star},\ldots,\alpha^{(N_c)\star}\}$ as follows.

Algorithm 1 KCA: SQP Approach

1: Set the kernel parameters and the loss function parameters ϵ,h.

2: Perform an eigendecomposition of the noncentered kernel matrix to obtain initial eigenvalues $\{\lambda_1,\ldots,\lambda_N\}$ and initial eigenvectors $\{\alpha^{(1)},\ldots,\alpha^{(N)}\}$.

3: **for** $l = 1$ to N_c **do**

4: Obtain the initial values for the bias term $b_l = (1/N)1_N^T\Omega\alpha^{(l)}$

5: Solve the SQP problem (39), (41) with initial values λ_l, $\alpha^{(l)}$, and b_l to obtain the solutions $\lambda_l^\star$, $\alpha^{(l)\star}$, and $b_l^\star$

6: **end for**

7: Sort the solutions according to the value of $\lambda_l^\star$ in decreasing order.

8: Normalize the components $\alpha^{(l)\star} \leftarrow \alpha^{(l)\star}/\|\alpha^{(l)\star}\|_2$, $l = 1, \ldots, N_c$.

B. Weighted Algorithm

The second algorithm is an implementation of the iterative weighting scheme (18). Given $\mathcal{X}$, and the number of iterations N_{iter}, the algorithm (2) gives a set of N_c components $\{\alpha^{(1)}, \ldots, \alpha^{(N_c)}\}$, where $N_r \leq N$.

Algorithm 2 KCA: Weighted Kernel PCA

1: Set the kernel parameters, the loss function parameters ϵ and h, and the number of principal components N_c to be extracted.

2: Compute the unweighted score variables $e_i^{(l)}$, $i = 1, \ldots, N$, $l = 1, \ldots, N_c$, associated to the first N_c principal components using classical kernel PCA.

3: Compute the weights $v_i^{(l)}$ such that $v_i^{(l)}(e_i^{(l)})^2 = L(e_i^{(l)})$.

4: Remove from the training data set the data points whose corresponding weight $v_i^{(l)}$ equals to zero in every component. The new data set has N_r data points.

5: Compute the unweighted score variables $e_i^{(l)}$, $l = 1, \ldots, N_c$, of the reduced data set using classical kernel PCA.

6: **for** $k = 1$ to N_{iter} **do**

7: Recompute the weights $v_i^{(1)}$ such that $v_i^{(1)}(e_i^{(1)})^2 = L(e_i^{(1)})$ with $\epsilon = 0$.

8: Find the first weighted principal component $\alpha^{(1)}$ which is the solution associated to the largest eigenvalue in $V\Omega_c\alpha = \lambda\alpha$, with $V = \text{diag}(v_1, \ldots, v_{N_r})$.

9: Recompute $e_i^{(1)}$ using $\alpha^{(1)}$.

10: **for** $\{l = 2$ to N_c **do**

11: Recompute the weights $v_i^{(l)}$ such that $v_i^{(l)}(e_i^{(j)})^2 = L(e_i^{(l)})$ with $\epsilon = 0$

12: Find the lth weighted principal component $\alpha^{(l)}$ which is the solution associated to the largest eigenvalue in (19).

13: Recompute $e_i^{(l)}$ using $\alpha^{(l)}$.

14: **end for**

15: **end for**

TABLE I
ACCUMULATED NUMBER OF SUPPORT VECTORS FOR THE SQP ALGORITHM AND THE REDUCED SET SELECTION METHOD [4] WITH RESPECT TO THE NUMBER OF COMPONENTS REQUIRED

# of Components	SQP Algorithm	Reduced Set [4]
1	190	160
2	222	221
3	286	257
4	290	272

TABLE II
FRACTION OF THE TOTAL NUMBER OF OUTLIERS PRESENT ON THE SUPPORT VECTOR SET. NOTE THAT THE REDUCED SET METHOD IS MORE SPARSE THAN THE PROPOSED METHODS, BUT THE MAJORITY OF THE OUTLIERS ARE IN THE SUPPORT VECTOR SET

	% of outliers	Computation times
Reduced Set [4]	98%	5.4 secs
Weighted Algorithm	3%	8.6 secs
SQP Algorithm	0%	24.7 secs

In general, empirical results suggest that only one iteration ($N_{\text{iter}} = 1$) is sufficient to obtain satisfactory results. Sparseness is achieved in step 4 by removing data points from the training set for which the corresponding weight is equal to zero across the components. The computational complexity of this algorithm is $\mathcal{O}(N_{\text{iter}}N_c(\xi N + N_c))$ compared with $\mathcal{O}(\xi N)$ for classical kernel PCA using the Lanczos method for the largest N_c components. The factor ξ is the maximum number of matrix–vector computations required and depends on the eigengap $\lambda_{N_c+1} - \lambda_{N_c}$ [26], [27].

VII. MODEL SELECTION

Model selection is a central issue in unsupervised learning techniques such as kernel PCA. In supervised learning, one can rely on many methods (such as validation set, cross validation, generalization bounds, etc.) for model selection. However, one often cannot apply such criteria in unsupervised learning due to the lack of an underlying objective function to the method. In kernel PCA, the parameters that define the model are the number of principal components N_c to be extracted and the kernel parameters [e.g., σ^2 for the radial basis function (RBF) kernel]. These parameters are typically chosen in a trial-and-error fashion. In the proposed approaches, model selection in a learning framework can be done naturally on the basis of the explicit constrained optimization formulation (11). We propose the following model selection criterion:

$$\max_{N_c,\sigma^2} \sum_{l=1}^{N_c} \sum_{j=1}^{N_{\text{val}}} L\left(\tilde{e}_j^{(l)}\right) \tag{42}$$

where $\tilde{e}_j^{(l)} = w^T\varphi(x_j^{\text{val}}) + b_l = \sum_{i=1}^{N} \alpha_i^{(l)} K(x_i, x_j^{\text{val}}) + b_l, x_l^{\text{val}}$ denotes the jth validation data point, $\alpha^{(l)}$ is the lth component, and b_l is the bias term for the lth component, $l = 1, \ldots, N_c$. The values of ϵ (sparseness) and h (robustness) are problem dependent and can be obtained heuristically from the empirical distribution of the score variables such that a given amount of the score values lie within the epsilon zone of the loss function.

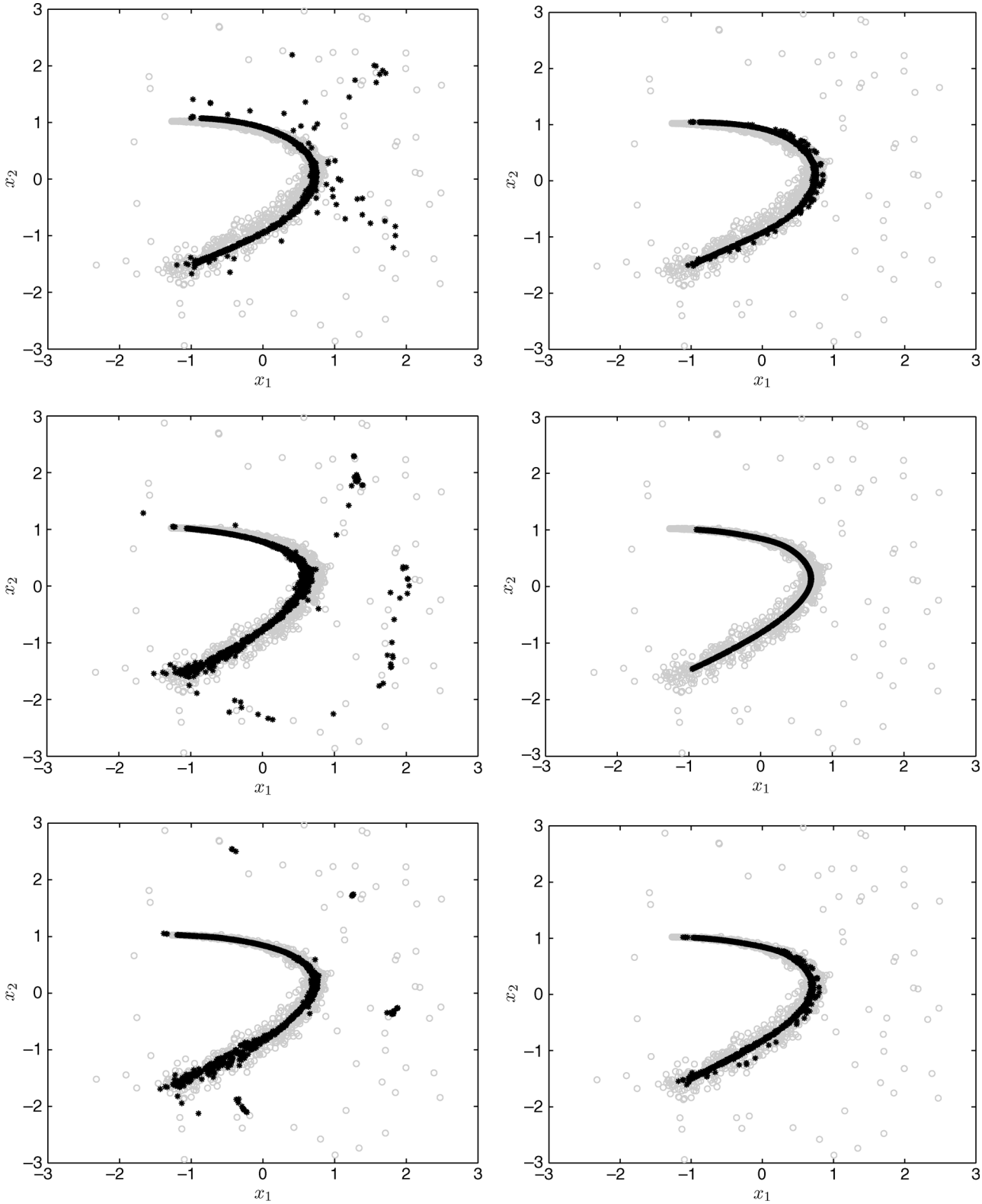

Fig. 7. Denoising results using three preimage algorithms. Original data is depicted in gray and denoised data point in black. Left column: Kernel PCA with $\sigma^2 = 0.45$ and $N_c = 40$. Right column: KCA with SQP algorithm with $\sigma^2 = 2.0$, $N_c = 8$, $\epsilon = 0.34$, and $h = 3.7$. Top: Preimage algorithm 1. Center: Preimage algorithm 2. Bottom: Preimage algorithm 3. In all cases, the SQP approach performs better in terms of robustness to outliers.

In the same way, h can be set such that a given percentage of the score values lie within the range $-h < e < h$.Fig. 3 shows the proposed heuristic.

VIII. Empirical Results

In this section, some experimental results are presented to illustrate the effectiveness of the proposed approaches. We used

TABLE III
MSE AND STANDARD DEVIATION ON TEST DATA OF THE DENOISED DIGITS COMPARED TO THE CLEAN TEST DATA

	Kernel PCA	KCA: SQP algorithm	KCA: Weighted Algorithm
Pre-image algorithm 1	0.60 ± 0.40	0.10 ± 0.15	0.28 ± 0.30
Pre-image algorithm 2	0.70 ± 0.30	0.21 ± 0.20	0.45 ± 0.32
Pre-image algorithm 3	0.54 ± 0.40	0.30 ± 0.23	0.32 ± 0.28

the MATLAB function *fmincon* to solve (39). This function is an SQP solver with a quasi-Newton approximation to the Hessian of the Lagrangian using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method. The step length is calculated with line search and the merit function described in [28]. All experiments reported were carried out using the RBF kernel. It is important to mention that the kernel should be bounded for these robust methods to work. The machine used is a Pentium 4, 2.8-GHz, 1-GB RAM. The values of ϵ and h were obtained using the heuristic proposed in Fig. 3, setting ϵ such that 10% of the score values lie within the epsilon-insensitive range and setting h such that 90% of the score values lie within the range $-h < e < h$.

Experiment 1—Denoising

The first experiment consists of a denoising application. Simulations were done with a curve-shaped data set in two dimensions and with the University of California at Irvine (UCI) *multiple features* handwritten digit data set. The curve data set consists of 360 data points for training, 420 for validation, and 320 for testing. The loss function parameters $\epsilon = 0.34$ and $h = 3.7$ were set according to Fig. 3. Model selection surfaces for the curve data set can be seen in Fig. 4 showing that in the weighted kernel PCA method, the criterion saturates with respect to the number of components N_c.

On the other hand, in kernel PCA, the variance of the projected variables is always increasing with respect to N_c. This means that fewer components have to be computed in the weighted kernel PCA algorithm. Fig. 5 shows a slice of the surface plots. The values of the proposed criterion and the kernel PCA criterion (variance) were rescaled for graphical purposes. Obtained parameters are $\sigma^2 = 0.45$ and $N_c = 40$ for kernel PCA and $\sigma^2 = 2.0$ and $N_c = 8$ for the SQP algorithm. Sparseness of the proposed method can be seen in Fig. 6 and Table I.

Note that sparseness varies from component to component. However, most of the outliers are not support vectors, therefore they can be removed from the data set. Table II shows the percentage of outliers present in the support vector set. The reduced set selection method [4] is more sparse than the proposed methods but its support vector set contains almost all outliers. The components found by the SQP algorithm are more sparse than the components obtained by the simplified weighting scheme but at a more expensive computational cost. Fig. 7 shows the denoising results using the SQP approach and classical kernel PCA with three different preimage algorithms.

The handwritten digit training set is composed of 195 digit images of 15×16 pixels. The validation set consists of 40 digits. We added outliers and Gaussian noise to the data set and set

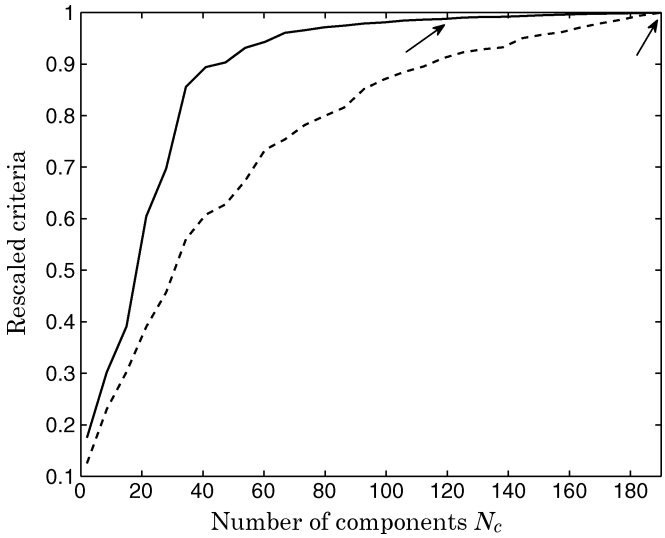

Fig. 8. Slice plot of the model selection surfaces for the best RBF kernel parameter σ^2. Dashed line: Average of the maximal variance on validation data (kernel PCA). Solid line: Average of the maximal $L(e_i^{\text{val}})$ on validation data (KCA with SQP algorithm). The arrows show the selected number of components. The value of the proposed criterion saturates with less components compared to kernel PCA. Obtained parameters are $\sigma^2 = 0.45$ and $N_c = 40$ for kernel PCA and $\sigma^2 = 2$ and $N_c = 8$ for the SQP algorithm.

$\epsilon = 0.1$ and $h = 4$ according to Fig. 3. Fig. 8 shows the slices of the model selection surfaces with similar results compared to Fig. 4. Obtained parameters are $\sigma = 350$ and $N_c = 190$ for kernel PCA and $\sigma^2 = 300$ and $N_c = 120$ for the SQP algorithm. The denoising results can be seen in Figs. 9 and 10 showing improved results of the SQP algorithm compared to kernel PCA in terms of robustness together with visually appealing digits. Table III shows the mean squared error (MSE) and standard deviation for kernel PCA and the proposed methods with respect to the three preimage algorithms. The MSE was calculated between the denoised digits and the test set without outliers and Gaussian noise. Fig. 11 shows sparseness results in terms of accumulated number of support vectors with respect to the number of components.

Experiment 2—Feature Extraction

The feature extraction experiment was carried out with a square-shaped data set in two dimensions consisting of 1000 data points. The weighted kernel PCA algorithm was used. The training scenario consisted of 300 data points for training, 300 data points for validation, and the remaining 400 data points for testing. We included outliers and performed ten randomizations of the training and validation sets. The parameters of the loss function were set to $\epsilon = 0.5$ and $h = 3$. Model selection surfaces and slices were obtained in a similar way as in the previous experiments. Selected parameters are $\sigma^2 = 0.45$

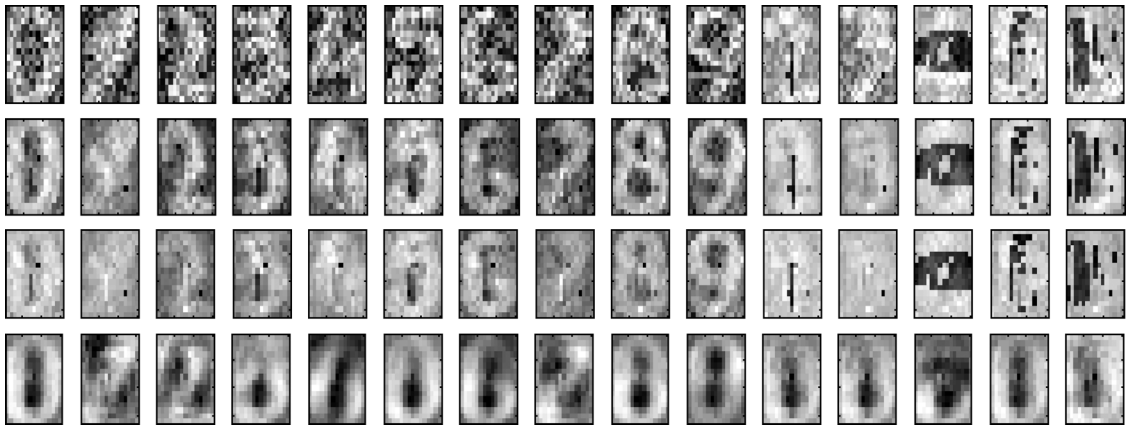

Fig. 9. Denoising results using kernel PCA with $\sigma^2 = 350$ and $N_c = 190$. First row: Test set corrupted with Gaussian noise and outliers. Second row: Denoising with preimage algorithm 1. Third row: Denoising with preimage algorithm 2. Fourth row: Denoising with preimage algorithm 3.

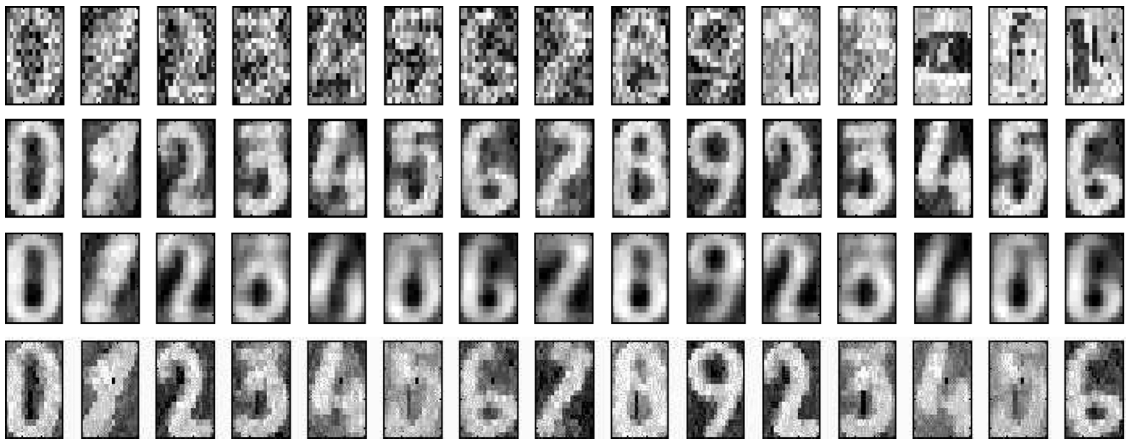

Fig. 10. Denoising results using the SQP algorithm with $\sigma^2 = 300$ and $N_c = 120$. First row: Test set corrupted with Gaussian noise and outliers. Second row: Denoising with preimage algorithm 1. Third row: Denoising with preimage algorithm 2. Fourth row: Denoising with preimage algorithm 3. The SQP algorithm show improved results in terms of robustness compared to Fig. 9.

and $N_c = 6$. Figs. 12–14 show the feature extraction results for kernel PCA and weighted kernel PCA, respectively. The contours of constant score value are shown in different levels of gray. We can see that in kernel PCA some feature values are completely determined by the outliers which is not the case in weighted kernel PCA.

IX. Conclusion

A new generalization to the kernel PCA formulation for general loss functions was derived. This formulation is based on the LS-SVM context and can be used to impose desirable properties such as sparseness and robustness in the analysis. These two properties are missing in the classical kernel PCA because it has an associated L_2 loss function, and the solutions are expressed in terms of dense kernel expansions. To overcome these issues, we proposed a loss function with an epsilon-insensitive zone for sparseness and bounded score function for robustness. We have shown two different algorithms to obtain components using this loss function. The first algorithm solves a set of nonlinear equations using an SQP method. This nonlinear solver was initialized using classical kernel PCA eigenvectors/values, which were shown to be good initial starting points leading to convergence in most cases. However, we only consider the first local solution obtained through this initialization while other

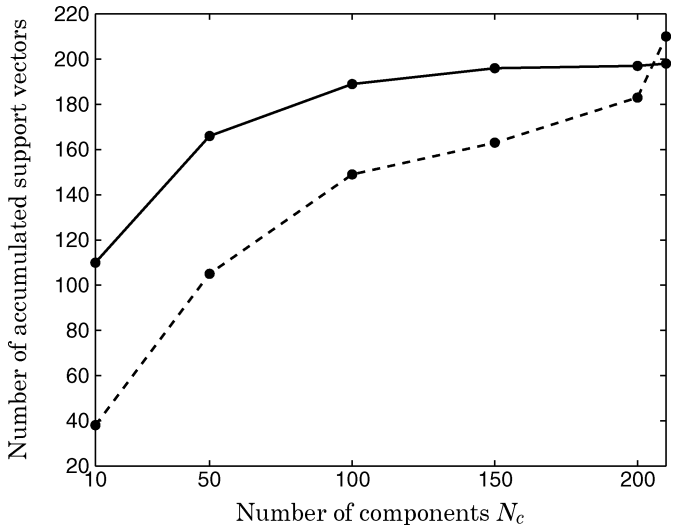

Fig. 11. Number of accumulated support vectors with respect to the number of components. Solid line: SQP algorithm. Dashed line: Reduced set selection method [4]. The reduced set method contains less support vectors but outliers are not removed.

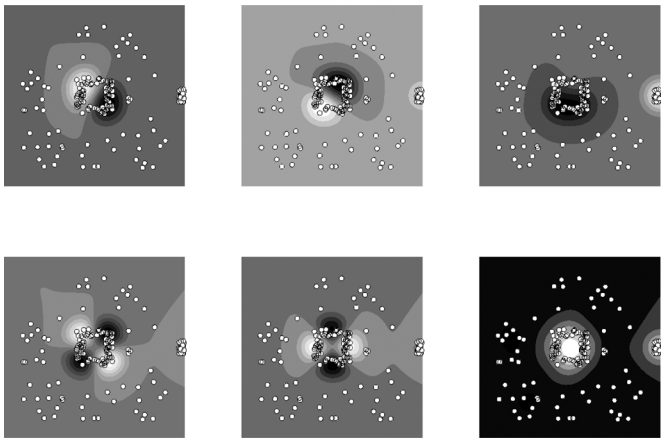

Fig. 12. Contours of constant projection value for the first six principal components with the RBF kernel and tuned parameters $\sigma^2 = 0.3$ and $N_c = 40$. The data points are shown as circles. Note that the outliers have influence in most of the components.

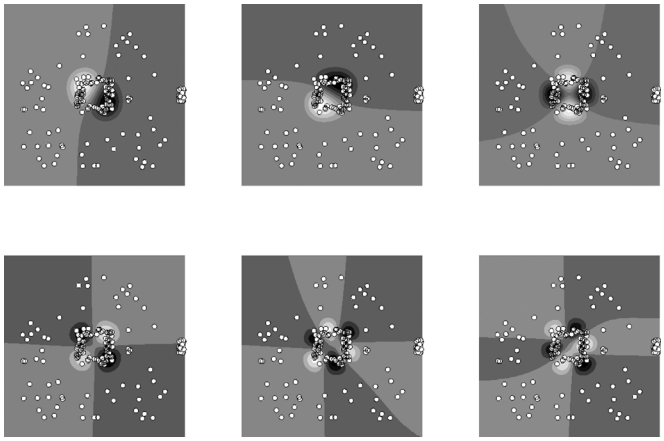

Fig. 13. Contours of constant projection value for the first six weighted principal components with the RBF kernel and tuned parameters $\sigma^2 = 0.45$, $N_c = 6$, $\epsilon = 0.5$, and $h = 3$. The data points are shown as circles. Note that in this case, the projection value is not affected by the outliers.

local optima may be interesting as well. The second algorithm is a simplified iterative weighting scheme that leads to a sequence of generalized eigenvalue problems, which can be solved efficiently. We proposed a model selection criterion based on the primal/dual constrained optimization framework to obtain the kernel parameters and the number of components. A heuristic to find good choices of the loss function parameters was also discussed. While mutual orthogonality of the scores variables in classical kernel PCA implies mutual orthogonality of the eigenvectors, that is not the case with a general loss function. However, orthogonality constraints can be added to the primal optimization problem. From simulations results, the proposed methods showed good performance in terms of robustness together with a sparse representation. The components obtained by the SQP algorithm were shown to be sparser than the components found by the weighted approach. However, the latter algorithm was shown to be computationally faster.

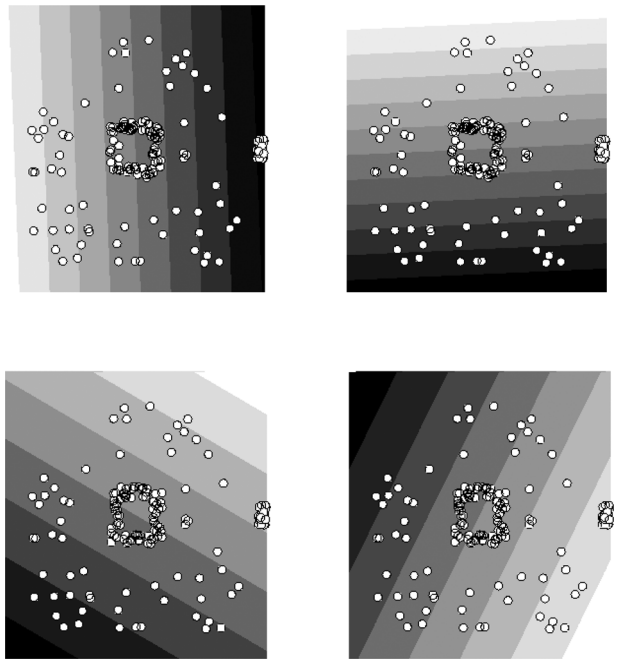

Fig. 14. Contours of constant projection value for the two components with the normalized linear kernel. Top: Kernel PCA. Bottom: Weighted kernel PCA with $\epsilon = 0.5$ and $h = 3$. Note that the nonlinear structure of the data is lost. In kernel PCA, the outlier cluster defines entirely the direction of the components which is not the case in weighted kernel PCA.

REFERENCES

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

[2] B. Schölkopf, A. J. Smola, and K.-R. Müller, "Nonlinear component analysis as a Kernel eigenvalue problem," *Neural Comput.*, vol. 10, pp. 1299–1319, 1998.

[3] S. Mika, B. Schölkopf, A. J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, "Kernel PCA and de-noising in feature spaces," in *Advances in Neural Information Processing Systems 11*, M. S. Kearns, S. A. Solla, and D. A. Cohn, Eds. Cambridge, MA: MIT Press, 1999.

[4] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, M. Scholz, G. Rätsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1000–1017, Sep. 1999.

[5] J. T. Kwok and I. W. Tsang, "The pre-image problem in kernel methods," *IEEE Trans. Neural Netw.*, vol. 15, no. 6, pp. 1517–1525, Nov. 2004.

[6] T. Takahashi and T. Kurita, "Robust de-noising by kernel PCA," in *Proc. Int. Conf. Artif. Neural Netw.*, 2002, pp. 739–744.

[7] J. A. K. Suykens, T. Van Gestel, J. Vandewalle, and B. De Moor, "A support vector machine formulation to PCA analysis and its kernel version," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 447–450, Mar. 2003.

[8] P. Huber, *Robust Statistics*. New York: Wiley, 1981.

[9] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*. New York: Wiley, 1987.

[10] F. Critchley, "Influence in principal component analysis," *Biometrika*, vol. 72, pp. 627–636, 1985.

[11] L. Xu and A. L. Yuille, "Robust principal component analysis by self-organizing rules based on statistical physics approach," *IEEE Trans. Neural Netw.*, vol. 6, no. 1, pp. 131–143, Jan. 1995.

[12] I. Higuchi and S. Eguchi, "The influence function of principal component analysis by self-organizing rule," *Neural Comput.*, vol. 10, pp. 1435–1444, 1998.

[13] A. J. Smola, O. Mangasarian, and B. Schölkopf, "Sparse kernel feature analysis," Data Mining Inst., Univ. Wisconsin, , Madison, WI, Tech. Rep. 99-03, 1999.

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

[15] C. Alzate and J. A. K. Suykens, "A weighted Kernel PCA formulation with out-of-sample extensions for spectral clustering methods," in *Proc. Int. Joint Conf. Neural Netw.*, 2006, pp. 138–144.
[16] J. A. K. Suykens, J. De Brabanter, L. Lukas, and J. Vandewalle, "Weighted least squares support vector machines: Robustness and sparse approximation," *Neurocomputing*, vol. 48, pp. 85–105, 2002.
[17] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1990.
[18] F. De la Torre and M. Black, "Robust principal component analysis for computer vision," in *Proc. Int. Conf. Comput. Vis.*, 2001, pp. 362–369.
[19] I. Higuchi and S. Eguchi, "Robust principal component analysis with adaptive selection for tuning parameters," *J. Mach. Learn. Res.*, vol. 5, pp. 453–471, 2004.
[20] C. Croux and G. Haesbroeck, "Principal component analysis based on robust estimators of the covariance or correlation matrix: Influence functions and efficiencies," *Biometrika*, vol. 87, no. 3, pp. 603–618, 2000.
[21] M. Hubert, P. J. Rousseeuw, and K. Vanden Branden, "ROBPCA: A new approach to robust principal component analysis," *Technometrics*, vol. 47, pp. 64–79, 2005.
[22] F. R. Hampel, E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel, *Robust Statistics: The Approach Based on Influence Functions*. New York: Wiley, 1986.
[23] A. Christmann and I. Steinwart, "On robustness properties of convex risk minimization methods for pattern recognition," *J. Mach. Learn. Res.*, vol. 5, pp. 1007–1034, Aug. 2004.
[24] V. Vapnik, *Statistical Learning Theory*. New York: Wiley, 1998.
[25] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer-Verlag, 2000.
[26] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: The Johns Hopkins University Press, 1996.
[27] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*. Oxford, U.K.: Clarendon, 1978.
[28] M. J. D. Powell, "A fast algorithm for nonlinearly constrained optimization calculations," in *Proc. Dundee Biennial Conf. Numer. Anal.*, G. A. Watson, Ed., 1978, pp. 144–157.



Carlos A. Alzate was born in Tuluá, Colombia, in 1980. He received the degree in electronic engineering from the Universidad Nacional de Colombia, Manizales, Colombia, in 2002 and the M.S. degree in artificial intelligence from the Katholieke Universiteit Leuven, Leuven, Belgium, in 2004, where he is currently working towards the Ph.D. degree at the Department of Electrical Engineering (ESAT).

His research interests include kernel methods, unsupervised learning, model selection, and optimization.



Johan A. K. Suykens (SM'05) was born in Willebroek, Belgium, on May 18, 1966. He received the degree in electromechanical engineering and the Ph.D. degree in applied sciences from the Katholieke Universiteit Leuven (K.U.Leuven), Leuven, Belgium, in 1989 and 1995, respectively.

In 1996, he was a Visiting Postdoctoral Researcher at the University of California, Berkeley. He was a Postdoctoral Researcher with the Fund for Scientific Research FWO Flanders and is currently a Professor (Hoogleraar) at the K.U.Leuven. His research interests are mainly in the areas of the theory and application of neural networks and nonlinear systems. He is author of the books *Artificial Neural Networks for Modelling and Control of Non-linear Systems* (Kluwer: Norwell, MA, 1996) and *Least Squares Support Vector Machines* (World Scientific: Singapore, 2002), coauthor of the book *Cellular Neural Networks, Multi-Scroll Chaos and Synchronization* (World Scientific, Singapore, 2005) and editor of the books *Nonlinear Modeling: Advanced Black-Box Techniques* (Kluwer: Norwell, MA, 1998) and *Advances in Learning Theory: Methods, Models and Applications* (IOS Press: Amsterdam, The Netherlands, 2003). In 1998, he organized an International Workshop on Nonlinear Modelling with Time-Series Prediction Competition.

Dr. Suykens has served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS (1997–1999 and 2004–2007) and since 1998 he has been serving as an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS. He received an IEEE Signal Processing Society 1999 Best Paper (Senior) Award and several Best Paper Awards at international conferences. He is a recipient of the International Neural Networks Society (INNS) 2000 Young Investigator Award for significant contributions in the field of neural networks. He has served as a Director and Organizer of the NATO Advanced Study Institute on Learning Theory and Practice (Leuven 2002), as a program cochair for the 2004 International Joint Conference on Neural Networks and the 2005 International Symposium on Nonlinear Theory and its Applications, and as an organizer of the 2007 International Symposium on Synchronization in Complex Networks.