# AN ITERATIVE ALGORITHM FOR ROBUST KERNEL PRINCIPAL COMPONENT ANALYSIS

**LEI WANG[1], YAN-WEI PANG[2], DAO-YI SHEN[1], NENG-HAI YU[1]**

[1]MOE-MS Key Laboratory of Multimedia Computing and Communication, Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei, 230027, China
[2]School of Electronic Information Engineering, Tianjin University, Tianjin, 300072, China
E-MAIL: wlei@ustc.edu, pyw@ustc.edu, dyshen@ustc.edu, ynh@ustc.edu.cn

**Abstract:**

***Principal Component Analysis* (PCA) has been proven to be an efficient method in dimensionality reduction, feature extraction and pattern recognition. *Kernel Principal Component Analysis* (KPCA) can be considered as a natural nonlinear generalization of PCA, which performs linear PCA in a high dimensional space implicitly by using kernel trick. However, both conventional PCA and KPCA suffer from the deficiency of being sensitive to outliers. Existing *robust KPCA* has to eigen-decompose the Gram matrix directly in each step and is much more computationally infeasible due to the large size of the matrix when the number of training samples is large. By extending existing robust PCA algorithm using kernel methods, we present a novel robust adaptive algorithm for calculating the kernel principal components. The proposed method not only preserves the characteristic of capturing underlying nonlinear structure of KPCA but also is robust against outliers by restraining the effect of outlying samples. Compared with existing robust KPCA methods, our method is performed without having to store the kernel matrix, which can reduce significantly the storage burden. In addition, our method shows the potential of expansibility to the incremental learning version. Experimental results on synthetic data indicate that our improved algorithm is effective and promising.**

**Keywords:**

**Robust principal component analysis; Robust kernel principal component analysis; Outliers; Dimensionality reduction; Feature extraction**

## 1. Introduction

*Principal Component Analysis* (PCA) is an efficient method for dimensionality reduction, feature extraction, and has been widely used in many fields, such as image processing, statistical analysis and pattern recognition [1]. Conventional PCA is to find a linear orthogonal basis transformation by an eigen-decomposition of the centered covariance matrix of the data set. Dimensionality reduction and feature extraction are achieved by projecting input data into the subspace spanned by a set of principal eigenvectors corresponding to the largest eigenvalues.

Linear PCA is suitable to describe data with Gaussian distribution, for it takes only second-order correlations into account. *Kernel PCA* (KPCA) can be considered as a natural nonlinear generalization of PCA, which can extract nonlinear structure from data set [2]. The basic idea of KPCA is to fist map the input data into some feature space via nonlinear map and then to execute linear PCA on the mapped data. It is generally computationally infeasible to execute PCA directly in the feature space due to the high dimensionality of the feature space. KPCA enables this by using kernel methods and formulating PCA as the equivalent kernel eigenvalue problem. On account of the attractive capability, KPCA based methods have been extensively investigated [3],[4],[5],[6], and have showed excellent performance.

However, both the classical PCA and KPCA algorithms, implemented in the sense of least mean squared error minimization, have the deficiency of instability when input samples are spoiled by outliers, the usual situations in many practical problems. As [7],[8] have reported, even small amount of outliers will significantly deteriorate the performance of standard PCA and KPCA. Nevertheless, it is practically not easy to separate the outliers from the true data. A number of efforts have been made to tackle this problem, and existing representative algorithms to compute the principal components robust against outliers include [7],[9],[10],[11]. However, to the best of our knowledge, the existing robust algorithms for kernel principal components are very limited except [8].

In [8], Lu et al. proposed a *robust KPCA*, which had to iteratively eigen-decompose the kernel matrix. The method recognizes the outliers by setting a global threshold of reconstruction error of the training samples and then eliminating the samples exceeding the threshold. Instead of giving an ascertain threshold, a percentage outliers take of the training samples is assumed, such as 2%-5%. One limitation of this method is that it is usually computationally intensive. Because the size of kernel matrix is the square of the number of training samples which is generally very large in real situations, it becomes much infeasible to maintain a large memory to store the huge matrix and expend high computation to eigen-decompose the matrix in each step. In addition, this method recognizes the outliers in an explicit way, i.e. each training sample is considered to be either a true data or an outlier, which is not reasonable.

Among the methods for robust PCA [7],[9],[10],[11], we focus on Xu et al.'s method [7]. In [7], the energy function for original PCA is first generalized by adding a binary decision field so that outliers are dealt with explicitly in order to enhance the robustness. Then a Gibbs distribution is defined using the generalized energy function, and the marginal distribution is also obtained which defined an effective energy function. In this way, the question is translated into the maximization of the marginal distribution, and a self-organizing rule for robust PCA is developed finally,

In this paper, a novel *Iterative Robust KPCA* (IRKPCA) is presented based on the research of Xu et al. [7] by kernellizing the adaptive rule of computing principal components. The resulting method enhances the robustness by restraining the effect of outliers efficiently. This is achieved by adding a weight term in the update rule which considers the outliers implicitly and smoothly. Furthermore, the proposed algorithm has three accessional merits: first, by the virtue of iterative computation, no large memory and direct eigen-decomposition are required to deal with the huge kernel matrix; second, it preserves the nonlinearity property of KPCA; third, it shows the potential of expansibility to incremental version.

The rest of this paper is organized as follows. Section 2 introduces briefly the original PCA, Xu et al.'s robust PCA and the standard KPCA. Section 3 formulates our Iterative Robust Kernel PCA algorithm. In Section 4, experimental results on synthetic data are presented. Finally, conclusions are given in Section 5.

## 2. Previous work

### 2.1. Principal component analysis

Given a data set with $l$ samples $x_i \in \Re^n$ and zero mean $(1/l)\sum_{i=1}^{l} x_i = 0$, $i$=1,...,$l$, classical PCA is to solve the following optimization problem:

$$J(w) = (1/l)\sum_{i=1}^{l} \| x_i - u_i \|^2 \qquad (1)$$

where $u=wy$ and $y=w^T x$. This is equivalent to solve the eigenvalue problem

$$\lambda w = Cw \qquad (2)$$

for eigenvalues $\lambda \geq 0$ and eigenvectors $w \in \Re^n \backslash 0$. $C$ is the covariance matrix $C = (1/l)\sum_{i=1}^{l} x_i x_i^T$.

While efficient and reliable numerical methods are discussed representatively in [12], adaptive approaches to PCA analysis are discussed e.g. in [1],[13]. From the computational point of view, it can be more advantageous to solve the eigenvalue problem (2) by iterative or adaptive methods which do not need to store and calculate the matrix $C$ directly. This is particularly useful when the matrix size is large. Another drawback of PCA is that it is sensitive to outliers. In contrast to PCA, robust PCA is robust against outliers.

### 2.2. Robust PCA

*Robust PCA* has been studied for many years, and many algorithms have been presented [7],[9],[10],[11]. In this paper, we fix our attention on the method of Xu et al. [7]. We review it here briefly.

To consider the effect of outliers, [7] generalizes the energy function defined in (1) to the following formula by adding a penalty item, which indicates the energy portion contributed by outliers:

$$E(V, w) = \sum_{i=1}^{l} V_i z(x_i, w) + \eta \sum_{i=1}^{l} (1 - V_i) \qquad (3)$$

where $z(x_i, w) = \| x_i - u_i \|^2$, $V$ is a binary field with $V_i \in 0,1$, $i$=1,...,$l$, and $\eta$ is a scalar threshold.

In (3), $z(x_i,w)$ is the energy portion contributed by $x_i$. $V_i$ can be considered as a decision indicator for deciding whether $x_i$ is a normal sample or an outlier. When $z(x_i,w) < \eta$, $V_i$ should be set to 1 as it is more reasonable to consider $x_i$ as a true sample and set to 0 otherwise.

It is not easy to minimize $E(V,w)$ with respect to $V$ and $w$ simultaneously because it is a mixture of discrete and continuous optimization. This is then achieved by defining a Gibbs distribution $P(V, w) = (1/Z)e^{-\beta E(V, w)}$ and then maximizing the marginal probability distribution $P_{\text{margin}}(w) = e^{-\beta E_{\text{eff}}(w)} / (Ze^{N\beta\eta})$ by averaging each $V_i$. Finally, the problem is equivalent to maximize the generalized energy function

$$E_{\text{eff}}(w) = -\frac{1}{\beta} \sum_i \log\{1 + e^{-\beta[z(x_i, w) - \eta]}\} \qquad (4)$$

$E_{\text{eff}}(w)$ can be regarded as a generalization of a robust redescending *M*-estimators [14] to the PCA problem. In (4) each item in the summation is close to $z(x_i,w)$ if it has a small value, but becomes constant as $z(x_i,w)\to\infty$. In this way, outliers which are more likely to yield large $z(x_i,w)$ are restrained effectively while the true samples yielding small $z(x_i,w)$ are affected very slightly.

The robust adaptive rules to compute first $k$ principal components are

$$w_j(t+1) = w_j(t) + \alpha(t)\cdot\frac{1}{1+e^{\beta(z(x_j(t)),w_j(t)-\eta)}} \cdot (x^j(t)y_j(t) - w_j(t)y_j(t)^2) \tag{5}$$

$$x^1(t) = x(t) \tag{6}$$

$$x^j(t) = x(t) - \sum_{r<j} w_r(t)y_r(t),\ j=2,\ldots,k \tag{7}$$

$$y_j(t) = w_j(t)^T x^j(t) \tag{8}$$

where $x(t)$ is the sample selected randomly from $l$ input examples at time $t$. $x^j(t)$, which is obtained by removing first $j$-1 reconstruction images, is the residual portion of input $x(t)$ to compute the $j$th vector $w_j(t)$, $\alpha(t)$ is the learning rate, $\beta$ is the Gibbs parameter and $\eta$ is the penalty threshold defined in (3). Formally, the update rules (5)-(8) can be considered as the generalization of the unrobust $k$-PCA rule GHA [15].

### 2.3. Kernel PCA

As stated in section 1, linear PCA fails to represent the underlying nonlinear structure of data, for it takes only second-order correlations into account. As a natural nonlinear extension of PCA, *Kernel PCA* (i.e. KPCA) computes the principal components in a possibly high-dimensional feature space $\mathcal{H}$ which is mapped from the input data space by a nonlinear map:

$$\begin{aligned}\Phi:\ & \Re^n \to \mathcal{H}\\ & x \mapsto \Phi(x)\end{aligned}$$

The feature space $\mathcal{H}$ is also called *Reproducing Kernel Hilbert Space* (RKHS).

Since the dimensionality of the feature space $\mathcal{H}$ may be very high (possibly infinite), it is infeasible to carry out the PCA analysis directly. Kernel techniques are then introduced to avoid this difficulty, which enable us to compute the inner product without having to evaluate the map explicitly.

Without loss of generality, we assume that samples are centered in the feature space, i.e. $\sum_{i=1}^{l}\Phi(x_i)=0$. Then the covariance matrix of the mapped samples becomes

$$C = \frac{1}{l}\mathbf{\Phi\Phi}^T \tag{9}$$

where $\mathbf{\Phi}=[\Phi(x_1),\ldots,\Phi(x_l)]$. We now have to find the eigenvalues $\lambda \geq 0$ and eigenvectors $w\in\mathcal{H}\setminus 0$ satisfying

$$\lambda w = \mathbf{C}w \tag{10}$$

It is easy to know that each eigenvector $w$ with $\lambda \geq 0$ lies in the space spanned by the training samples $\Phi(x_1),\ldots,\Phi(x_l)$, therefore, $w$ can be linearly expended by

$$w = \sum_{i=1}^{l} a^i\Phi(x_i) \tag{11}$$

Substituting (9) and (11) into (10), and denoting the *Gram matrix* as $\mathbf{K}=\mathbf{\Phi\Phi}^\mathbf{T}$, we obtain $l\lambda\mathbf{K}a=\mathbf{K}^2a$. This leads to the following equivalent *kernel eigenvalue problem*

$$l\lambda a = \mathbf{K}a \tag{12}$$

The above derivation assumes that all projected samples $\Phi(x)$ are centered in $\mathcal{H}$. When this is not true, the Gram matrix $\mathbf{K}$ should be replaced by

$$\hat{\mathbf{K}} = \mathbf{K} - \mathbf{1}_l\mathbf{K} - \mathbf{K}\mathbf{1}_l + \mathbf{1}_l\mathbf{K}\mathbf{1}_l \tag{13}$$

where $\mathbf{1}_l=(1/l)_{l\times l}$.

Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_l$ denote the eigenvalues of $\mathbf{K}$ in (12), and $a_1,a_2,\ldots,a_l$ the corresponding complete set of eigenvectors, with $\lambda_k$ being the last nonzero eigenvalue. We normalize $a_1,a_2,\ldots,a_l$ by requiring that the corresponding vectors in $\mathcal{H}$ be normalized, that is

$$(w_j\cdot w_j)=1,\ \text{for all } j=1,\ldots,k$$

By virtue of equations (11) and (12), this becomes to

$$\begin{aligned}1 &= \sum_{p,q=1}^{l} a_j^p a_j^q(\Phi(x_p)\cdot\Phi(x_q))\\ &= \sum_{p,q=1}^{l} a_j^p a_j^q\mathbf{K}_{pq} = (a_j\cdot\mathbf{K}a_j) = \lambda_j(a_j\cdot a_j)\end{aligned} \tag{14}$$

Now, we can extract the nonlinear principal components of the test sample $x$:

$$(w\cdot\Phi(x)) = \sum_{i=1}^{l} a_i(\Phi(x_i)\cdot\Phi(x)) = \sum_{i=1}^{l} a_i k(x_i,x) \tag{15}$$

which is achieved by using kernel function without the expensive operation that explicitly projects samples to the feature space $\mathcal{H}$.

There are great choices of kernel functions, and the best selection of kernel remains an open topic. Presently, polynomial kernel, radial basis function kernel and sigmoid kernel are widely used.

## 3. Proposed method for Robust KPCA

### 3.1. Derivation of Iterative Robust Kernel PCA

Although KPCA has been used in several applications successfully and showed better performance than PCA, it suffers from the sensitivity to outliers in the same way. In this section, we present a novel robust method which is able to compute kernel principal components.

Firstly, we rewrite the update rule (5) of Xu et al.'s method for robust PCA to the matrix form:

$$\begin{aligned}\mathbf{W}(t+1) &= \mathbf{W}(t)\\ &+\alpha(t)(x(t))y(t)^T - \mathrm{UT}[y(t)y(t)^T]\mathbf{W}(t))\mathbf{Z}(t)\end{aligned} \tag{16}$$

where $\mathbf{W}(t)$ is a $n\times k$ matrix, with the $j$th column being the eigenvector corresponding to the $j$th largest eigenvalue when the iteration converges, $y(t)=\mathbf{W}(t)^T x(t)$, UT(•) sets all elements below the diagonal of is matrix argument to zero, thus making it upper triangular and $\mathbf{Z}(t)$ is a diagonal matrix. For notational convenience, we denote the function

$$f(z) = \frac{1}{1+e^{\beta(z-\eta)}} \tag{17}$$

Consequently, the $j$th value on the diagonal of $\mathbf{Z}(t)$ can be expressed as $f(z(x^j,w_j))$. $z(x^j,w_j)$ and $x^j$ orginate from (3) and (7) respectively.

$$f(z(x^j,w_j)) = \frac{1}{1+e^{\beta(z(x^j(t),w_j(t))-\eta)}} \tag{18}$$

Update rule (16) can be represented in the feature space $\mathcal{H}$ as

$$\begin{aligned}\mathbf{W}(t+1) &= \mathbf{W}(t)+\alpha(t)\\ &(\Phi(x(t))y(t)^T - \mathrm{UT}[y(t)y(t)^T]\mathbf{W}(t))\mathbf{Z}(t)\end{aligned} \tag{19}$$

where the input is the mapped data $\Phi(x(t))$, the columns of $\mathbf{W}(t)$ are vectors in $\mathcal{H}$ and $y(t)=\mathbf{W}(t)^T\Phi(x(t))$.

From the derivation of KPCA, it is known that $w(t)$ can be expanded in the projected samples $\Phi(x_i)$, as a result,

$$\mathbf{W}(t) = \mathbf{\Phi}\mathbf{A}(t) \tag{20}$$

where $\mathbf{A}(t)=[a_1(t),a_2(t),...,a_l(t)]$ is the expansion coefficients matrix with the size of $l\times k$. Using this representation, the update rule becomes

$$\begin{aligned}\mathbf{\Phi}\mathbf{A}(t+1) &= \mathbf{\Phi}\mathbf{A}(t)+\alpha(t)\\ &(\Phi(x(t))y(t)^T - \mathrm{UT}[y(t)y(t)^T]\mathbf{\Phi}\mathbf{A}(t))\mathbf{Z}(t)\end{aligned} \tag{21}$$

The mapped pattern $\Phi(x(t))$ can also be represented as $\Phi(x(t))=\mathbf{\Phi}b(t)$, where $b(t)=[0,0,...,1,...,0]^T$ is a vector in $\Re^l$ with only the $i$th element is 1 when $x_i$ in input data set is selected as $x(t)$ at time $t$. In this way, the rule can be rewritten solely in terms of the expansion coefficients as:

$$\begin{aligned}\mathbf{A}(t+1) &= \mathbf{A}(t)\\ &+\alpha(t)(b(t)y(t)^T - \mathrm{UT}[y(t)y(t)^T]\mathbf{A}(t))\mathbf{Z}(t)\end{aligned} \tag{22}$$

where

$$x(t) = x_i \tag{23}$$

$$y(t) = \mathbf{A}^T(t)\mathbf{K}_{:,i} \tag{24}$$

$$\mathbf{Z}(t) = \mathrm{diag}([f(z^1(t)), f(z^2(t)),..., f(z^k(t))]) \tag{25}$$

$$\begin{aligned}z^j(t) &= \| \Phi(x_i) - \sum_{p\le j} w_p y_p \|^2\\ &= k(x_i,x_i) - 2\sum_{p\le j}\sum_{q=1}^{l} k(x_i,x_q)a_p^q y_p\\ &-\sum_{p\le j}\sum_{q\le j} y_p a_p^T \mathbf{K} a_q y_q\\ &= \mathbf{K}_{i,i} - 2\sum_{p\le j}\mathbf{K}_{i,:}a_p y_p - (\sum_{p\le j} a_p y_p)^T \mathbf{K}\\ &(\sum_{p\le j} a_p y_p)\end{aligned} \tag{26}$$

In the same way, the kernel matrix $\mathbf{K}$ should be preprocessed according to (13) such that the data are centered in space $\mathcal{H}$.

The rules presented above provide a practical implementation of the robust PCA discussed in section 2.2. in space $\mathcal{H}$. Furthermore, it could be regarded as a robust extension of the *Kernel Hebbian Algorithm* (KHA) [16], which calculates kernel principal components iteratively. During the implementation, $\mathbf{A}$ should be randomly initialized and the update will converge if parameters $\alpha$, $\beta$ and $\eta$ are selected properly. For the details of the convergence of KHA, refer to [16].

### 3.2. Parameters selection

There are three parameters in our procedure, the learning rate $\alpha$, the Gibbs parameter $\beta$ and the scalar penalty threshold $\eta$, which need to be selected reasonably. The larger learning rate $\alpha$ is, the faster the learning and the bigger the fluctuation in the learning process. The inverse of $\beta$ decides the sharpness of the Gibbs distribution and the sensitivity of the weighting function (17). In company with $\beta$, $\eta$ determines the penalty weight during the update process according to the degree how much a sample is considered as an outlier.

Some suggestions have already been made about the selection of the learning rate $\alpha$. [7],[15] stated that $\alpha(t)$ should decrease to zero as $t\to\infty$ and satisfies certain conditions, e.g.

$$\sum_t \alpha(t)=\infty,\ \sum_t \alpha(t)^q<\infty \quad \text{for some } q>1, \qquad (27)$$

when computing principal components using GHA and iterative robust PCA (5) respectively. Typically, it was chosen as $\alpha(t)=1/t$. In [16], it was also proven that the local convergence of KHA followed from the local convergence of GHA for learning rate $1/t$, in $\Re^l$.

$\beta$ was recommended to start at a small value and increase with a rate of $O(\ln t)$ and $\eta$ change according to $\beta$ i.e. a small $\eta$ for small $\beta$ as the penalty weight was not sensitive and a large $\eta$ for a larger $\beta$. to restrain the true samples to be considered as outliers [7].

In our experiments, we selected fixed values for $\alpha$, $\beta$ and $\eta$ for simplicity.

## 4. Simulation experiments

In this section, we have evaluated the robustness and effectiveness of our Iterative Robust KPCA (IRKPCA) algorithm compared to the standard KPCA. We conducted experiments on the artificial data described in literature [2]. The data consisted of three clusters in two-dimensional space. Each cluster had 100 samples fallowing Gaussian distribution with standard deviation 0.1 and means [-0.5 -0.2], [0 0.6], [0.5 0] respectively. These 300 samples constitute the *true data set*.

For the purpose of comparison, 10 outlying samples were also generated randomly with a distribution different from Gaussian, which constituted an *outliers set*.

In our experiments, polynomial kernel of degree 2 was used. As discussed in section 3.2, three parameters should be set properly: the learning rate $\alpha$, Gibbs parameter $\beta$ and penalty threshold $\eta$. We fixed $\alpha$=0.001, $\beta$=5 and $\eta$=3 for simplicity.

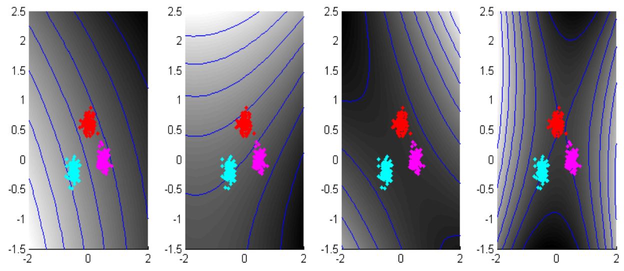

Figure 1.Contour lines of constant value of the first four PCs for the *true data set* obtained from KPCA.

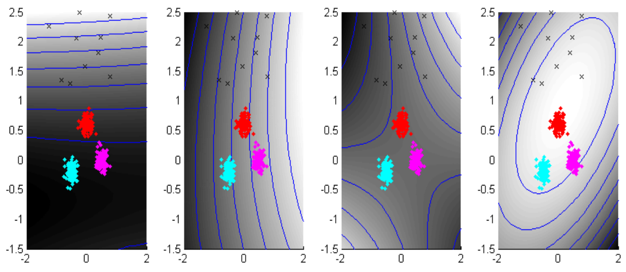

Figure 2. Contour lines of constant value of the first four PCs for the data set including *outliers set* obtained from KPCA.

When without the presence of outliers, KPCA finds the kernel principal components correctly. Figure 1 shows the first four PCs of the *true data set* extracted by standard KPCA. From left to right, the PCs are shown in the order of decreasing eigenvalue size. The results illuminate the advantage of using nonlinear kernel in the facet of reflecting the data structure. The first two PCs separate the three clusters nicely, and the PCs 3-4 split up the clusters into halves.

When the data are spoiled by outliers, KPCA exposes its instability. Figure 2 is the result of KPCA on the data spoiled by 10 outliers (about 3%). The PCs obtained are very different from the desired results and cannot split the three clusters properly.

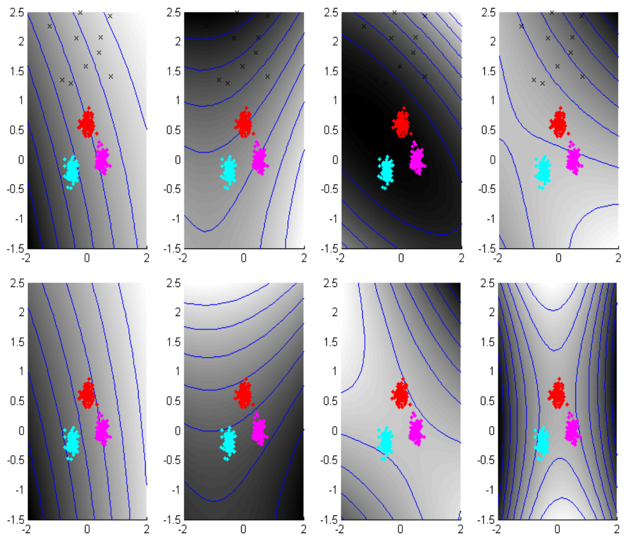

Figure 3. Results of the first four PCs obtained from IRKPCA for dataset with (the top row) and without (bottom row) the *outliers*

Our proposed method performs more accurately in this situation. Figure 3 is the results of our IRKPCA. As figures in the first row show, IRKPCA can restrain the effect of outliers effectively and the PCs obtained are very close to the real ones, especially the first two components. It is also equivalent to KPCA when no outliers are presented, just as figures in the second row show.

Figure 4 is the comparative result of the eigenvalue distributions obtained in figures 1-3. Obviously, the proposed method outperforms KPCA method in the aspect of robustness.

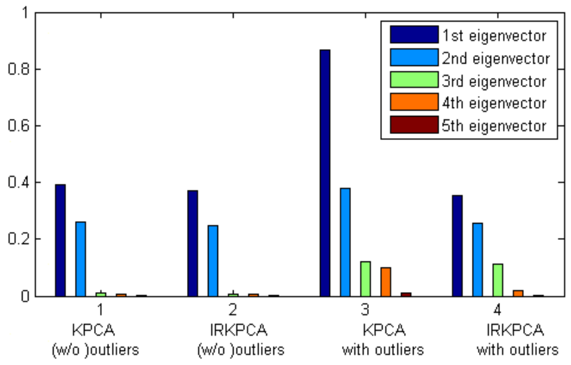

Figure 4. Comparative result of eigenvalue distributions corresponding to figures 1-3

## 5. Conclusions and future work

In this paper, we have proposed a novel iterative method to compute kernel principal components. This method enhances the robustness against outliers in the data by adding a weight term in the update rule. Experimental results show that the proposed method is superior to the standard KPCA in terms of robustness. By the virtue of iterative computation, large memory is not required to maintain the huge kernel matrix, which implies the possibility of practical implements. Therefore, more experiments should be carried out to examine the computational efficiency. In addition, the incremental version of our method is worthy of investigation.

## Acknowledgements

This work is supported by the National Natural Science Foundation of China under grand number 60605005 and the Open Fund of National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences under grand number 2005-2.

## References

[1] K. Fukunaga, Introduction to Statistical Pattern Recognition: Academic Pr, 1990.

[2] B. Scholkopf, A. Smola, and K. R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," Neural Computation, vol. 10, pp. 1299-1319, Jul 1 1998.

[3] S. Mika, B. Scholkopf, A. Smola, K. R. Muller, M. Scholz, and G. Ratsch, "Kernel PCA and de-noising in feature spaces." vol. 11, 1999, pp. 536-542.

[4] B. Mak, J. T. Kwok, and S. Ho, "Kernel eigenvoice speaker adaptation," IEEE Transactions on Speech and Audio Processing, vol. 13, pp. 984-992, Sep 2005.

[5] B. Y. Liu and J. Zhang, "Eigenspectra versus eigenfaces: Classification with a kernel-based nonlinear representor," Advances in Natural Computation, Pt 1, Proceedings, vol. 3610, pp. 660-663, 2005.

[6] J. Yang, A. F. Frangi, J. Y. Yang, D. Zhang, and Z. Jin, "KPCA plus LDA: A complete kernel fisher discriminant framework for feature extraction and recognition," IEEE Transactions on PAMI, vol. 27, pp. 230-244, Feb 2005.

[7] L. Xu and A. L. Yuille, "Robust principal component analysis by self-organizing rules based on statistical physics approach," IEEE Transactions on Neural Networks, vol. 6, pp. 131-143, 1995.

[8] C. D. Lu, T. Y. Zhang, X. Z. Du, and C. P. Li, "A robust kernel PCA algorithm," in Proceedings of 2004 International Conference on Machine Learning and Cybernetics, 2004, pp. 3084-3087.

[9] M. Brand, "Incremental singular value decomposition of uncertain data with missing values," European Conference on Computer Vision, pp. 707-720, 2002.

[10] F. De La Torre and M. J. Black, "A framework for robust subspace learning," International Journal of Computer Vision, vol. 54, pp. 117-142, 2003.

[11] Y. Li, "On incremental and robust subspace learning," Pattern Recognition, vol. 37, pp. 1509-1518, 2004.

[12] G. H. Golub and C. F. Van Loan, Matrix Computation: The Johns Hopkins University Press Baltimore, 1996.

[13] K. I. Diamantaras and S. Y. Kung, Principal component neural networks: theory and applications: John Wiley & Sons, Inc. New York, NY, USA, 1996.

[14] P. J. Huber, "Robust Statistic," NewYork: Wiley, 1981.

[15] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," Neural Networks, vol. 2, pp. 459-473, 1989.

[16] K. Kwang In, M. O. Franz, and B. Scholkopf, "Iterative Kernel Principal Component Analysis for Image Modeling," PAMI, IEEE Transactions on, vol. 27, pp. 1351-1366, 2005.