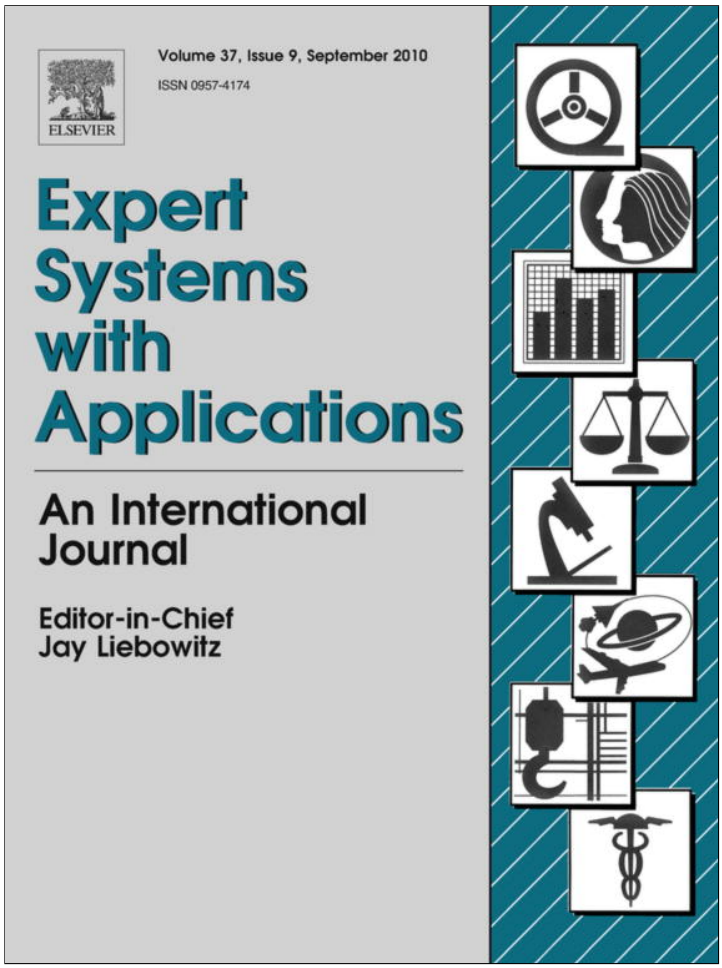

Expert Systems with Applications 37 (2010) 6531–6537

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Approximations of the standard principal components analysis and kernel PCA

Rui Zhang [a], Wenjian Wang [b,*], Yichen Ma [c]

[a] School of Science, Shandong University of Technology, Zibo 255049, PR China
[b] School of Computer and Information Technology, Key Laboratory of Computational Intelligence & Chinese Information Processing of Ministry of Education, Shanxi University, Taiyuan 030006, PR China
[c] School of Science, Xi'an Jiaotong University, Xi'an 710049, PR China

ARTICLE INFO

Keywords:
PCA
KPCA
Kernel space
Orthogonal projection

ABSTRACT

Principal component analysis (PCA) is a powerful technique for extracting structure from possibly high-dimensional data sets, while kernel PCA (KPCA) is the application of PCA in a kernel-defined feature space. For standard PCA and KPCA, if the size of dataset is large, it will need a very large memory to store kernel matrix and a lot of time to calculate eigenvalues and corresponding eigenvectors. The aim of this paper is to learn linear and nonlinear principal components by using a few partial data points and determine which data points can be used. To verify the performance of the proposed approaches, a series of experiments on artificial datasets and UCI benchmark datasets are accomplished. Simulation results demonstrate that the proposed approaches can compete with or outperform the standard PCA and KPCA in generalization ability but with much less memory and time consuming.

1. Introduction

Principal component analysis is a powerful technique for extracting structure from possibly high-dimensional data sets, and it has received much more attentions in many literatures (Cadima & Jolliffe, 1995; Croux & Haesbroeck, 2000; Higuchi & Eguchi, 2004; Jolliffe, 1986, 1995; Jolliffe & Uddin, 2003; McCabe, 1984; Misra et al., 2002; Schökopf, Smolar, & Muller, 1998; Shawe-Taylor & Cristianini, 2005; Suykens, Van Gestel, Vandewalle, & De Moor, 2003; Tao, Wu, & Wang, 2007; Vines, 2000). From the view point of mathematics, PCA is an orthogonal transformation of the coordinate system in which the data are described. The new coordinate values, by which the data are represented, are called principal components. It is often the case that a small number of principal components are sufficient to account for the main structure embedded in data. KPCA is the application of PCA in a kernel-defined feature space.

For standard PCA and KPCA, different data points will play different roles in determining all principal components. In general, the closer the data points to the mean center, the less important the data points contributing to PCA due to the norms of their projections to an arbitrary direction are very small. For a large scale data set, it needs a very large memory to store kernel matrix and a lot of time to calculate eigenvalues and corresponding eigenvectors. In our study, we can determine all the principal components by using a few partial data points. The aim of this paper is to solve the key problem, i.e., how to choose data points from training data so as to obtain all the principal components. In the proposed algorithms, we select partial data points from the training dataset under the same threshold with the standard PCA and KPCA, and then obtain equivalent eigenvalues and eigenvectors with that of PCA and KPCA. To verify the performance of the proposed approaches, a series of experiments on artificial datasets and UCI benchmark datasets are carried out. Simulation results demonstrate that the proposed approaches can compete with or outperform the standard PCA and KPCA in generalization ability but with much less memory and time consuming.

The paper is organized as follows. In Section 2, the standard PCA and KPCA is described briefly. The approaches and corresponding algorithms for approximating the PCA and KPCA by using partial data points are illustrated in Section 3. Simulation experiments and discussions are presented in Section 4. The last section concludes the proposed works.

2. The standard PCA and kernel PCA

2.1. The standard PCA

PCA takes an initial subset of the principal axes of the training data and projects the data (both training and testing data) into the space spanned by the set of eigenvectors. Data are projected into the subspace spanned by the first k eigenvectors of the covariance matrix of the training set for some $k < l$. The new coordinates are known as the principal coordinates with the eigenvectors referring to the principal axes.

* Corresponding author. Tel.: +86 351 7017567; fax: +86 351 7018176.
E-mail address: wjwang@sxu.edu.cn (W. Wang).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.02.133

The primal PCA algorithm performs the following computation:

Input:
- A data set $S = \{x_i\}_{i=1}^{l}$, dimension k.

Process:
- $\mu = \frac{1}{l}\sum_{i=1}^{l}\mathbf{x}_i$
- $C = \frac{1}{l}\sum_{i=1}^{l}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)'$
- $[U, \bigwedge] = eig(lC)$
- $\tilde{x}_i = U_k' x_i,\ i = 1, \ldots, l$

Output:
- Transformed data $\widetilde{S} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_l\}$

2.2. The standard KPCA

KPCA is the application of PCA in a kernel-defined feature space by the dual representation. We use U_k to denote the subspace spanned by the first k eigenvectors in the feature space. We can compute the k-dimensional vector projection of new data into this subspace as

$$P_{U_k}(\phi(x)) = \left(\sum_{i=1}^{l}\alpha_i^j k(x_i, x)\right)_{j=1}^{k} \tag{1}$$

where

$$\alpha^j = \lambda_j^{-1/2} v_j$$

is given in terms of the corresponding eigenvector and eigenvalue of the kernel matrix. Eq. (1) forms the basis of kernel PCA.

The KPCA algorithm performs the following computation:

Input:
- A data set $S = \{x_i\}_{i=1}^{l}$, dimension k.

Process:
- $K_{ij} = k(x_i, x_j),\ i,j = 1, \ldots, l$
- $K = K - \frac{1}{l}\vec{j}\vec{j}'K - \frac{1}{l}K\vec{j}\vec{j}' + \frac{1}{l^2}(\vec{j}'K\vec{j})\vec{j}\vec{j}'$
- $[V, \bigwedge] = eig(K)$
- $\alpha^j = \lambda_j^{-1/2} v_j,\ j = 1, \ldots, k$
- $\tilde{x} = \left(\sum_{i=1}^{l}\alpha_i^j k(x_i, x)\right)_{j=1}^{k}$

Output:
- Transformed data $\widetilde{S} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_l\}$

where $\vec{j}$ is the all 1s vector.

An alternative characterization of the principal components (or principal axes) of a dataset will be important for the analysis of KPCA. We first introduce some additional notations. We use $P_U(\phi(x))$ to denote the orthogonal projection of an embedded point $\Phi(x)$ into the subspace U. The difference

$$P_U^{\perp} = \Phi(x) - P_U(\Phi(x))$$

is the projection into the orthogonal subspace and it refers to the residual. We will typically assess the quality of a projection by the average of the squared norms of the residuals of the training data

$$\frac{1}{l}\sum_{i=1}^{l}\|P_U^{\perp}\|^2$$

The next theorem shows that using the space spanned by the first k principal components of the covariance matrix can minimize this quantity.

Theorem 1. *Given a training set S with covariance matrix C, the orthogonal projection $P_{U_k}(\Phi(x))$ into the subspace U_k spanned by the first k eigenvectors of C is the k-dimensional orthogonal projection minimizing the average squared distance between each training point and its image, in other words, U_k solves the optimization problem*

$$\begin{aligned} &\min_{U} \quad J^{\perp}(U) = \sum_{i=1}^{l}\|P_U^{\perp}(\Phi(x_i))\|_2^2 \\ &s.t. \quad dim\, U = k \end{aligned} \tag{2}$$

Furthermore, the value of $J^{\perp}(U)$ at the optimum is given by

$$J^{\perp}(U) = \sum_{i=k+1}^{N}\lambda_i \tag{3}$$

where $\lambda_1, \ldots, \lambda_N$ are the eigenvalues of the matrix lC in decreasing order.

3. Approximations of the standard PCA and KPCA

In standard PCA and KPCA, all data points are used. For a large scale data set, it needs a very large memory to store kernel matrix and a lot time to calculate eigenvalues and corresponding eigenvectors. For the algorithms of PCA and KPCA, we can see that different data points will play different roles. In general, the closer the data points to the mean center, the less important the data points contributing to PCA or KPCA because their projections to an arbitrary direction are very small. So we can discard those data points which are close to the mean center and use the rest data points to approximate the standard PCA and KPCA.

3.1. Approximation of PCA

In this section, we suppose the training dataset $S = \{x_i\}_{i=1}^{l}$ are centered. For standard PCA,

$$\begin{aligned} J^{\perp}(U_k) &= \sum_{i=1}^{l}\|P_{U_k}^{\perp}(x_i)\|_2^2 = \sum_{i=1}^{l}\|x_i - P_{U_k}(x_i)\|_2^2 \\ &= \sum_{i=1}^{l}\|x_i\|_2^2 - \sum_{i=1}^{l}\|P_{U_k}(x_i)\|_2^2 = \sum_{i=1}^{N}\lambda_i - \sum_{i=1}^{k}\lambda_i \end{aligned}$$

where $\lambda_1, \ldots, \lambda_N$ are the eigenvalues of the matrix lC in decreasing order.

In the algorithm of standard PCA, the dimension k is not given in advance. It can be determined according to the following equation:

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{N}\lambda_i} \leqslant \alpha$$

or

$$\frac{\sum_{i=1}^{k}\lambda_i}{\sum_{i=1}^{N}\lambda_i} \geqslant 1 - \alpha$$

where α is a threshold, and it is often less than 0.30. $1 - \alpha$ is often called contribution ratio.

In our proposed approach, we select a subset S_k which contains m data points. Without loss of generality, we suppose training dataset $S = \{x_i\}_{i=1}^{l}$ is arranged in decreasing order according to the norms of $\{x_i\}$.

Denote

$$J^{\perp}(U_k) \triangleq \sum_{i=1}^{l}\|x_i\|^2 - \sum_{i=1}^{m}\|P_{U_k}x_i\|^2$$

We have the following result.

Theorem 2. *Given a training set $S = \{x_i\}_{i=1}^{l}$ and an arbitrary threshold α, there exits a subset S_m which consists at least $m \in \mathbb{N}$ $(m \leqslant l)$ data points of S with covariance matrix C_m, and one k-dimensional space U_k spanned by the first k eigenvectors of C_m, such that*

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{l}\|x_i\|^2} \leqslant \alpha$$

Proof. Given an arbitrary constant γ $(0 < \gamma < 1)$, we can uniquely determine the least constant $m \in \mathbb{N}$ $(m \leqslant l)$ according to the following equation:

$$\frac{\sum_{i=1}^{m}\|x_i\|^2}{\sum_{i=1}^{l}\|x_i\|^2} \geqslant 1-\gamma$$

Let $C_m = \frac{1}{m}\sum_{i=1}^{m} x_i x_i'$, $\beta_1, \beta_2, \ldots, \beta_N$ be the eigenvalues of matrix mC_m in decreasing order and U_k be a k-dimensional space spanned by the first k eigenvectors of mC_m.

Thus

$$\sum_{i=1}^{N}\beta_i = \sum_{i=1}^{m}\|x_i\|^2 \geqslant (1-\gamma)\sum_{i=1}^{l}\|x_i\|^2$$

Since $P_{U_k}(x_i)$ is an orthogonal projection, it follows Pythagoras's theorem, i.e.,

$$J_0^{\perp}(U_k) \triangleq \sum_{i=1}^{m}\|P_{U_k}^{\perp}x_i\|^2 = \sum_{i=1}^{m}\|x_i\|^2 - \sum_{i=1}^{m}\|P_{U_k}x_i\|^2 = \sum_{i=1}^{N}\beta_i - \sum_{i=1}^{k}\beta_i$$

Let

$$\frac{J_0^{\perp}(U_k)}{\sum_{i=1}^{N}\beta_i} = 1 - \frac{\sum_{i=1}^{k}\beta_i}{\sum_{i=1}^{N}\beta_i} \leqslant \gamma$$

or

$$\frac{\sum_{i=1}^{k}\beta_i}{\sum_{i=1}^{N}\beta_i} \geqslant 1-\gamma$$

Then the least value of k $(k \in \mathbb{N})$ can be determined.

$$\begin{aligned} J^{\perp}(U_k) &= \sum_{i=1}^{l}\|x_i\|^2 - \sum_{i=1}^{m}\|P_{U_k}x_i\|^2 = \sum_{i=1}^{l}\|x_i\|^2 - \sum_{i=1}^{k}\beta_i \\ &\leqslant \sum_{i=1}^{l}\|x_i\|^2 - (1-\gamma)\sum_{i=1}^{N}\beta_i = \sum_{i=1}^{l}\|x_i\|^2 - (1-\gamma)^2\sum_{i=1}^{l}\|x_i\|^2 \\ &= \gamma(2-\gamma)\sum_{i=1}^{l}\|x_i\|^2 < 2\gamma\sum_{i=1}^{l}\|x_i\|^2 \end{aligned}$$

Hence

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{l}\|x_i\|^2} < 2\gamma$$

In order to obtain

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{l}\|x_i\|^2} < 2\gamma \leqslant \alpha$$

it needs

$$\gamma \leqslant \frac{\alpha}{2}$$

This completes the proof. □

In sequence, the proposed approximation algorithm of PCA (APCA) is summarized as following.

Input:
- A data set $S = \{x_i\}_{i=1}^{l}$, threshold α.

Process:
- $\mu = \frac{1}{l}\sum_{i=1}^{l}\mathbf{x}_i$
- $S_0 \triangleq \{x_i - \mu\}_{i=1}^{l}$ sorted in decreasing order according to the norm of $x_i - \mu$
- Determining the least number m $(m \in \mathbb{N})$ according to the following equation

$$\frac{\sum_{i=1}^{m}\|x_i-\mu\|^2}{\sum_{i=1}^{l}\|x_i-\mu\|^2} \geqslant 1-\frac{\alpha}{2}$$

- $C_m = \frac{1}{m}\sum_{i=1}^{m}(\mathbf{x}_i-\mu)(\mathbf{x}_i-\mu)'$
- $[U, \bigwedge] = eig(mC_m)$
- Determining the least number k $(k \in \mathbb{N})$ according to the following equation

$$\frac{\sum_{i=1}^{k}\beta_i}{\sum_{i=1}^{N}\beta_i} \geqslant 1-\frac{\alpha}{2}$$

where $\beta_1, \beta_2, \ldots, \beta_N$ are the eigenvalues of matrix mC_m in decreasing order.
- $\tilde{x}_i = U_k' x_i,\ i = 1, \ldots, l$

Output:
- Transformed data $\widetilde{S} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_l\}$

3.2. Approximation of KPCA

Consider an embedding map

$$\Phi : x \in \mathbb{R}^N \mapsto \Phi(x) \in F \subseteq \mathbb{R}^n$$

The choice of the map Φ aims at converting the nonlinear relations into linear ones. The map Φ is used to recode dataset S as $\widehat{S} = \{\Phi(x_i)\}_{i=1}^{l}$. The center of the set $\widehat{S}$ is $\Phi_{\widehat{S}} = \frac{1}{l}\sum_{i=1}^{l}\Phi(x_i)$. With all points in the feature space we will not have an explicit vector representation for these points. Despite this apparent inaccessibility of the point $\Phi_{\widehat{S}}$, we can compute the distance of the image of a point x_i from the center of mass $\Phi_{\widehat{S}}$

$$\begin{aligned} \|\Phi(x) - \Phi_{\widehat{S}}\|^2 &= \langle\Phi(x_i), \Phi(x_i)\rangle + \langle\Phi_{\widehat{S}}, \Phi_{\widehat{S}}\rangle - 2\langle\Phi(x_i), \Phi_{\widehat{S}}\rangle \\ &= k(x_i, x_i) + \frac{1}{l^2}\sum_{i,j=1}^{l}k(x_i, x_j) - \frac{2}{l}\sum_{j=1}^{l}k(x_i, x_j) \end{aligned} \tag{4}$$

Without loss of generality, we suppose training dataset $S = \{x_i\}_{i=1}^{l}$ is sorted in decreasing order according to $\|\Phi(x) - \Phi_{\widehat{S}}\|^2$, the corresponding center kernel matrix is denoted as $\widehat{K}$ which is calculated by

$$\widehat{K} = K - \frac{1}{l}\vec{j}\vec{j}'K - \frac{1}{l}K\vec{j}\vec{j}' + \frac{1}{l^2}(\vec{j}'K\vec{j})\vec{j}\vec{j}' \tag{5}$$

where K is kernel matric, $K_{ij} = \langle\Phi(x_i), \Phi(x_j)\rangle = k(x_i, x_j)$. For standard KPCA,

$$\begin{aligned} J^{\perp}(U_k) &= \sum_{i=1}^{l}\|P_{U_k}^{\perp}(\Phi(x_i))\|_2^2 = \sum_{i=1}^{l}\|\Phi(x_i) - P_{U_k}(\Phi(x_i))\|_2^2 \\ &= \sum_{i=1}^{l}\|\Phi(x_i)\|_2^2 - \sum_{i=1}^{l}\|P_{U_k}(\Phi(x_i))\|_2^2 = \sum_{i=1}^{N}\lambda_i - \sum_{i=1}^{k}\lambda_i \end{aligned}$$

where $\lambda_1, \ldots, \lambda_N$ are the eigenvalues of the kernel matrix $\widehat{K}$ in decreasing order.

In the algorithm of standard KPCA, the dimension k is also not given in advance. Similarly, it can be determined according to the following equation:

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{N}\lambda_i} \leqslant \alpha$$

or

$$\frac{\sum_{i=1}^{k}\lambda_i}{\sum_{i=1}^{N}\lambda_i} \geqslant 1-\alpha$$

where α is a threshold, and often less than 0.30.

In the proposed approach, we select a subset S_k which contains m data points.

Denoting

$$J^{\perp}(U_k) \triangleq \sum_{i=1}^{l}\|\Phi(x_i)\|^2 - \sum_{i=1}^{m}\|P_{U_k}\Phi(x_i)\|^2$$

We have the following result.

Theorem 3. *Suppose we perform PCA in the feature space, dataset $\widehat{S} = \{\Phi(x_i)\}_{i=1}^{l}$. Given an arbitrary threshold α, there exits a subset S_m which consists at least $m \in \mathbb{N}$ $(m \leqslant l)$ data points of $\widehat{S}$ with kernel matrix K_m that is made of the first m rows and first m columns of $\widehat{K}$, and one k-dimensional space U_k spanned by the first k eigenvectors of K_m, such that*

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{l}\|\Phi(x_i)\|^2} \leqslant \alpha$$

Proof. Given an arbitrary constant γ $(0<\gamma<1)$, we can uniquely determine the least constant $m \in \mathbb{N}$ $(m \leqslant l)$ according to the following equation:

$$\frac{\sum_{i=1}^{m}\|\Phi(x_i)\|^2}{\sum_{i=1}^{l}\|\Phi(x_i)\|^2} \geqslant 1-\gamma$$

Let $\beta_1, \beta_2, \ldots, \beta_m$ be the eigenvalues of matrix K_m in decreasing order and U_k be a k-dimensional space spanned by the first k eigenvectors of K_m,where K_m is made of the first m rows and first m columns of $\widehat{K}$.

Then

$$\sum_{i=1}^{N}\beta_i = \sum_{i=1}^{m}\|\Phi(x_i)\|^2 \geqslant (1-\gamma)\sum_{i=1}^{l}\|\Phi(x_i)\|^2$$

Since $P_{U_k}(\Phi(x_i))$ is an orthogonal projection, it follows from Pythagoras's theorem, i.e.,

$$\begin{aligned} J_0^{\perp}(U_k) \triangleq \sum_{i=1}^{m}\|P_{U_k}^{\perp}\Phi(x_i)\|^2 &= \sum_{i=1}^{m}\|\Phi(x_i)\|^2 - \sum_{i=1}^{m}\|P_{U_k}\Phi(x_i)\|^2 \\ &= \sum_{i=1}^{N}\beta_i - \sum_{i=1}^{k}\beta_i \end{aligned}$$

Let

$$\frac{J_0^{\perp}(U_k)}{\sum_{i=1}^{N}\beta_i} = 1 - \frac{\sum_{i=1}^{k}\beta_i}{\sum_{i=1}^{N}\beta_i} \leqslant \gamma$$

or

$$\frac{\sum_{i=1}^{k}\beta_i}{\sum_{i=1}^{N}\beta_i} \geqslant 1-\gamma$$

then the least value of k $(k \in \mathbb{N})$ can be determined.

$$\begin{aligned} J^{\perp}(U_k) &= \sum_{i=1}^{l}\|\Phi(x_i)\|^2 - \sum_{i=1}^{m}\|P_{U_k}\Phi(x_i)\|^2 \\ &= \sum_{i=1}^{l}\|\Phi(x_i)\|^2 - \sum_{i=1}^{k}\beta_i \leqslant \sum_{i=1}^{l}\|\Phi(x_i)\|^2 - (1-\gamma)\sum_{i=1}^{N}\beta_i \\ &= \sum_{i=1}^{l}\|\Phi(x_i)\|^2 - (1-\gamma)^2\sum_{i=1}^{l}\|\Phi(x_i)\|^2 \\ &= \gamma(2-\gamma)\sum_{i=1}^{l}\|\Phi(x_i)\|^2 < 2\gamma\sum_{i=1}^{l}\|\Phi(x_i)\|^2 \end{aligned}$$

Hence

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{l}\|\Phi(x_i)\|^2} < 2\gamma$$

In order to obtain

$$\frac{J^{\perp}(U_k)}{\sum_{i=1}^{l}\|\Phi(x_i)\|^2} < 2\gamma \leqslant \alpha$$

it needs

$$\gamma \leqslant \frac{\alpha}{2}$$

This completes the proof. □

The proposed approximation algorithm of KPCA (AKPCA) is summarized as follows.

Input:
- A data set $S = \{x_i\}_{i=1}^{l}$, kernel function $k(x,y)$, threshold α.

Process:
- Sorting S in decreasing order according to $\|\Phi(x_i) - \Phi_{\widehat{S}}\|^2$.
- Computing $\widehat{K}$ according to formula (5)
- Determining the least number m $(m \in \mathbb{N})$ according to the following equation

$$\frac{\sum_{i=1}^{m}\widehat{K}_{ii}}{\sum_{i=1}^{l}\widehat{K}_{ii}} \geqslant 1-\frac{\alpha}{2}$$

- $K_m \triangleq K_{1:m\times 1:m}$
- $[V, \bigwedge] = eig(K_m)$
- Determining the least number k $(k \in \mathbb{N})$ according to the following equation

$$\frac{\sum_{i=1}^{k}\beta_i}{\sum_{i=1}^{N}\beta_i} \geqslant 1-\frac{\alpha}{2}$$

where $\beta_1, \beta_2, \ldots, \beta_m$ are the eigenvalues of matrix K_m in decreasing order.
- $\alpha^j = \beta_j^{-1/2} v_j,\ j = 1, \ldots, k$
- $\tilde{x} = \left(\sum_{i=1}^{l}\alpha_i^j k(x_i, x)\right)_{j=1}^{k}$

Output:
- Transformed data $\widetilde{S} = \{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_l\}$

4. Experiments and discussions

To verify the proposed APCA and AKPCA, four tests are provided. The first and second tests are used to compare the number of used data and CUP time between the APCA, AKPCA and the standard PCA, KPCA on one artificial and six UCI benchmark datasets based on the same threshold α. To further testify the performance and learning effectiveness of the APCA and AKPCA, in the third and forth tests, some comparison experiments are also implemented on UCI data sets for classification problems. All experiments are carried out by using Matlab 7.0, and on 2.0 GHz Intel(R) Pentium(R) D CPU machine with 1KMB RAM.

Test 1: In this test, we apply the standard PCA and the proposed APCA to obtain principal components, respectively. Based on the same threshold, the number of used data and the CPU time are verified on one synthetic dataset and six datasets from the UCI benchmark repository which is listed in Table 1. The synthetic data are generated from a normal distribution with mean zero and standard deviation one. The comparison results are listed in Tables 2–4. In these tables, the symbol rate1 represents the ratio of the number of used data in APCA to that of in PCA, and rate2 represents the ratio of the CPU time of PCA to that of APCA, α represents the threshold stated above.

From Tables 2–4, it can be seen that the more the number of used data, the more the CPU time by PCA. The CPU time of PCA is 20 times longer than that of APCA in maximum. The used data points by APCA is more smaller than that by the standard PCA. Particularly, we only use 25–50% thyroid data points with α from 0.3 to 0.1. The first two columns of thyroid data points are shown in Fig. 1. From Fig. 1, it can be observed that a small circle around the mean center contains a lot of thyroid data which means that the more number of data points around the mean center, the more effective and practical the proposed approach.

Test 2: Similar to Test 1, we apply the standard KPCA and the proposed AKPCA to obtain principal components, respectively. Based on the same threshold α, the number of used data and the CPU time are verified on six data sets from the UCI benchmark repository listed in Table 5. In the following tables, the symbol rate1 represents the ratio of the number of used data in AKPCA to that in KPCA and rate2 represents the ratio of the CPU time in KPCA to that in AKPCA. The comparison results are listed in Tables 6–8.

From Tables 6–8, we can see that similar to Test 1, the more the number of used data, the more the CPU time by the KPCA. The mean CPU time of the KPCA is about two times longer than that of the AKPCA. Therefore, the proposed AKPCA is effective and practical, and it can save lots of CPU time.

Table 1
One synthetic and six benchmark datasets from UCI benchmark repository used in Test 1.

Dataset	No. of attributes	No. of data
Synthetic data	50	10,000
Ringnorm	20	7000
Twonorm	20	7000
Waveform	21	4600
Image	18	1300
German	20	700
Thyroid	5	140

Table 2
The comparison results between the standard PCA and APCA with $\alpha = 0.3$.

Dataset		No. of used data	rate1(%)	CPU time	rate2
Synthetic data	PCA	10000		0.0559	23.6
	APCA	7966	79.7	0.0024	
Ringnorm	PCA	7000		0.0056	17.6
	APCA	4153	59.3	3.1926e−4	
Twonorm	PCA	7000		0.0048	15.5
	APCA	5330	76.1	3.1213e−4	
Waveform	PCA	4600		0.0035	11.8
	APCA	3422	74.4	3.009e−4	
Image	PCA	1300		9.9283e−4	4.6
	APCA	745	57.3	2.1783e−4	
German	PCA	700		7.9604e−4	2.6
	APCA	501	71.6	3.9449e−5	
Thyroid	PCA	140		6.8324e−5	1.3
	APCA	34	24.3	5.4405e−5	

Table 3
The comparison results between the standard PCA and APCA with $\alpha = 0.2$.

Dataset		No. of used data	rate1(%)	CPU time	rate2
Synthetic data	PCA	10,000		0.0299	12.4
	APCA	8586	85.9	0.0024	
Ringnorm	PCA	7000		0.0049	15.8
	APCA	4886	69.8	3.081e−4	
Twonorm	PCA	7000		0.0052	17.4
	APCA	5806	82.9	3.0043e−4	
Waveform	PCA	4600		0.0038	12.5
	APCA	3755	81.6	3.002e−4	
Image	PCA	1300		9.05e−4	4.6
	APCA	872	67.1	1.9855e−4	
German	PCA	700		7.5438e−4	2.3
	APCA	557	79.6	3.2539e−4	
Thyroid	PCA	140		6.7172e−5	1.1
	APCA	49	35	5.9112e−5	

Table 4
The comparison results between the standard PCA and APCA with $\alpha = 0.1$.

Dataset		No. of used data	rate1(%)	CPU time	rate2
Synthetic data	PCA	10,000		0.0299	12.3
	APCA	9245	92.5	0.0024	
Ringnorm	PCA	7000		0.0050	16.8
	APCA	5772	82.5	2.9603e−4	
Twonorm	PCA	7000		0.005	16.7
	APCA	6335	90.5	2.975e−4	
Waveform	PCA	4600		0.0040	13.6
	APCA	4126	89.7	2.9149e−4	
Image	PCA	1300		0.001	5.0
	APCA	1028	79.1	2.0075e−4	
German	PCA	700		7.5094e−4	2.5
	APCA	619	88.4	3.0656e−5	
Thyroid	PCA	140		6.7813e−5	1.3
	APCA	76	54.3	5.3954e−5	

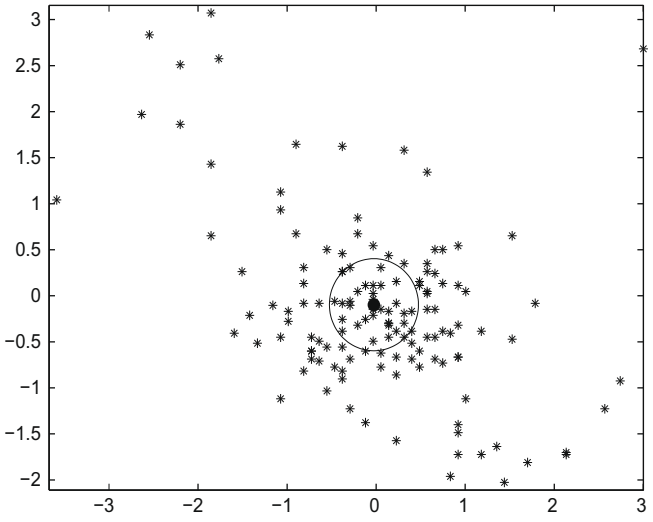

Fig. 1. The first two columns of thyroid dataset.

Test 3: In this test, we apply the standard KPCA and the proposed AKPCA to classification problems on low dimension datasets. The accuracy of classification is verified on six data sets from the

Table 5
The benchmark data sets from UCI benchmark repository.

Dataset	No. of attributes	No. of training data	No. of test data
Banana	2	400	4900
Breast_cancer	9	200	77
Diabetics	8	468	300
Thyroid	5	140	75
German	20	700	300
Flare_solar	9	666	400

Table 6
The comparison results between the standard KPCA and AKPCA with $\alpha = 0.3$.

Dataset		No. of used data	rate1(%)	CPU time	rate2
Banana	KPCA	400		1.5631	1.8
	AKPCA	335	83.8	0.8834	
Breast_cancer	KPCA	200		0.1141	1.5
	AKPCA	170	85	0.0782	
Diabetics	KPCA	468		1.9233	1.7
	AKPCA	398	85	1.1601	
German	KPCA	700		4.5532	1.7
	AKPCA	595	85	2.6242	
Flare_solar	KPCA	666		3.6529	2.3
	AKPCA	526	79	1.6049	
Thyroid	KPCA	140		0.072	2.1
	AKPCA	114	81.4	0.0342	

Table 7
The comparison results between the standard KPCA and AKPCA with $\alpha = 0.2$.

Dataset		No. of used data	rate1(%)	CPU time	rate2
Banana	KPCA	400		1.5086	1.5
	AKPCA	356	89	1.0092	
Breast_cancer	KPCA	200		0.1139	1.4
	AKPCA	180	90	0.0809	
Diabetics	KPCA	468		1.9228	1.4
	AKPCA	421	90	1.4271	
German	KPCA	700		4.5508	1.4
	AKPCA	630	90	3.2757	
Flare_solar	KPCA	666		3.6527	1.8
	AKPCA	573	86	2.0876	
Thyroid	KPCA	140		0.072	1.6
	AKPCA	122	87.1	0.0439	

Table 8
The comparison results between the standard KPCA and AKPCA with $\alpha = 0.1$.

Dataset		No. of used data	rate1(%)	CPU time	rate2
Banana	KPCA	400		1.4896	1.3
	AKPCA	378	94.5	1.194	
Breast_cancer	KPCA	200		0.1138	1.2
	AKPCA	193	95	0.0988	
Diabetics	KPCA	468		1.9216	1.2
	AKPCA	445	95.1	1.64	
German	KPCA	700		4.5486	1.2
	AKPCA	665	95	3.6978	
Flare_solar	KPCA	666		3.6178	1.3
	AKPCA	620	93.1	2.7886	
Thyroid	KPCA	140		0.1315	2.6
	AKPCA	131	93.6	0.0511	

Table 9
The comparison results on mean accuracy between the standard KPCA and AKPCA.

Dataset		$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$
Banana	KPCA	0.8594	0.8583	0.8602
	AKPCA	0.8694	0.8759	0.8775
Breast_cancer	KPCA	0.6883	0.6883	0.6753
	AKPCA	0.3117	0.6710	0.6796
Diabetics	KPCA	0.67	0.67	0.6711
	AKPCA	0.49	0.6167	0.5956
German	KPCA	0.7211	0.28	0.5755
	AKPCA	0.7233	0.7233	0.7233
Flare_solar	KPCA	0.6475	0.6583	0.6617
	AKPCA	0.6550	0.6525	0.6592
Thyroid	KPCA	0.7867	0.7467	0.9733
	AKPCA	0.92	0.9467	0.9467

UCI benchmark repository listed in Table 5. Gaussian kernel $k(x,y) = e^{\frac{-\|x-y\|^2}{2\sigma^2}}$ is used with $\sigma = 0.5$. For the sake of simplicity, we select parameter $C = 10, 100, 1000$, respectively. In the following tables, the accuracy represents mean classification result. The comparison results are listed in Table 9.

From Table 9, it can be seen that with different threshold α, the AKPCA behaves very well both for classification performance and training time. When the threshold $\alpha = 0.1$, the experiment results on four datasets (banana, German, flare-solar and thyroid) by the AKPCA exceed that by the classical KPCA . Only on two datasets (breast-cancer and diabetics), the performance of the AKPCA is inferior to that of the KPCA. When the threshold $\alpha = 0.2$, comparing with the classical KPCA, the numbers of datasets with superior and slight inferior performance by the AKPCA are respective three (banana, German and thyroid) and three (breast-cancer, diabetics and flare-solar). When the threshold $\alpha = 0.3$, comparing with the classical KPCA, the numbers of datasets with superior and slight inferior performance by the AKPCA are the same three, (banana, breast-cancer and German) and three (diabetics flare-solar and thyroid).

Test 4: Similar to Test 3, we apply the standard PCA and the proposed APCA to classification problems. The performance of the APCA is verified on six datasets from the UCI benchmark repository listed in Table 10. The comparison results are listed in Table 11.

From Table 11, it can be seen that with different threshold α, the APCA also behaves very well for classification performance and training time. When the threshold $\alpha = 0.1$, the experiment results on one dataset (image) by the APCA exceed that by the classical PCA, and on four datasets (thyroid, German, waveform and twonorm), the performance of the APCA is equal to that of the PCA. Only on ringnorm, the performance of the APCA is nearly equal to that of the PCA. When the threshold $\alpha = 0.2$, comparing with the classical PCA, the numbers of datasets with superior and slightly inferior performance by the APCA are respective two (thyroid and image) and four (German, waveform, twonorm and ringnorm). When the threshold $\alpha = 0.3$, comparing with the classical PCA, the numbers of datasets with superior, equivalent and slight inferior performance by the APCA are respective one

Table 10
The benchmark data sets from UCI benchmark repository used in Test 4.

Dataset	No. of attributes	No. of training data	No. of test data
Thyroid	5	140	75
German	20	700	300
Image	18	1300	1010
Waveform	21	400	4600
Twonorm	20	400	7000
Ringnorm	20	400	7000

Table 11
The comparison results on mean accuracy between the standard PCA and APCA.

Dataset		$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$
Thyroid	PCA	0.9822	0.96	0.96
	APCA	0.9822	0.9822	0.96
German	PCA	0.72	0.7233	0.7233
	APCA	0.72	0.72	0.7233
Image	PCA	0.9519	0.9244	0.8977
	APCA	0.9532	0.9267	0.9231
Waveform	PCA	0.6707	0.6724	0.7159
	APCA	0.6707	0.6707	0.6709
Twonorm	PCA	0.4994	0.4996	0.5030
	APCA	0.4994	0.4994	0.4994
Ringnorm	PCA	0.4944	0.4983	0.5431
	APCA	0.4940	0.4947	0.4954

(image), two (thyroid and German) and three (waveform, twonorm and ringnorm).

All the experiment results support that the proposed APCA and AKPCA could compete with or outperform the standard PCA and KPCA in generalization performance but with much less memory and time consuming.

5. Conclusion

The contribution of this paper is that we learn linear PCA with partial data points, and extend this method to KPCA. Moreover, based on the given threshold, we could determine which data points can be used and so the store memory and time consumption can be saved greatly.

To validate the performance of the proposed method, four tests are carried out. The results of Test 1 and Test 2 show that the standard PCA and KPCA need more data and CPU time than the proposed APCA and AKPCA with the same threshold. Test 3 and Test 4 illustrate that the proposed APCA and AKPCA can compete with or outperform the standard PCA and KPCA in generalization ability, but they only need much less memory and time consuming. Therefore, the proposed methods are very efficient and practical.

Acknowledgements

The work described in this paper was partially supported by the National Natural Science Foundation of China (Nos. 10671153, 60673095, 60975035), Hi-Tech R&D (863) Program (No. 2007 AA01Z165), Key Project of Science Technology Researches of Ministry of Education, Chinese (No. 208021), Program for New Century Excellent Talents in University (NCET-07-0525), Program for the Top Young Academic Leaders of Higher Learning Institutions (TYAL), the TianYuan Special foundation of the National Natural Science Foundation of China (No. 10926152), Key Project of Natural Science Foundation of Shanxi Province (No. 2009011017-2) and Project for Returned Overseas (No. 2008-14) of Shanxi Province.

References

Cadima, J., & Jolliffe, I. (1995). Loadings and correlations in the interpretation of principal components. *Journal of Applied Statistics, 22*, 203–214.

Croux, C., & Haesbroeck, G. (2000). Principal component analysis based on robust estimators of the covariance or correlation matrix: Influence functions and efficiencies. *Biometrika, 87*, 603–618.

Higuchi, I., & Eguchi, S. (2004). Robust principal component analysis with adaptive selection for tuning parameters. *Journal of Machine Learning Research, 5*, 453–471.

Jolliffe, I. T. (1986). *Principal component analysis*. New York: Springer.

Jolliffe, I. T. (1995). Rotation of principal components: Choice of normalization constraints. *Journal of Applied Statistics, 22*, 29–35.

Jolliffe, I. T., & Uddin, M. (2003). A modified principal component technique based on the lasso. *Journal of Computational and Graphical Statistics, 12*, 531–547.

McCabe, G. (1984). Principal variables. *Technometrics, 26*, 137–144.

Misra, J., Schmitt, W., Hwang, D., Hsiao, L., Gullans, S., Stephanopoulos, G., et al. (2002). Interactive exploration of microarray gene expression patterns in a reduced dimensional space. *Genome Research, 12*, 1112–1120.

Schökopf, B., Smolar, A. J., & Muller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation, 10*, 1299–1319.

Shawe-Taylor, J., & Cristianini, N. (2005). *Kernel methods for pattern analysis*. Beijing: China Machine press.

Suykens, J. A. K., Van Gestel, T., Vandewalle, J., & De Moor, B. (2003). A support vector machine formulation to PCA analysis and its kernel version. *IEEE Transactions on Neural Networks, 14*(2), 447–450.

Tao, Q., Wu, G., & Wang, J. (2007). Learning linear PCA with convex semi-definite programming. *Pattern Recognition, 40*(10), 2633–2640.

Vines, S. (2000). Simple principal components. *Applied Statistics, 49*, 441–451.