Pattern Recognition 45 (2012) 1426–1435

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

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

A new discriminant subspace analysis approach for multi-class problems

Wenming Zheng [a,*], Zhouchen Lin [b]

[a] The Key Laboratory of Child Development and Learning Science (Ministry of Education), Research Center for Learning Science, Southeast University, Nanjing 210096, China
[b] Microsoft Research Asia, Beijing 100080, China

ARTICLE INFO

Article history:
Received 8 April 2010
Received in revised form
19 September 2011
Accepted 22 October 2011
Available online 4 November 2011

Keywords:
Fukunaga–Koontz Transform
Common principal component analysis
Feature extraction

ABSTRACT

Fukunaga–Koontz Transform (FKT) is a famous feature extraction method in statistical pattern recognition, which aims to find a set of vectors that have the best representative power for one class while the poorest representative power for the other class. Li and Savvides [1] propose a one-against-all strategy to deal with multi-class problems, in which the two-class FKT method can be directly applied to find the presentative vectors of each class. Motivated by the FKT method, in this paper we propose a new discriminant subspace analysis (DSA) method for the multi-class feature extraction problems. To solve DSA, we propose an iterative algorithm for the joint diagonalization (JD) problem. Finally, we generalize the linear DSA method to handle nonlinear feature extraction problems via the kernel trick. To demonstrate the effectiveness of the proposed method for pattern recognition problems, we conduct extensive experiments on real data sets and show that the proposed method outperforms most commonly used feature extraction methods.

1. Introduction

Fukunaga–Koontz Transform (FKT) is a famous feature extraction method in statistical pattern recognition [2,3]. It was originally proposed by Fukunaga and Koontz for two-class feature extraction problems. The basic idea of FKT is to find a set of representative vectors that simultaneously represent two classes, in which the vectors that best represent one class will be the least representative ones for the other class. FKT has been widely used in many applications during the past thirty years, including image classification [4], face detection [5] and face recognition [6,7]. To handle the multi-class feature extraction problem, Li and Savvides [1] propose to use a one-against-all strategy such that the two-class FKT method can be directly applied to find the presentative vectors of each class. More specifically, they choose one class as an independent class and use all the remaining classes as a new class, and then apply the FKT method to find the most representative vectors for the chosen class. This procedure is repeated until each class has its own representative vectors. However, it should be noted that this approach works in a relative manner rather than an absolute manner, i.e., the eigenvectors representing each class are solved independently rather than in a unified manner [1]. Hence, the best representative vectors for one class may not be the poor ones for other classes.

Motivated by the FKT method, in this paper we propose a new discriminant subspace analysis (DSA) to deal with the multi-class feature extraction problems. In this method, we firstly borrow the FKT idea of whitening the summation of all the class covariance matrices, and then find an orthogonal matrix that best simultaneously diagonalize all the transformed class covariance matrices in the whitening space. Considering that there may not exist an orthogonal matrix that can exactly and simultaneously diagonalize more than two-class covariance matrices [2], we can only resort to the joint diagonalization (JD) technique of multiple matrices [8,9] to achieve this goal. To this end, in this paper we propose an iterative algorithm to solve the JD problem using the conjugate gradient method on the Stiefel manifold (the set of all orthogonal matrices) [10]. Compared with the original JD algorithm proposed by Flury [8] that uses the maximal likelihood method, the major advantage of our algorithm is that it can remove the non-singularity constraint on the class covariance matrices and therefore can still be applicable when the number of samples of each class is relatively small. Moreover, to obtain a discriminant subspace for each class, we adopt a method similar to the FKT of choosing the vectors from the columns of the transformation matrix that best represent one class while have less representation power for other classes to span the discriminant subspace for that class.

Our DSA can be viewed as an extension of the common principal component analysis (CPCA) method [8] for discrimination problems. The CPCA method, originally proposed by Flury [8], aims to find a common orthogonal matrix for multiple class covariance matrices. However, since the data samples may not

[*] Corresponding author. Tel.: +86 25 83795664 1019.
E-mail addresses: wenming_zheng@seu.edu.cn (W. Zheng), zhoulin@microsoft.com (Z. Lin).

0031-3203/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2011.10.021

share the same metric, we cannot guarantee that the columns of the orthogonal matrix of CPCA with the best representation power for one class will be the poor representation vectors of other classes. In contrast to CPCA, our DSA can overcome these drawbacks of CPCA, by whitening the data samples such that they share the same metric. Moreover, to make the DSA method able to capture the nonlinear structure of the data samples, we also generalize the linear DSA method by utilizing the kernel trick, which has been successfully used in the nonlinear extensions of PCA [11] and linear discriminant analysis (LDA) [12]. We call the nonlinear DSA method as the kernel DSA or simply the KDSA method for short.

This paper is organized as follows. In Section 2, we briefly review the FKT method and the CPCA method. In Section 3, we present our DSA method. The KDSA method is introduced in Section 4. Then the experiments are presented in Section 5. Finally, we conclude our paper in Section 6.

2. Brief review of FKT and CPCA

2.1. Fukunaga–Koontz Transform (FKT)

Let $\mathbf{X}_1$ and $\mathbf{X}_2$ be two data matrices, where each column is a d-dimensional vector. Then the autocorrelation matrices of $\mathbf{X}_1$ and $\mathbf{X}_2$ can be expressed as $\mathbf{R}_1 = (1/N_1)\mathbf{X}_1\mathbf{X}_1^T$ and $\mathbf{R}_2 = (1/N_2)\mathbf{X}_2\mathbf{X}_2^T$, respectively, where N_1 and N_2 represent the number of the columns of $\mathbf{X}_1$ and $\mathbf{X}_2$, respectively. Let $\mathbf{R} = \mathbf{R}_1 + \mathbf{R}_2$. Performing the singular value decomposition (SVD) of $\mathbf{R}$, we obtain

$$\mathbf{R} = (\mathbf{V}\ \ \mathbf{V}^{\perp})\begin{pmatrix} \mathbf{\Lambda} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}\begin{pmatrix} \mathbf{V}^T \\ \mathbf{V}^{\perp T} \end{pmatrix}, \tag{1}$$

where $\mathbf{\Lambda}$ is a diagonal matrix whose diagonal entries are positive and $\mathbf{0}$ denotes zero matrices. Let $\mathbf{P} = \mathbf{V}\mathbf{\Lambda}^{-1/2}$. Then we obtain

$$\mathbf{P}^T\mathbf{R}\mathbf{P} = \mathbf{P}^T(\mathbf{R}_1 + \mathbf{R}_2)\mathbf{P} = \hat{\mathbf{R}}_1 + \hat{\mathbf{R}}_2 = \mathbf{I},$$

where $\hat{\mathbf{R}}_1 = \mathbf{P}^T\mathbf{R}_1\mathbf{P}$, $\hat{\mathbf{R}}_2 = \mathbf{P}^T\mathbf{R}_2\mathbf{P}$ and $\mathbf{I}$ is the identity matrix. Let

$$\hat{\mathbf{R}}_1\mathbf{x} = \lambda_1\mathbf{x} \tag{2}$$

be the eigen-analysis of $\hat{\mathbf{R}}_1$. Then we have

$$\hat{\mathbf{R}}_2\mathbf{x} = (\mathbf{I} - \hat{\mathbf{R}}_1)\mathbf{x} = (1-\lambda_1)\mathbf{x}. \tag{3}$$

Eqs. (2) and (3) show that $\hat{\mathbf{R}}_1$ and $\hat{\mathbf{R}}_2$ share the same eigenvector $\mathbf{x}$, but the corresponding eigenvalues are different (the eigenvalues of $\hat{\mathbf{R}}_2$ is $\lambda_2 = 1-\lambda_1$) and they are bounded between 0 and 1. Therefore, the eigenvectors which best represent class 1 (i.e., $\lambda_1 \approx 1$) are the poorest ones for representing class 2 (i.e., $\lambda_2 = 1-\lambda_1 \approx 0$). Suppose the SVD of $\hat{\mathbf{R}}_1$ is $\hat{\mathbf{R}}_1 = \mathbf{Q}_1\mathbf{\Lambda}_1\mathbf{Q}_1^T$ and let $\hat{\mathbf{P}} = \mathbf{P}\mathbf{Q}_1$, then we have that $\hat{\mathbf{P}}^T\mathbf{R}\hat{\mathbf{P}} = \mathbf{I}$, $\hat{\mathbf{P}}^T\mathbf{R}_1\hat{\mathbf{P}} = \mathbf{\Lambda}_1$ and $\hat{\mathbf{P}}^T\mathbf{R}_2\hat{\mathbf{P}} = \mathbf{I} - \mathbf{\Lambda}_1$. So $\hat{\mathbf{P}}$ simultaneously diagonalizes $\mathbf{R}_1$ and $\mathbf{R}_2$.

It is notable that the above two-class FKT solution method cannot be simply extended to the general multi-class problem. This is because there may not exists a matrix that can exactly diagonalize more than two autocorrelation matrices simultaneously. For multi-class problems, Li and Savvides [1] use a one-against-all strategy to construct a sequence of two-class FKT.

2.2. Common principal component analysis (CPCA)

Suppose that we have c data matrices $\mathbf{X}_i$ $(i = 1, 2, \ldots, c)$ from d-dimensional data space. Let $\mathbf{u}_i$ denote the mean of the i-th data matrix $\mathbf{X}_i$ and N_i denote the number of the columns of $\mathbf{X}_i$ (i.e., the number of samples in the i-th class). Then the covariance matrix of the i-th data matrix can be expressed as

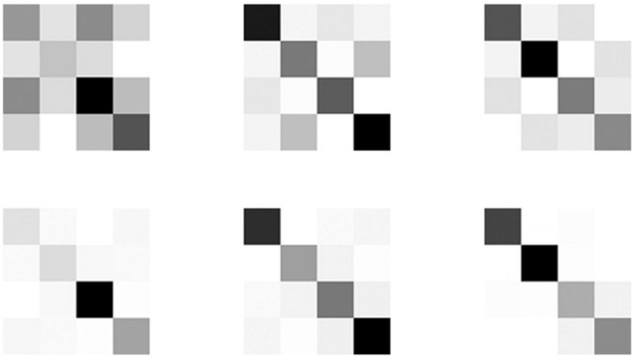

Fig. 1. An example of JD of the class covariance matrices. The first row shows the three 4 × 4 class covariance matrices. The second row shows the results of simultaneous diagonalization after performing the JD procedure. The grayscale corresponds to the magnitude of the matrix entries, where the darker pixels indicate larger values.

$$\mathbf{\Sigma}_i = \frac{1}{N_i}\mathbf{X}_i\mathbf{X}_i^T - \mathbf{u}_i\mathbf{u}_i^T, \quad i = 1, 2, \ldots, c. \tag{4}$$

The goal of CPCA is to find an orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{d\times d}$ that simultaneously diagonalizes the c class covariance matrices $\mathbf{\Sigma}_i$, i.e.,

$$\mathbf{Q}^T\mathbf{\Sigma}_i\mathbf{Q} = \mathbf{\Lambda}_i$$

$$\text{s.t.} \quad \mathbf{Q}^T\mathbf{Q} = \mathbf{I}, \tag{5}$$

where ideally $\mathbf{\Lambda}_i$ should be diagonal.

The optimal solution to (5) can be found by performing the JD of all the class covariance matrices $\mathbf{\Sigma}_i$. The concept of JD is illustrated in Fig. 1, where usually $\mathbf{\Lambda}_i$ cannot be exactly diagonal when $c > 2$. One of the most well-known JD algorithms to solve the optimization problem in (5) was proposed by Flury [8], which is based on maximum likelihood estimation. However, this algorithm may not be applicable when the class covariance matrices $\mathbf{\Sigma}_i$ are singular because the inverses of class covariance matrices are involved. Moreover, it should be noted that the principal components associated with different class covariance matrices do not share the same metric and hence cannot guarantee that the columns of $\mathbf{Q}$ that best represent one class will be the poor ones for other classes.

3. Discriminant subspace analysis (DSA)

Suppose that $\mathbf{P}$ is the whitening matrix of the summation of $\mathbf{\Sigma}_i$ $(i = 1, 2, \ldots, c)$, i.e.,

$$\sum_{i=1}^{c}\mathbf{P}^T\mathbf{\Sigma}_i\mathbf{P} = \sum_{i=1}^{c}\hat{\mathbf{\Sigma}}_i = \mathbf{I}, \tag{6}$$

where $\hat{\mathbf{\Sigma}}_i = \mathbf{P}^T\mathbf{\Sigma}_i\mathbf{P}$ $(i = 1, 2, \ldots, c)$.

Similar to the CPCA method, we perform JD on the c *whitening* class covariance matrices $\hat{\mathbf{\Sigma}}_i$ $(i = 1, 2, \ldots, c)$, i.e., seeking an orthogonal matrix $\mathbf{Q}$ satisfying the following constraints:

$$\mathbf{Q}^T\hat{\mathbf{\Sigma}}_i\mathbf{Q} = \mathbf{\Lambda}_i$$

$$\text{s.t.} \quad \mathbf{Q}^T\mathbf{Q} = \mathbf{I}, \tag{7}$$

where ideally $\mathbf{\Lambda}_i$ should be diagonal. Compare (7) with (5), we can see that $\mathbf{Q}$ is actually the orthogonal transform matrix of CPCA defined on $\hat{\mathbf{\Sigma}}_i$, while CPCA performs on the original covariance matrix $\mathbf{\Sigma}_i$ $(i = 1, 2, \ldots, c)$. Consequently, our DSA can be seen as a generalization of CPCA.

To solve the optimal solution $\mathbf{Q}$, we formulate the optimization problem (7) into the following form:

$$\mathbf{Q}^* = \arg \min_{\mathbf{Q}^T\mathbf{Q}=\mathbf{I}} g(\mathbf{Q}), \tag{8}$$

where the objective function $g(\mathbf{Q})$ is defined as

$$g(\mathbf{Q}) = \frac{1}{4}\sum_{i=1}^{c} \|\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} - \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\|_F^2, \tag{9}$$

in which each term in the right-hand side of (9) measures how close $\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}$ is to be diagonal and $\mathrm{diag}(\mathbf{A})$ represents a diagonal matrix whose diagonal entries are the same as those of $\mathbf{A}$.

To solve the optimization problem in Eq. (8), we use the conjugate gradient method on the Stiefel manifold [10]. The pseudo-code is presented in Algorithm 1. To run Algorithm 1, we have to solve two subproblems: first, to compute the derivative of $g(\mathbf{Q})$ with respect to $\mathbf{Q}$; second, to minimize $g(\mathbf{Q}_k(t))$ therein over t, where $\mathbf{Q}_k(t)$ has the form of $\mathbf{Q}_k(t) = \mathbf{Q}_k e^{t\mathbf{A}_k}$ and $\mathbf{A}_k$ is a skew-symmetric matrix.

Algorithm 1. The conjugate gradient method for minimizing $g(\mathbf{Q})$ on the Stiefel manifold (adapted from [10]).

```
Input:
  • Covariance matrices $\hat{\boldsymbol{\Sigma}}_1, \hat{\boldsymbol{\Sigma}}_2, \ldots, \hat{\boldsymbol{\Sigma}}_c$ and the threshold $\varepsilon > 0$.
Initialization:
  1. Choose an orthogonal matrix $\mathbf{Q}_0$;
  2. Compute $\mathbf{G}_0 = \mathbf{Z}_0 - \mathbf{Q}_0\mathbf{Z}_0^T\mathbf{Q}_0$, where $\mathbf{Z}_0 = \frac{dg}{d\mathbf{Q}}|_{\mathbf{Q}_0}$;
  3. Set $\mathbf{H}_0 = -\mathbf{G}_0$, $\mathbf{A}_0 = \mathbf{Q}_0^T\mathbf{H}_0$ and $k \leftarrow 0$;
Do while $\|\mathbf{A}_k\|_F > \varepsilon$
  1. Minimize $g(\mathbf{Q}_k(t))$ over $t$, where $\mathbf{Q}_k(t) = \mathbf{Q}_k\mathbf{M}(t)$ and
     $\mathbf{M}(t) = e^{t\mathbf{A}_k}$;
  2. Set $t_k \leftarrow t_{min}$ and $\mathbf{Q}_{k+1} \leftarrow \mathbf{Q}_k(t_k)$, where
     $t_{min} = \arg\min_t g(\mathbf{Q}_k(t))$;
  3. Compute $\mathbf{G}_{k+1} = \mathbf{Z}_{k+1} - \mathbf{Q}_{k+1}\mathbf{Z}_{k+1}^T\mathbf{Q}_{k+1}$, where
     $\mathbf{Z}_{k+1} = \frac{dg}{d\mathbf{Q}}|_{\mathbf{Q}_{k+1}}$;
  4. Parallel transport tangent vector $\mathbf{H}_k$ to the point $\mathbf{Q}_{k+1}$:
     $\tau(\mathbf{H}_k) \leftarrow \mathbf{H}_k\mathbf{M}(t_k)$;
  5. Compute the new search direction:
     $\mathbf{H}_{k+1} = -\mathbf{G}_{k+1} + \gamma_k\tau(\mathbf{H}_k)$, where $\gamma_k = \frac{\langle \mathbf{G}_{k+1} - \mathbf{G}_k, \mathbf{G}_{k+1} \rangle}{\langle \mathbf{G}_k, \mathbf{G}_k \rangle}$ and
     $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{Tr}(\mathbf{A}^T\mathbf{B})$;
  6. if $k+1 \equiv 0 \mod d(d-1)/2$, then reset $\mathbf{H}_{k+1} = -\mathbf{G}_{k+1}$.
  7. Set $\mathbf{A}_{k+1} \leftarrow \mathbf{Q}_{k+1}^T\mathbf{H}_{k+1}$;
  8. Set $k \leftarrow k+1$;
Output:
  • Set $\mathbf{Q} \leftarrow \mathbf{Q}_k$ and output $\mathbf{Q}$.
```

For the first subproblem, the derivative of $g(\mathbf{Q})$ can be found to be[1]

$$\frac{dg(\mathbf{Q})}{d\mathbf{Q}} = \frac{1}{2}\left(\sum_{i=1}^{c}\hat{\boldsymbol{\Sigma}}_i^2\right)\mathbf{Q} - \sum_{i=1}^{c}\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}). \tag{10}$$

For the second subproblem, we notice that $g(\mathbf{Q}_k(t))$ is a smooth function of t, hence its minimal point can be found by Newton's iteration method [13] as it must be a zero of $f_k(t) = dg(\mathbf{Q}_k(t))/dt$. To find the zeros of $f_k(t)$ by Newton's iteration method,[2] we have to know the derivative of $f_k(t)$. $f_k(t)$ and $df_k(t)/dt$ can be found to be

$$f_k(t) = \mathrm{Tr}\left(A_k\sum_{i=1}^{c}\mathbf{S}_{i,k}(t)\mathrm{diag}(\mathbf{S}_{i,k}(t))\right) \tag{11}$$

and

$$\frac{df_k(t)}{dt} = \mathrm{Tr}\Bigg(\mathbf{A}_k\sum_{i=1}^{c}[((\mathbf{S}_{i,k}(t)\mathbf{A}_k)^T + \mathbf{S}_{i,k}(t)\mathbf{A}_k)\mathrm{diag}(\mathbf{S}_{i,k}(t)) + \mathbf{S}_{i,k}(t)\mathrm{diag}((\mathbf{S}_{i,k}(t)\mathbf{A}_k)^T + \mathbf{S}_{i,k}(t)\mathbf{A}_k)]\Bigg), \tag{12}$$

respectively, where $\mathbf{S}_{i,k}(t) = \mathbf{Q}_k^T(t)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t)$.

Now denote the optimal solution to (7) by $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_r]$, where r is the dimensionality of $\hat{\boldsymbol{\Sigma}}_i$. Then, by the philosophy of CPCA, the columns of $\mathbf{Q}$ are the common eigenvectors of all $\hat{\boldsymbol{\Sigma}}_i$. So, to find the most representative vectors for each class, we compute $d_{i,j} = \mathbf{q}_j^T\hat{\boldsymbol{\Sigma}}_i q_j$ $(j = 1, 2, \ldots, r)$ to measure the representative power of vector $\mathbf{q}_j$ for class i. In this case, the vectors $\mathbf{q}_{i_1}, \mathbf{q}_{i_2}, \ldots, \mathbf{q}_{i_k}$ that correspond to the top k largest values of $d_{i,j}$ $(j = 1, 2, \ldots, r)$ are the most representative vectors for class i. Let $\mathbf{Q}_i = [\mathbf{q}_{i_1}, \mathbf{q}_{i_2}, \ldots, \mathbf{q}_{i_k}]$ $(i = 1, 2, \ldots, c)$ and $\mathbf{x}$ be a test sample. Then, if $\mathbf{x}$ is from the i-th class data set, then the reconstruction error of $\mathbf{P}^T(\mathbf{x} - \mathbf{u}_i)$ by $\mathbf{Q}_i$ should be the least among those by $\mathbf{Q}_j$ $(j \neq i)$. Consequently, we can assign the class label, denoted by c^*, of $\mathbf{x}$ according to the following criterion:

$$c^*(\mathbf{x}) = \arg\min_i\{\|\mathbf{z}_i\|\}, \tag{13}$$

where $\mathbf{z}_i$ is given by

$$\mathbf{z}_i = (\mathbf{I} - \mathbf{Q}_i\mathbf{Q}_i^T)\mathbf{P}^T(\mathbf{x} - \mathbf{u}_i). \tag{14}$$

We summarize our DSA algorithm in Algorithm 2.

Algorithm 2. Discriminant subspace analysis.

```
Input: Data matrices $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_c]$ and a test sample $\mathbf{x}$.
  1. Compute the covariance matrix of $\mathbf{X}_i$: $\boldsymbol{\Sigma}_i = \frac{1}{N_i}\mathbf{X}_i\mathbf{X}_i^T - \mathbf{u}_i\mathbf{u}_i^T$
     $(i = 1, 2, \ldots, c)$, where $N_i$ is the number of columns of $\mathbf{X}_i$;
  2. Compute $\boldsymbol{\Sigma} = \sum_{i=1}^{c}\boldsymbol{\Sigma}_i$;
  3. Perform the SVD of $\boldsymbol{\Sigma}$: $\boldsymbol{\Sigma} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^T$, and let $\mathbf{P} = \mathbf{V}\boldsymbol{\Lambda}^{-1/2}$;
  4. Set $\hat{\boldsymbol{\Sigma}}_i = \mathbf{P}^T\boldsymbol{\Sigma}_i\mathbf{P}$;
  5. Solve the orthogonal matrix $\mathbf{Q}$ that best simultaneously
     diagonalizes $\hat{\boldsymbol{\Sigma}}_1, \ldots, \hat{\boldsymbol{\Sigma}}_c$ using Algorithm 1;
  6. Find the most discriminant vectors $\mathbf{Q}_i$ for each class;
  7. Find the class identifier $c^*(\mathbf{x})$ for $\mathbf{x}$ by Eq. (13).
Output:
  • $c^*(\mathbf{x})$.
```

4. Kernel discriminant subspace analysis (KDSA)

We now generalize the linear DSA to the nonlinear case. Let Φ be a nonlinear mapping that maps the columns of the data matrices $\mathbf{X}_i$ $(i = 1, 2, \ldots, c)$ from the d-dimensional data space $\mathcal{R}^d$ to a high-dimensional feature space $\mathcal{F}$, i.e.,

$$\Phi : \mathcal{R}^d \to \mathcal{F}. \tag{15}$$

Let $\mathbf{X}_i^\Phi$ denote the corresponding data matrices $\mathbf{X}_i$ in the feature space $\mathcal{F}$, then the covariance matrix of the i-th class data set in $\mathcal{F}$ can be expressed as

$$\boldsymbol{\Sigma}_i^\Phi = \frac{1}{N_i}\mathbf{X}_i^\Phi(\mathbf{X}_i^\Phi)^T - \mathbf{u}_i^\Phi(\mathbf{u}_i^\Phi)^T \quad (i = 1, 2, \ldots, c), \tag{16}$$

where $\mathbf{u}_i^\Phi = (1/N_i)\sum_{j=1}^{N_i}\mathbf{X}_{ij}^\Phi$ denotes the mean vector of the i-th data set and $\mathbf{X}_{ij}^\Phi$ denotes the j-th column of $\mathbf{X}_i^\Phi$. Let

$$\boldsymbol{\Sigma}^\Phi = \sum_{i=1}^{c}\boldsymbol{\Sigma}_i^\Phi = \sum_{i=1}^{c}\left[\frac{1}{N_i}\mathbf{X}_i^\Phi(\mathbf{X}_i^\Phi)^T - \mathbf{u}_i^\Phi(\mathbf{u}_i^\Phi)^T\right] \tag{17}$$

[1] The details of deducing Eqs. (10)–(12) are given in Appendix.

[2] Some trivial tricks, e.g., by checking whether $g(\mathbf{Q}_k(t))$ decreases, should be adopted in order not to find the maximal points of $g(\mathbf{Q}_k(t))$ as they are also the zeros of $f_k(t)$.

and let $\mathbf{P}^{\Phi}$ be the whitening matrix of $\mathbf{\Sigma}^{\Phi}$, such that

$$\mathbf{P}^{\Phi^T}\mathbf{\Sigma}^{\Phi}\mathbf{P}^{\Phi} = \mathbf{P}^{\Phi^T}(\mathbf{\Sigma}_1^{\Phi}+\mathbf{\Sigma}_2^{\Phi}+\cdots+\mathbf{\Sigma}_c^{\Phi})\mathbf{P}^{\Phi} = \sum_{i=1}^{c}\hat{\mathbf{\Sigma}}_i = \mathbf{I}, \tag{18}$$

where

$$\hat{\mathbf{\Sigma}}_i = \mathbf{P}^{\Phi^T}\mathbf{\Sigma}_i^{\Phi}\mathbf{P}^{\Phi} \quad (i=1,2,\ldots,c). \tag{19}$$

In this case, solving the KDSA problem boils down to solving the same optimization problem as (7).

To find the transformation matrix $\mathbf{P}^{\Phi}$, we perform the singular value decomposition (SVD) [2] on $\mathbf{\Sigma}^{\Phi}$. More specifically, let ω_p^{Φ} $(p=1,2,\ldots,m)$ denote the eigenvectors of $\mathbf{\Sigma}^{\Phi}$ corresponding to the nonzero eigenvalue $\lambda_p > 0$, then we have

$$\mathbf{\Sigma}^{\Phi}\omega_p^{\Phi} = \lambda_p\omega_p^{\Phi}. \tag{20}$$

From the literature [11], we know that ω_p^{Φ} can be expressed as a linear combination of the columns of $\mathbf{X}_i^{\Phi}$ $(i=1,2,\ldots,c)$. Let $\mathbf{X}^{\Phi} = [\mathbf{X}_1^{\Phi}\ \mathbf{X}_2^{\Phi}\ \cdots\ \mathbf{X}_c^{\Phi}]$. Then, for each ω_p^{Φ}, there exists α_p such that

$$\omega_p^{\Phi} = \mathbf{X}^{\Phi}\alpha_p. \tag{21}$$

Combining (17), (20) and (21), we obtain that α_p are the eigenvectors of the following eigensystem:

$$\sum_{i=1}^{c}\frac{1}{N_i}\overline{\mathbf{K}}_i\overline{\mathbf{K}}_i^T\alpha_p = \lambda_p\mathbf{K}\alpha_p, \tag{22}$$

where $\mathbf{K} = (\mathbf{K}_{ij})_{c\times c}$ is a $c \times c$ block matrix, $\mathbf{K}_{ij} = (\mathbf{X}_j^{\Phi})^T\mathbf{X}_i^{\Phi}$ whose entries can be computed via the kernel trick (i.e., the product of two vectors, $\mathbf{x}^{\Phi}$ and $\mathbf{y}^{\Phi}$ can be computed via the kernel function $(\mathbf{y}^{\Phi})^T\mathbf{x}^{\Phi} = \mathrm{kernel}(\mathbf{x},\mathbf{y})$), $\overline{\mathbf{K}}_i = \mathbf{K}_i - \mathbf{K}_i\mathbf{N}_i$, $\mathbf{K}_i = [\mathbf{K}_{i1}\ \mathbf{K}_{i2}\ \cdots\ \mathbf{K}_{ic}]^T$, $\mathbf{N}_i$ is an $N_i \times N_i$ matrix with all entries being $1/N_i$, and α_p is divided by $\sqrt{\lambda_p}$ such that $\|\omega_p^{\Phi}\| = 1$.

Let $\mathbf{V}^{\Phi} = [\omega_1^{\Phi}\ \omega_2^{\Phi}\ \cdots\ \omega_m^{\Phi}] = \mathbf{X}^{\Phi}\mathbf{U}$ and $\mathbf{\Lambda} = [\lambda_1\ \lambda_2\ \cdots\ \lambda_m]$, where $\mathbf{U} = [\alpha_1\ \alpha_2\ \cdots\ \alpha_m]$. Then from (20) we obtain the following SVD expression of $\mathbf{\Sigma}^{\Phi}$[3]:

$$\mathbf{\Sigma}^{\Phi} = \mathbf{V}^{\Phi}\mathbf{\Lambda}\mathbf{V}^{\Phi^T}. \tag{23}$$

From (23), we obtain that

$$\mathbf{P}^{\Phi} = \mathbf{V}^{\Phi}\mathbf{\Lambda}^{-1/2} = \mathbf{X}^{\Phi}\mathbf{U}\mathbf{\Lambda}^{-1/2}. \tag{24}$$

According to (16), (19) and (24), we have

$$\begin{aligned}\hat{\mathbf{\Sigma}}_i &= (\mathbf{P}^{\Phi})^T\mathbf{\Sigma}_i^{\Phi}\mathbf{P}^{\Phi}\\ &= \mathbf{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{X}^{\Phi})^T\left(\frac{1}{N_i}\mathbf{X}_i^{\Phi}(\mathbf{X}_i^{\Phi})^T - \mathbf{u}_i^{\Phi}(\mathbf{u}_i^{\Phi})^T\right)\mathbf{X}^{\Phi}\mathbf{U}\mathbf{\Lambda}^{-1/2}\\ &= \frac{1}{N_i}\mathbf{\Lambda}^{-1/2}\mathbf{U}^T\overline{\mathbf{K}}_i\overline{\mathbf{K}}_i^T\mathbf{U}\mathbf{\Lambda}^{-1/2}.\end{aligned} \tag{25}$$

Hence, the projection of the test sample $\mathbf{x}^{\Phi} - \mathbf{u}_i^{\Phi}$ onto the matrix $\mathbf{P}^{\Phi}$ can be expressed as

$$(\mathbf{P}^{\Phi})^T(\mathbf{x}^{\Phi} - \mathbf{u}_i^{\Phi}) = \mathbf{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{X}^{\Phi})^T(\mathbf{x}^{\Phi} - \mathbf{u}_i^{\Phi}) = \mathbf{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{k} - \mathbf{K}_i\mathbf{n}_i), \tag{26}$$

where $\mathbf{n}_i$ is an $N_i \times 1$ vector with all element being $1/N_i$ and $\mathbf{k} = (\mathbf{X}^{\Phi})^T\mathbf{x}^{\Phi}$ can be computed via the kernel trick.

Let $\mathbf{Q}$ be the optimal solution to (19) and let $\mathbf{Q}_i$ be the matrices whose columns are those of $\mathbf{Q}$ with the best representation power for class i. Then the class label of $\mathbf{x}^{\Phi}$ can expressed as $c^*(\mathbf{x}) = \arg\min_i\{\|\mathbf{z}_i\|\}$, where

$$\begin{aligned}\mathbf{z}_i &= (\mathbf{I} - \mathbf{Q}_i\mathbf{Q}_i^T)(\mathbf{P}^{\Phi})^T(\mathbf{x}^{\Phi} - \mathbf{u}_i^{\Phi})\\ &= (\mathbf{I} - \mathbf{Q}_i\mathbf{Q}_i^T)\mathbf{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{k} - \mathbf{K}_i\mathbf{n}_i).\end{aligned} \tag{27}$$

We summarize our kernel discriminant subspace analysis (KDSA) based algorithm in Algorithm 3.

Algorithm 3. Kernel discriminant subspace analysis.

```
Input: Data matrices $\mathbf{X} = [\mathbf{X}_1,\mathbf{X}_2,\ldots,\mathbf{X}_c]$, kernel function
kernel$(\cdot,\cdot)$, and a test sample $\mathbf{x}$.
  1. Compute $\mathbf{K}_{ij} = (\mathbf{X}_j^{\Phi})^T\mathbf{X}_i^{\Phi}$, $\mathbf{N}_i$, $\overline{\mathbf{K}}_i = \mathbf{K}_i - \mathbf{K}_i\mathbf{N}_i$, and the
     coefficient vector $\mathbf{n}_i$;
  2. Solve the eigenvectors $\alpha_i$ of $\mathbf{K} = (\mathbf{K}_{ij})_{c\times c}$ corresponding to
     the nonzero eigenvalues $\lambda_i$, and set $\alpha_i \leftarrow \alpha_i/\sqrt{\lambda_i}$
     $(i=1,2,\ldots,m)$;
  3. Set $\mathbf{U} = [\alpha_1\ \alpha_2\ \cdots\ \alpha_m]$ and $\mathbf{\Lambda} = [\lambda_1\ \lambda_2\ \cdots\ \lambda_m]$;
  4. Compute $\hat{\mathbf{\Sigma}}_i = \frac{1}{N_i}\mathbf{\Lambda}^{-1/2}\mathbf{U}^T\overline{\mathbf{K}}_i\overline{\mathbf{K}}_i^T\mathbf{U}\mathbf{\Lambda}^{-1/2}$;
  5. Solve the orthogonal matrix $\mathbf{Q}$ that best simultaneously
     diagonalizes $\hat{\mathbf{\Sigma}}_1,\ldots,\hat{\mathbf{\Sigma}}_c$ using Algorithm 1;
  6. Find the most discriminant vectors $\mathbf{Q}_i$ for each class;
  7. Find the class identifier $c^*(\mathbf{x})$ for $\mathbf{x}$ by Eq. (13), where $\mathbf{z}_i$ is
     computed according to (27).
Output:
  • $c^*(\mathbf{x})$.
```

5. Experiments

In this section, we test the effectiveness of the proposed JD algorithm as well as the recognition performance of the proposed DSA and KDSA methods on four real data sets, i.e., the IRIS data set [14], Ekman's POFA (Picture of Facial Affect) database [15], the ORL face database [16], and the texture database [17]. The brief description of these data sets are given as follows:

1. The IRIS data set was originally used by Fisher [14] for the study of taxonomic problems. It consists of 150 samples from three classes, where each class contains 50 samples and each sample is a four-dimensional feature vector.
2. The POFA database consists of 110 facial images covering six basic emotions (i.e., happy, angry, sad, surprise, disgust, and fear) plus the neutral emotion. There are 14 subjects in total (six males and eight females).
3. The ORL face database consists of 40 subjects, and each one contains 10 different images taken at different time and with slightly varying lighting. The size of each original face image is 112 × 92 pixels, with a 256-level grayscale.
4. The texture image database used in this experiment comprises the 13 textures from the Brodatz album [17]. Each texture has the size of 512 × 512 pixels, digitized at six different rotation angles (0°, 30°, 60°, 90°, 120° and 150°). Similar to the method in [18], all the images are divided into 16 disjoint 128 × 128 subimages. Hence, we obtain a texture image data set of 1248 samples in total, each of the 13 classes having 96 samples.

To evaluate the recognition performance of the proposed DSA and KDSA methods, we also use the PCA method, the LDA method [19], the LDA/GSVD method [20], the LDA/FKT method [7], the KPCA method [11], the KLDA method [21] and the KFKT method [1], respectively, to conduct the same experiments for comparison. When using the methods of PCA, LDA, LDA/GSVD, LDA/FKT, KPCA, and KLDA in the experiments, we choose the nearest neighbor (NN) rule with the Euclidean distance as the classifier. Moreover, throughout the experiments, we use the monomial kernel function and the Gaussian kernel function, defined as

$$\mathrm{kernel}(\mathbf{x},\mathbf{y}) = (\mathbf{x}^T\mathbf{y})^{\mathrm{ker}} \tag{28}$$

and

$$\mathrm{kernel}(\mathbf{x},\mathbf{y}) = \exp\left\{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{\sigma}\right\}, \tag{29}$$

[3] Here we utilize the orthogonality among the eigenvectors ω_i^{Φ}.

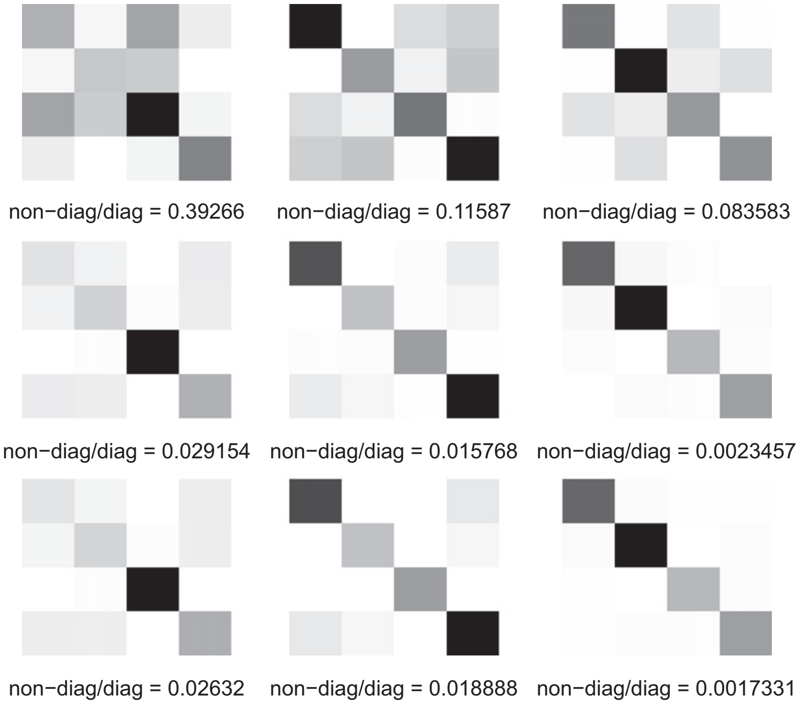

Fig. 2. Comparison of the effectiveness of simultaneously diagonalizing the class covariance matrices of two JD algorithms: the first row shows the three class covariance matrices of the IRIS data samples, the second row shows the corresponding class covariance matrices after performing Ziehe et al.'s JD algorithm, and the third row shows the corresponding class covariance matrices after performing our JD algorithm. Below each covariance matrix is the ratio of the sum of squared non-diagonal entries to that of the diagonal ones.

respectively, to calculate the inner product of any two vectors in the feature space $\mathcal{F}$, where ker and σ denote the degree of monomial kernel and the Gaussian kernel parameter, respectively.

5.1. Experiments on the IRIS data set

In this experiment, we aim to use the IRIS data set to demonstrate the effectiveness of our JD algorithm. We also compare with the JD algorithm proposed by Ziehe et al. [9]. We design our experiment according to the following procedures:

1. Calculate the class covariance matrices of the data samples $\mathbf{x}_i$.
2. Solve the transform matrix $\mathbf{P}$ that whitens the summation of the three class covariance matrices.
3. Transform the data samples with the matrix $\mathbf{P} : \mathbf{y}_i = \mathbf{P}^T\mathbf{x}_i$.
4. Calculate the new class covariance matrices using the transformed data samples, and then perform the JD algorithms on the new class covariance matrices.

Fig. 2 depicts the class covariance matrices and the results after performing Ziehe et al.'s JD algorithm and ours, where the grayscale corresponds to the magnitude of the matrix entries and the darker pixels indicate larger values. The first row of Fig. 2 lists the three class covariance matrices of the data samples $\mathbf{y}_i$, the second one lists the results after performing Ziehe et al.'s JD algorithm, whereas the third one lists the results after performing our JD algorithm. Below each covariance matrix, we present the ratio of the sum of squared non-diagonal entries to that of the diagonal ones, indicating the performance of the JD algorithms. The smaller the ratio value, the better the JD algorithm. Moreover, we also show the numerical results of the class covariance matrices in Tables 1 and 2, where Table 1 shows the numerical results of three class covariance matrices to be used for simultaneous diagonalization, and Table 2 shows the numerical results after performing the JD algorithms of the Ziehe et al. and ours, respectively.

Table 1
Class covariance matrices to be used for simultaneous diagonalization.

Class no.	4 × 4 Class covariance matrices			
1	0.1570	−0.0356	0.1808	0.0490
	−0.0356	0.1176	−0.1084	−0.0210
	0.1808	−0.1084	0.3981	0.0411
	0.0490	−0.0210	0.0411	0.2425
2	0.3820	−0.0026	−0.0618	−0.0852
	−0.0026	0.1818	0.0235	−0.1029
	−0.0618	0.0235	0.2504	−0.0079
	−0.0852	−0.1029	−0.0079	0.3783
3	0.4611	0.0381	−0.1189	0.0362
	0.0381	0.7006	0.0849	0.1239
	−0.1189	0.0849	0.3515	−0.0332
	0.0362	0.1239	−0.0332	0.3792

From Fig. 2 and Tables 1 and 2, we can see that after performing the JD operation, each class covariance matrix becomes close to be diagonal. Moreover, we can see that our JD algorithm achieves some improvement over Ziehe et al.'s JD algorithm in diagonalizing the covariance matrices of classes 1 and 3.

5.2. Experiments on Ekman's facial expression database

In this experiment, we use Ekman's POFA database to evaluate the recognition performance of the proposed DSA method and the KDSA method. To evaluate the performance of Ziehe's JD algorithm with ours, we use both JD algorithms to realize the DSA and the KDSA algorithms, respectively. Before the experiment, we preprocess the facial images by manually cropping each facial image such that the non-facial regions of each image are removed. Then we scale the cropped images to a size of 120 × 120 pixels. Fig. 3 shows examples of some cropped images. Finally, we concatenate each cropped image into a 14 400 × 1 vector and normalize it into a unit vector.

We adopt the leave-one-subject-out strategy to conduct the experiment. That is, to select the facial images of one subject as the testing data and use the rest as the training data. We repeat this procedure until all the facial images have been used once as the testing data. Table 3 shows the average test error rates of the various linear feature extraction methods as well as the various kernel based nonlinear feature extraction methods with different choices of monomial kernel degrees and Gaussian kernel parameters, where the monomial kernel degrees are set from 1 to 5 and the Gaussian kernel parameters are empirically fixed at 0.5, 5, 50, and 500. It can be clearly seen from Table 3 that the proposed KDSA method achieves a lowest error rate (=21.82%) among the various methods when Gaussian kernel is used. In this example, we see that the KDSA methods implemented by Ziehe's JD algorithms and by our JD algorithm achieve the similar better recognition results.

Table 2
Results of performing the Ziehe et al.'s JD algorithm and our JD algorithm, respectively, on the three class covariance matrices in Table 1.

Class no.	Covariance matrices calculated by Ziehe's algorithm			
1	0.0689	0.0322	−0.0018	0.0469
	0.0322	0.1109	−0.0062	0.0420
	−0.0018	−0.0062	0.5389	0.0033
	0.0469	0.0420	0.0033	0.1965
2	0.3788	−0.0064	0.0106	−0.0488
	−0.0064	0.1355	0.0109	−0.0243
	0.0106	0.0109	0.2149	0.0092
	−0.0488	−0.0243	0.0092	0.4633
3	0.5523	−0.0257	−0.0087	0.0019
	−0.0257	0.7537	−0.0048	−0.0176
	−0.0087	−0.0048	0.2462	−0.0125
	0.0019	−0.0176	−0.0125	0.3402
Class no.	**Covariance matrices calculated by our algorithm**			
1	0.0671	0.0311	−0.0022	0.0434
	0.0311	0.1081	−0.0064	0.0408
	−0.0022	−0.0064	0.5389	0.0030
	0.0434	0.0408	0.0030	0.2010
2	0.3822	−0.0112	0.0105	−0.0527
	−0.0112	0.1366	0.0105	−0.0276
	0.0105	0.0105	0.2149	0.0098
	−0.0527	−0.0276	0.0098	0.4588
3	0.5507	−0.0199	−0.0083	0.0094
	−0.0199	0.7553	−0.0041	−0.0133
	−0.0083	−0.0041	0.2462	−0.0128
	0.0094	−0.0133	−0.0128	0.3402

5.3. Experiments on the ORL face database

In this experiment, we aim to evaluate the representation ability and the recognition performance of the DSA method and the KDSA method. In the experiment, each face image is also cropped such that the non-facial regions of image are removed, and scale the cropped images to a size of 64 × 64 pixels. Then, we concatenate each face image into a 4096 × 1 vector and normalize it into a unit vector.

To show the performance on image representation, we randomly choose 70 images of seven subjects from the database and compute the representative vectors using the PCA method, the FKT method (a special case of the KFKT method when the monomial kernel with degree 1 is used) [1] and the DSA method, respectively. Fig. 4 shows the principal eigenvectors corresponding to the first seven largest eigenvalues of PCA (the first row) as well as the most representative vector of FKT (the second row) and DSA (the third row) corresponding to each subject, respectively. The images shown in the last row of Fig. 4 are the average image of each class. Moreover, to quantitatively evaluate the representative ability of the eigenvectors of PCA, FKT and DSA, we define the following average reconstruction error of the face images:

$$\varepsilon = \frac{1}{|\mathcal{I}|}\sum_{i \in \mathcal{I}} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|, \tag{30}$$

where

$$\hat{\mathbf{x}}_i = \omega\omega^T(\mathbf{x}_i - \overline{\mathbf{x}}) + \overline{\mathbf{x}}$$

Table 3
Average test error rates of various methods (%) on Ekman's POFA data set.

Methods	Average test error rates			
PCA	55.45			
LDA	25.45			
LDA/FKT	33.64			
LDA/GSVD	23.64			
DSA by Ziehe's JD	29.09			
DSA by proposed JD	28.18			
Monomial kernel	ker=2	ker=3	ker=4	ker=5
KPCA	55.45	55.45	55.45	55.45
KLDA	28.12	26.36	23.64	25.45
KFKT	32.73	34.55	32.73	32.73
KDSA by Ziehe's JD	28.18	30.91	28.18	26.36
KDSA by proposed JD	30.91	30.00	28.18	27.27
Gaussian kernel	$\sigma=0.5$	$\sigma=5$	$\sigma=50$	$\sigma=500$
KPCA	53.64	52.73	51.82	51.82
KLDA	26.36	27.27	26.36	25.45
KFKT	34.55	32.73	31.82	30.91
KDSA by Ziehe's JD	33.64	30.91	23.64	21.82
KDSA by proposed JD	31.82	31.82	21.82	21.82

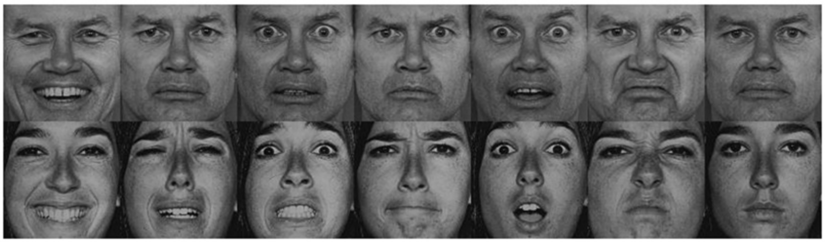

Fig. 3. Examples of cropped images of Ekman's POFA database.

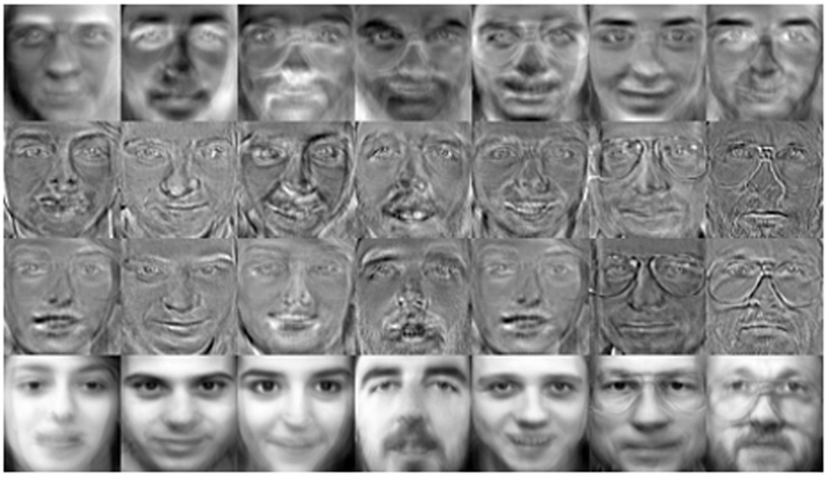

Fig. 4. Plot of the representative vectors obtained by PCA, FKT and DSA. The images in the first, second, and third rows denote the representative vectors of PCA, FKT [1], and DSA, respectively. The fourth row shows the average image of each class.

is the reconstruction of $\mathbf{x}_i$, $\mathcal{I}=\{\mathbf{x}_i\}$ denotes the whole face image set, $|\mathcal{I}|$ denotes the number of images in $\mathcal{I}$, $\overline{\mathbf{x}}=(1/|\mathcal{I}|)\sum_{i\in\mathcal{I}}\mathbf{x}_i$, and ω denotes a representative vector which can be computed by PCA, FKT, or DSA.

According to Eq. (30), we obtain that the average reconstruction errors of PCA, FKT and DSA are $\varepsilon_{PCA}=0.1781$, $\varepsilon_{FKT}=0.1480$ and $\varepsilon_{DSA}=0.1477$, respectively, where all 70 face images are used in the evaluation. For PCA, the first principal eigenvector is chosen as the representative vector, whereas, for both FKT and DSA, the principal eigenvector specific to each class is used as the representative error for calculating the average reconstruction error of the face images belonging to that class. From the average reconstruction error results and Fig. 4, we can see that, compared with the average images, both FKT and DSA achieve lower average reconstruction error than PCA and hence are more powerful than PCA in representing the subjects.

To evaluate the recognition performance of both DSA and KDSA, we randomly select five face images per subject as the training data and use the other five images as the testing data. Then, we use the training data set to train the various feature extraction algorithms and use the test data set to evaluate the recognition performance of the various methods. We totally conduct 10 trials of the experiments. The final test error rate is obtained by averaging the test error rates of all the trials. In the experiments, all the face images are concatenated into a vector and normalized into unit vectors. Table 4 shows the average test error rates and the standard deviations of the various methods, including the DSA method and the KDSA method implemented by Ziehe's JD algorithm. From Table 4, we see that the lower error rates are achieved when our DSA and KDSA methods are used. Especially, the lowest error rate ($=4.35\%$) is achieved when the DSA method via the proposed JD algorithm is used. In addition, the experimental results in this example show that the proposed JD algorithm achieves a slight better performance than Ziehe's JD algorithm when they are used in the DSA and the KDSA methods.

Table 4
Average test error rates (%) and standard deviations (shown in the brackets) of various methods on the ORL data set.

Methods	Average test error rates		
PCA	18.85 (2.86)		
LDA	6.10 (1.81)		
LDA/FKT	6.55 (1.88)		
LDA/GSVD	6.50 (1.83)		
DSA by Ziehe's JD	5.35 (2.19)		
DSA by proposed JD	**4.35 (1.78)**		
Monomial kernel	ker=2	ker=3	ker=4
KPCA	13.95 (2.91)	14.00 (2.73)	14.30 (2.46)
KLDA	5.55 (2.20)	5.55 (2.25)	5.95 (2.24)
KFKT	6.75 (2.44)	6.65 (2.80)	7.20 (2.94)
KDSA by Ziehe's JD	5.30 (2.61)	5.95 (2.47)	6.40 (2.66)
KDSA by proposed JD	4.55 (2.30)	5.25 (2.29)	5.15 (2.11)
Gaussian kernel	$\sigma=0.5$	$\sigma=5$	$\sigma=50$
KPCA	14.25 (2.53)	13.95 (2.71)	13.85 (2.86)
KLDA	6.10 (2.54)	5.90 (1.76)	6.05 (1.80)
KFKT	7.75 (3.29)	6.45 (2.63)	6.25 (2.81)
KDSA by Ziehe's JD	7.35 (2.65)	5.95 (2.27)	5.45 (1.79)
KDSA by proposed JD	6.45 (2.33)	4.95 (1.66)	5.05 (1.71)

5.4. Experiments on texture classification

In this experiment, we aim to evaluate the classification performance of the KDSA method on texture database under different kernel functions and different sizes of the training data set. The uniform local binary pattern, denoted by $\mathrm{LBP}_{8,2}^{u2}$ [18], is used to describe each texture image in the experiment. In this case, we totally obtain a 59-dimensional LBP vector to describe a texture image, resulting in a set of 59-dimensional feature vectors with 1248 samples.

To evaluate the recognition performance of various kernel based methods, we randomly select l ($=16$, 24, 32, 40) LBP vectors from each class as the training data and use the rest as the test data. We totally conduct 10 trials of the experiments and average the results as the final results. Figs. 5 and 6 show the average test error rates (%) of these methods with different choices of the monomial kernel degrees and the Gaussian kernel parameters, respectively. From both Figs. 5 and 6, we can see that our DSA/KDSA method achieves better results than the other methods.

6. Conclusions and discussions

In this paper, we have proposed a new DSA approach for multi-class feature extraction and classification problems. By adopting the kernel trick, we also extend the DSA method to deal with the nonlinear feature extraction problems. Moreover, to solve the related JD problem, we also propose a new algorithm

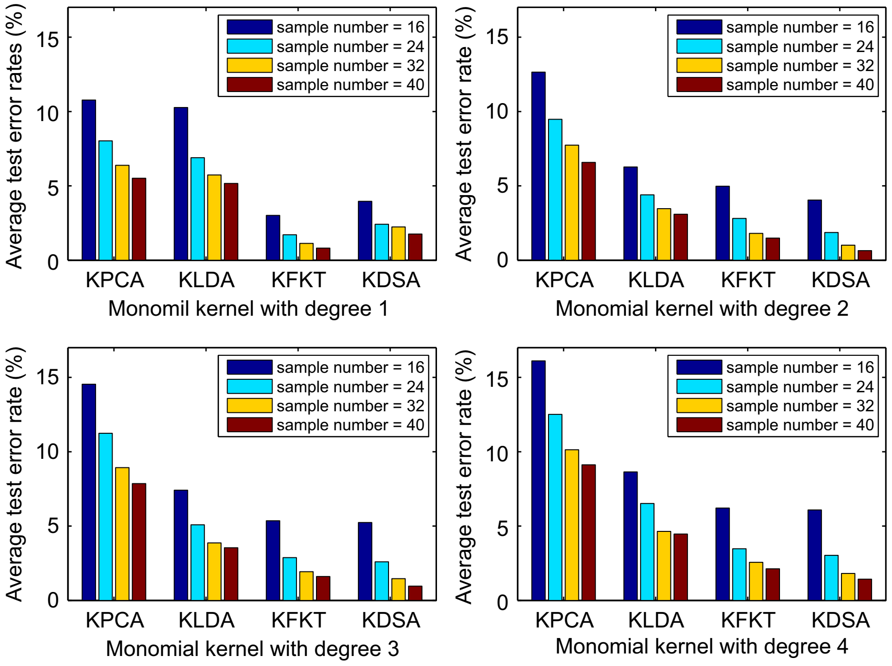

Fig. 5. Plots of the average test error rates of four methods with different choices of the monomial kernel degree.

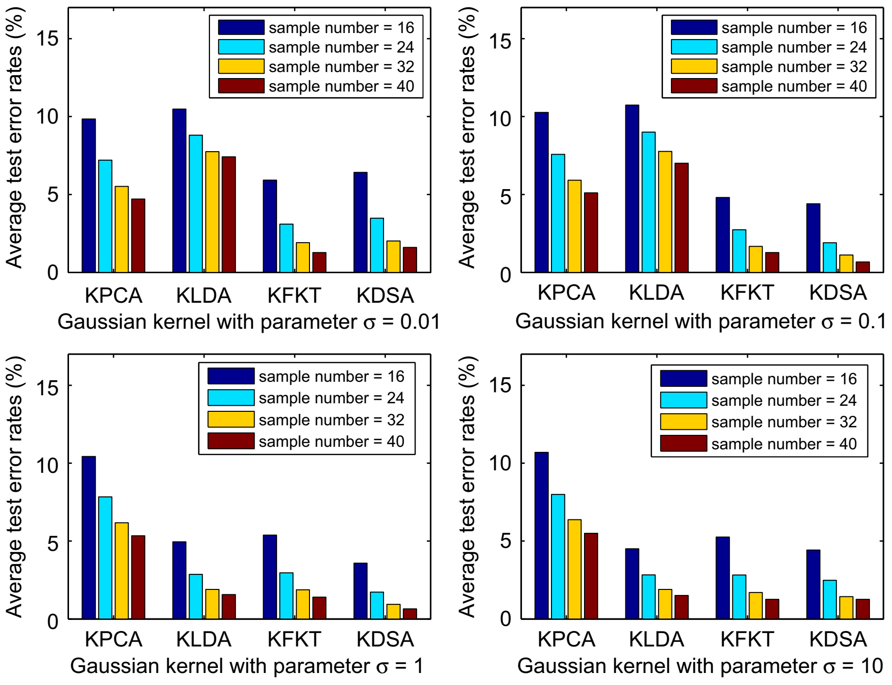

Fig. 6. Plots of the average test error rates of four methods with different choices of the Gaussian kernel parameters.

by using the conjugate gradient method on the Stiefel manifold. To evaluate the effectiveness of the new JD algorithm as well as the performance of the DSA method and the KDSA method, we conducted experiments on four real data sets, and the experimental results confirm the better performance of our methods. Additionally, in the experiments of facial expression recognition and face recognition, we see that the KDSA method may not achieve higher recognition rates than the linear DSA method. This problem may be due to the inappropriate choices of the kernel mappings or the kernel function parameters. To achieve a better performance of the KDSA method, we may use the kernel optimization approach [22], and that will be our further work.

Acknowledgements

The authors would like to thank the anonymous reviewers for their comments and suggestions. This work was partly supported by the National Basic Research Program of China (973 Program) under the grant no. 2011CB302202, and partly by the Natural

Science Foundation of China under Grant no. 60872160 and the Jiangsu Natural Science Foundation under Grant no. BK2010243.

Appendix A. Proof of Eq. (10)

$$\begin{aligned}
g(\mathbf{Q}) &= \frac{1}{4}\sum_i \|\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} - \mathrm{diag}(\mathbf{Q}^T\hat{\Sigma}_i\mathbf{Q})\|_F^2 \\
&= \frac{1}{4}\sum_i \mathrm{Tr}\left[\left(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} - \left(\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right)\right)\right. \\
&\quad \left.\times\left(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} - \left(\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right)\right)\right] \\
&= \frac{1}{4}\sum_i \mathrm{Tr}\left[\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{Q} - 2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\left(\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right)\right. \\
&\quad \left. + \left(\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)^2\mathbf{e}_j\mathbf{e}_j^T\right)\right], \qquad (31)
\end{aligned}$$

where $\mathbf{e}_j$ is the j-th column of the identity matrix and we have used the following identity:

$$\mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}) = \sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T. \qquad (32)$$

So the total differentiation of g is

$$\begin{aligned}
\mathrm{d}g = \frac{1}{4}\sum_i \mathrm{Tr}&\left[((\mathrm{d}\mathbf{Q})^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i^2\mathrm{d}\mathbf{Q})\right. \\
&-2((\mathrm{d}\mathbf{Q})^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathrm{d}\mathbf{Q})\left(\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right) \\
&-2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\left(\sum_j(\mathbf{e}_j^T((\mathrm{d}\mathbf{Q})^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathrm{d}\mathbf{Q})\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right) \\
&\left.+2\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)(\mathbf{e}_j^T((\mathrm{d}\mathbf{Q})^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathrm{d}\mathbf{Q})\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right]. \qquad (33)
\end{aligned}$$

The entry-wise differentiation of g w.r.t. the (p,q)-th entry $\mathbf{Q}_{pq}$ of $\mathbf{Q}$ is

$$\begin{aligned}
\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}_{pq}} &= \frac{1}{4}\sum_i \mathrm{Tr}\left[((\mathbf{e}_p\mathbf{e}_q^T)^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{e}_p\mathbf{e}_q^T)\right. \\
&\quad -2((\mathbf{e}_p\mathbf{e}_q^T)^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p\mathbf{e}_q^T)\left(\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right) \\
&\quad -2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\left(\sum_j(\mathbf{e}_j^T((\mathbf{e}_p\mathbf{e}_q^T)^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p\mathbf{e}_q^T)\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right) \\
&\quad \left.+2\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)(\mathbf{e}_j^T((\mathbf{e}_p\mathbf{e}_q^T)^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q} + \mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p\mathbf{e}_q^T)\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right] \\
&= \frac{1}{4}\sum_i \mathrm{Tr}\left[2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{e}_p\mathbf{e}_q^T - 2\mathbf{e}_q\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\right. \\
&\quad -2\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p\mathbf{e}_q^T \\
&\quad -2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\left(\sum_j(\mathbf{e}_j^T(\mathbf{e}_q\mathbf{e}_p^T)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right) \\
&\quad -2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\left(\sum_j(\mathbf{e}_j^T\mathbf{Q}\hat{\boldsymbol{\Sigma}}_j(\mathbf{e}_p\mathbf{e}_q^T)\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right) \\
&\quad +2\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)(\mathbf{e}_j^T((\mathbf{e}_q\mathbf{e}_p^T)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T \\
&\quad \left.+2\sum_j(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)(\mathbf{e}_j^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i(\mathbf{e}_p\mathbf{e}_q^T)\mathbf{e}_j)\mathbf{e}_j\mathbf{e}_j^T\right] \\
&= \frac{1}{4}\sum_i \mathrm{Tr}[2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{e}_p\mathbf{e}_q^T - 2\mathbf{e}_q\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}) \\
&\quad -2\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_j)\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p\mathbf{e}_q^T \\
&\quad -2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}((\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)\mathbf{e}_q\mathbf{e}_q^T) - 2\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}((\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p)\mathbf{e}_q\mathbf{e}_q^T) \\
&\quad +2(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)\mathbf{e}_q\mathbf{e}_q^T \\
&\quad +2(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p)\mathbf{e}_q\mathbf{e}_q^T] \\
&= \frac{1}{2}\sum_i [\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{e}_p - \mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\mathbf{e}_q \\
&\quad -\mathbf{e}_q^T\mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p - \mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}(\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)\mathbf{e}_q \\
&\quad -\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p)\mathbf{e}_q + \mathbf{e}_q^T(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)\mathbf{e}_q \\
&\quad +\mathbf{e}_q^T(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p)\mathbf{e}_q] \\
&= \frac{1}{2}\sum_i [\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i^2\mathbf{e}_p - \mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\mathbf{e}_q \\
&\quad -\mathbf{e}_q^T\mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p - (\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q) \\
&\quad -(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p) + (\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_p^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q) \\
&\quad +(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\mathbf{e}_q)(\mathbf{e}_q^T\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{e}_p)] \\
&= \frac{1}{2}\mathbf{e}_q^T\mathbf{Q}^T\left(\sum_i\hat{\boldsymbol{\Sigma}}_i^2\right)\mathbf{e}_p - \mathbf{e}_q^T\left(\sum_i\mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\right)\mathbf{e}_p. \qquad (34)
\end{aligned}$$

So rearranging the above entry-wise differentiation in a matrix form, we have

$$\begin{aligned}
\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}} &= \frac{1}{2}\left[\mathbf{Q}^T\left(\sum_i\hat{\boldsymbol{\Sigma}}_i^2\right)\right]^T - \left(\sum_i\mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q})\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\right)^T \\
&= \frac{1}{2}\left(\sum_i\hat{\boldsymbol{\Sigma}}_i^2\right)\mathbf{Q} - \sum_i\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}\ \mathrm{diag}(\mathbf{Q}^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}). \qquad (35)
\end{aligned}$$

Appendix B. Proof of Eq. (11)

Denote $g_k(t) = g(\mathbf{Q}_k(t))$, then $f_k(t) = \mathrm{d}g_k(t)/\mathrm{d}t$. Note that

$$\begin{aligned}
f_k(t) = \frac{\mathrm{d}g_k(t)}{\mathrm{d}t} &= \mathrm{Tr}\left(\left.\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}}\right|_{\mathbf{Q}_k(t)}\left(\frac{\mathrm{d}\mathbf{Q}_k(t)}{\mathrm{d}t}\right)^T\right) = \mathrm{Tr}\left(\left.\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}}\right|_{\mathbf{Q}_k(t)}\left(\frac{\mathrm{d}\mathbf{Q}_k e^{t\mathbf{A}_k}}{\mathrm{d}t}\right)^T\right) \\
&= \mathrm{Tr}\left(\left.\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}}\right|_{\mathbf{Q}_k(t)}(\mathbf{Q}_k e^{t\mathbf{A}_k}\mathbf{A}_k)^T\right) = \mathrm{Tr}\left(\left.\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}}\right|_{\mathbf{Q}_k(t)}(\mathbf{Q}_k(t)\mathbf{A}_k)^T\right) \\
&= \mathrm{Tr}\left(\left.\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}}\right|_{\mathbf{Q}_k(t)}\mathbf{A}_k^T\mathbf{Q}_k^T(t)\right). \qquad (36)
\end{aligned}$$

Consider that $\mathbf{A}_k$ is a skew-symmetric matrix, i.e., $\mathbf{A}_k^T = -\mathbf{A}_k$, we obtain that Eq. (36) can be rewritten as

$$f_k(t) = -\mathrm{Tr}\left(\left.\frac{\mathrm{d}g}{\mathrm{d}\mathbf{Q}}\right|_{\mathbf{Q}_k(t)}\mathbf{A}_k\mathbf{Q}_k^T(t)\right). \qquad (37)$$

Let $\mathbf{S}_{i,k}(t) = \mathbf{Q}_k^T(t)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t)$ and $\mathbf{S}_k^{(2)} = \mathbf{Q}_k^T(t)(\sum_i\hat{\boldsymbol{\Sigma}}_i^2)\mathbf{Q}_k(t)$. Substituting Eq. (10) into (37), we have

$$\begin{aligned}
f_k(t) &= -\mathrm{Tr}\left(\left[\frac{1}{2}\left(\sum_i\hat{\boldsymbol{\Sigma}}_i^2\right)\mathbf{Q}_k(t) - \sum_i\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t)\mathrm{diag}(\mathbf{Q}_k^T(t)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t))\right]\mathbf{A}_k\mathbf{Q}_k^T(t)\right) \\
&= -\mathrm{Tr}\left(\mathbf{A}_k\left[\frac{1}{2}\mathbf{Q}_k^T(t)\left(\sum_i\hat{\boldsymbol{\Sigma}}_i^2\right)\mathbf{Q}_k(t) - \sum_i\mathbf{S}_{i,k}(t)\mathrm{diag}(\mathbf{S}_{i,k}(t))\right]\right) \\
&= -\frac{1}{2}\mathrm{Tr}(\mathbf{A}_k\mathbf{S}_k^{(2)}(t)) + \mathrm{Tr}\left(\mathbf{A}_k\left[\sum_i\mathbf{S}_{i,k}(t)\mathrm{diag}(\mathbf{S}_{i,k}(t))\right]\right). \qquad (38)
\end{aligned}$$

From the fact that $\mathbf{A}_k^T = -\mathbf{A}_k$ and $\mathbf{S}_k^{(2)}$ is a symmetric matrix, we have

$$f_k(t) = \mathrm{Tr}\left(\mathbf{A}_k\left[\sum_i \mathbf{S}_{i,k}(t)\mathrm{diag}(\mathbf{S}_{i,k}(t))\right]\right). \tag{39}$$

Appendix C. Proof of Eq. (12)

First we write

$$\frac{\mathrm{d}f_k(t)}{\mathrm{d}t} = \mathrm{Tr}\left(\mathbf{A}_k\left[\sum_i\left(\frac{\mathrm{d}(\mathbf{S}_{i,k}(t))}{\mathrm{d}t}\mathrm{diag}(\mathbf{S}_{i,k}(t)) + \mathbf{S}_{i,k}(t)\frac{\mathrm{d}(\mathrm{diag}(\mathbf{S}_{i,k}(t)))}{\mathrm{d}t}\right)\right]\right). \tag{40}$$

On the other hand, we have that

$$\begin{aligned}\frac{\mathrm{d}(\mathbf{S}_{i,k}(t))}{\mathrm{d}t} &= \frac{\mathrm{d}(\mathbf{Q}_k^T(t))}{\mathrm{d}t}\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t) + \mathbf{Q}_k^T(t)\hat{\boldsymbol{\Sigma}}_i\frac{\mathrm{d}(\mathbf{Q}_k(t))}{\mathrm{d}t}\\ &= (\mathbf{Q}_k(t)\mathbf{A}_k)^T\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t) + \mathbf{Q}_k^T(t)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t)\mathbf{A}_k\\ &= \mathbf{A}_k^T\mathbf{Q}_k^T(t)\hat{(}t)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t) + \mathbf{Q}_k^T(t)\hat{\boldsymbol{\Sigma}}_i\mathbf{Q}_k(t)\mathbf{A}_k\\ &= (\mathbf{S}_{i,k}(t)\mathbf{A}_k)^T + \mathbf{S}_{i,k}(t)\mathbf{A}_k\end{aligned} \tag{41}$$

and

$$\frac{\mathrm{d}(\mathrm{diag}(\mathbf{S}_{i,k}(t)))}{\mathrm{d}t} = \mathrm{diag}\left(\frac{\mathrm{d}(\mathbf{S}_{i,k}(t))}{\mathrm{d}t}\right) = \mathrm{diag}((\mathbf{S}_{i,k}(t)\mathbf{A}_k)^T + \mathbf{S}_{i,k}(t)\mathbf{A}_k). \tag{42}$$

Thus, we obtain that

$$\begin{aligned}\frac{\mathrm{d}f_k(t)}{\mathrm{d}t} = \mathrm{Tr}\Bigg(\mathbf{A}_k\sum_i[((\mathbf{S}_{i,k}(t)\mathbf{A}_k)^T + \mathbf{S}_{i,k}(t)\mathbf{A}_k)\mathrm{diag}(\mathbf{S}_{i,k}(t))\\ + \mathbf{S}_{i,k}(t)\mathrm{diag}((\mathbf{S}_{i,k}(t)\mathbf{A}_k)^T + \mathbf{S}_{i,k}(t)\mathbf{A}_k)]\Bigg).\end{aligned} \tag{43}$$

References

[1] Y.-H. Li, M. Savvides, Kernel Fukunaga–Koontz transform subspaces for enhanced face recognition, in: Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'07), 2007, pp. 1–8.

[2] K. Fukunaga, Introduction to Statistical Pattern Recognition, second ed., Academic Press, New York, 1990.

[3] K. Fukunaga, W.L.G. Koontz, Application of the Karhunen–Loeve expansion to feature selection and ordering, IEEE Transactions on Computers C-26 (1970) 281–289.

[4] J.R. Leger, S.H. Lee, Image classification by an optical implementation of the Fukunaga–Koontz transform, Journal of the Optical Society of America 72 (5) (1982) 556–564.

[5] M.-H. Yang, D.J. Kriegman, N. Ahuja, Detecting faces in images: a survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (1) (2002) 34–58.

[6] S. Zhang, T. Sim, When fisher meets Fukunaga–Koontz: a new look at linear discriminants, in: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), 2006, pp. 323–329.

[7] S. Zhang, T. Sim, Discriminant subspace analysis: a Fukunaga–Koontz approach, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (10) (2007) 1732–1745.

[8] B. Flury, Common principal components in *k* groups, Journal of the American Statistical Association 79 (388) (1984) 892–898.

[9] A. Ziehe, P. Laskov, G. Nolte, K.-R. Müller, A fast algorithm for joint diagonalization with non-orthogonal transformations and its application to blind source separation, Journal of Machine Learning Research 5 (2004) 777–800.

[10] A. Edelman, T.A. Arias, S.T. Smith, The geometry of algorithms with orthogonality constraints, SIAM Journal on Matrix Analysis and Applications 20 (2) (1998) 302–353.

[11] B. Schölkopf, A. Smola, K.R. Müller, Nonlinear component analysis as a kernel Eigenvalue problem, Neural Computation 10 (1998) 1299–1319.

[12] G. Baudat, F. Anouar, Generalized discriminant analysis using a kernel approach, Neural Computation 12 (2000) 2385–2404.

[13] A. Quarteroni, R. Sacco, F. Saleri, Numerical Mathematics, Springer, New York, 2000.

[14] R.A. Fisher, The use of multiple measurements in taxonomic problems, Annals of Eugenics 7 (1936) 179–188.

[15] P. Ekman, W.V. Friesen, Pictures of Facial Affect, University of California Medical Center, CA, 1976.

[16] W. Zheng, L. Zhao, C. Zou, A modified algorithm for generalized discriminant analysis, Neural Computation 16 (6) (2004) 1283–1297.

[17] T. Brodatz, Textures: A Photographic Album for Artists and Designers, Dover, New York, 1996.

[18] T. Ojala, M. Pietikäinen, T. Mäenpää, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 971–987.

[19] P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, Eigenfaces vs. Fisherfaces: recognition using class specific linear projection, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (7) (1997) 711–720.

[20] P. Howland, H. Par, Generalizing discriminant analysis using the generalized singular value decomposition, IEEE Transactions Pattern Analysis and Machine Intelligence 26 (8) (2004) 995–1006.

[21] M.H. Yang, Kernel Eigenfaces vs. kernel Fisherfaces: face recognition using kernel methods, in: Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition, 2002.

[22] H. Xiong, M.N.S. Swamy, M.O. Ahmad, Optimizing the kernel in the empirical feature space, IEEE Transactions on Neural Networks 16 (2) (2005) 460–474.

Wenming Zheng received the Ph.D. degree in signal processing from Southeast University in 2004. He is currently a professor in Research Center for Learning Science (RCLS), Southeast University, China. His research interests include neural computation, pattern recognition, machine learning, and computer vision.

Zhouchen Lin received the Ph.D. degree in applied mathematics from Peking University in 2000. He is currently a researcher in Visual Computing Group, Microsoft Research Asia. His research interests include computer vision, computer graphics, pattern recognition, statistical learning, document processing, and human computer interaction.