Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 40 (2007) 2786–2797
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

Face recognition using spectral features

Fei Wang[a,*], Jingdong Wang[b], Changshui Zhang[a], James Kwok[b]

[a]*Department of Automation, State Key Laboratory of Intelligent Technology and Systems, Tsinghua University, Beijing 100084, PR China*
[b]*Department of Computer Science, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong*

Received 22 January 2006; received in revised form 9 January 2007; accepted 18 January 2007

Abstract

Face recognition is a challenging task in computer vision and pattern recognition. It is well-known that obtaining a low-dimensional feature representation with enhanced discriminatory power is of paramount importance to face recognition. Moreover, recent research has shown that the face images reside on a possibly nonlinear manifold. Thus, how to effectively exploit the hidden structure is a key problem that significantly affects the recognition results. In this paper, we propose a new unsupervised nonlinear feature extraction method called *spectral feature analysis* (SFA). The main advantages of SFA over traditional feature extraction methods are: (1) SFA does not suffer from the small-sample-size problem; (2) SFA can extract discriminatory information from the data, and we show that linear discriminant analysis can be subsumed under the SFA framework; (3) SFA can effectively discover the nonlinear structure hidden in the data. These appealing properties make SFA very suitable for face recognition tasks. Experimental results on three benchmark face databases illustrate the superiority of SFA over traditional methods.

Keywords: Face recognition; Spectral features; Kernel

1. Introduction

In the past few decades, face recognition has been an active research topic and a variety of methods have been proposed. From the face representation viewpoint, these methods can be classified into two categories: geometric feature-based approach [1–3] and template-based approach [4–6]. The former approach analyzes explicit local features (such as the eyes, mouth and nose) and their geometric relationships. Representative works include the *Hidden Markov Model* (HMM) [2] and the elastic bunch graph matching algorithm [3]. However, perfect extraction of the local features is difficult to implement [1]. On the other hand, template-based methods determine the face identity by measuring the correlation between the face and some reference templates. Here, a face image of size $n \times m$ is often represented by a vector in $\mathbb{R}^{nm}$, which can be very high-dimensional for typical values of n and m. Many dimension reduction or feature extraction techniques have been proposed in this context [4–6], and the most prominent examples are *principal component analysis* (PCA) [4] and *linear discriminant analysis* (LDA) [5].

PCA is a popular unsupervised method which aims at extracting a subspace in which the variance of the projected data is maximized (or, equivalently, the reconstruction error is minimized). However, in face recognition, it has been observed that *intra-person* variations (i.e., variations of the same person's face due to illumination, expression, viewing direction, etc.) are often larger than *inter-person* variations (i.e., variations due to changes in the person's identity) [7]. In using PCA for face recognition (the so-called *eigenface* method [4]), some unwanted intra-person variations might still be retained in the PCA projections (see, e.g., Ref. [8]), and thus PCA is suboptimal for classification.

LDA is a supervised method which searches for a discriminative subspace where patterns belonging to the same class are as tight as possible while patterns belonging to the other classes are more separated. Because of the use of class information, LDA-based algorithms often perform better than PCA-based algorithms [5]. However, LDA suffers from the *small-sample-size* problem. This is particularly problematic in

* Corresponding author. Tel.: +86 10 62796872; fax: +86 10 62786911.
E-mail address: feiwang03@mails.tsinghua.edu.cn (F. Wang).

0031-3203/$30.00
doi:10.1016/j.patcog.2007.01.023

face recognition applications [5,9], where there are typically very few training samples per class compared to the high dimensionality of the face images, and it in turn makes the between-class scatter matrix singular. Moreover, LDA is also more sensitive to the particular choice of the training set, which makes LDA sometimes perform even poorer than PCA [10].

In recent years, it has been revealed that face images often reside on a nonlinear manifold [11–13]. However, both PCA and LDA work efficiently only in the Euclidean space and are not good at discovering the nonlinear structure hidden in the face image manifold. Recently, *locality preserving projection* (LPP) [6] is proposed which can preserve the local structure of the face images. However, LPP is still a linear method, and thus hard to handle the nonlinear structure existing in the face images. Moreover, LPP still encounters the small-sample-size problem [6]. Although we can employ the technique in Ref. [5] to first reduce the face dimensionality by using PCA, some useful information may also be lost in the process [9].

Our work here is motivated by the recent works on spectral clustering [14–16], which is a class of clustering methods based on the spectral graph theory [17]. These clustering methods usually take two steps. First, the patterns are embedded in a lower-dimensional space such that the clusters are more "obvious" or separate. Then, a classical clustering algorithm (such as the K-means clustering algorithm) is performed in the embedded space [16]. Spectral clustering can yield impressively good results when traditional clustering approaches fail, and it has been shown to be very powerful in fields such as computer vision [14] and VLSI design [18].

Inspired by the success of spectral clustering, we present in this paper a related unsupervised feature extraction method called *spectral feature analysis* (SFA). The embedded data resulting from the first step of spectral clustering will be referred to as *spectral features*. Since spectral clustering can only compute spectral features for the training set, a natural question is how to obtain spectral features for the unseen (i.e., testing) data. To address this problem, we first propose a weighted extension of the standard *kernel principal component analysis* (KPCA) [19] called *weighted kernel principal component analysis* (WKPCA).[1] Then, we show that spectral clustering is a special case of WKPCA and, consequently, spectral features of the test set can be obtained by WKPCA.

Our method is particularly suitable for face recognition because of the following properties:

(1) SFA does not suffer from the small-sample-size problem. In both LDA and LPP, the resultant computational problems involve generalized eigenvalue problems, which in turn require matrix inversions. The sizes of the matrices involved (namely, $\mathbf{S}_w$ in Ref. [5], and $\mathbf{XDX}^{\mathrm{T}}$ in Ref. [6]) are dependent on the dimensionality of the image, which are often very large compared to the number of training images. These matrices may thus be singular and the small-sample-size problem occurs. On the other hand, although the resultant computational problem in SFA is still a generalized eigenvalue problem, the size of the matrix that has to be inverted is only dependent on the number of training patterns. Hence, the small-sample-size problem will not occur.

(2) As spectral clustering can effectively group patterns into clusters, SFA can also effectively extract discriminative information from the data. Furthermore, we will prove that LDA can be subsumed under the SFA framework.

(3) SFA can effectively discover the intrinsic nonlinear manifold structure hidden in the data. Indeed, we will prove that *kernel* LPP is SFA. In other words, our method can exploit and preserve the data's nonlinear manifold structure.

[1] Note that WKPCA is the kernel version of WPCA [20], however, the weights in our method are derived from spectral clustering, not derived from robust analysis as in Ref. [20].

Table 1
Notations

$\mathbf{X}$	Data matrix ($[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M]$)
$\mathbf{\Phi}$	Data matrix in the feature space ($[\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \ldots, \Phi(\mathbf{x}_M)]$)
$\mathbf{W}$	Weight matrix
$\mathbf{D}$	Degree matrix
$\mathbf{S}$	Data covariance matrix
$\mathbf{C}$	Data covariance matrix in the feature space
$\mathbf{m}$	Data mean vector
$\boldsymbol{\mu}$	Data mean vector in the feature space
$(\theta^k, \mathbf{u}^k)$	kth eigenvalue–eigenvector pair of $\mathbf{S}$
$(\lambda^k, \mathbf{v}^k)$	kth eigenvalue–eigenvector pair of $\mathbf{C}$
$\mathbf{A}$	Similarity or affinity matrix
$\mathbf{K}$	Kernel matrix
$\boldsymbol{\alpha}^k$	kth eigenvector of $\mathbf{A}$

The rest of this paper is organized as follows. Section 2 first introduces some related works. Section 3 describes *weighted* PCA (WPCA), WKPCA and SFA in detail, which is then followed by some analysis and discussions in Section 4. Experimental results are presented in Section 5, and the last section gives some concluding remarks.

In the sequel, bold capital letters are for matrices while bold lower cases are for vectors. Subscript $_i$ represents the ith component of a vector, and subscript $_{ij}$ represents the (i, j)th entry of a matrix. $\mathbf{A}^{\mathrm{T}}$ denotes the transpose of $\mathbf{A}$. We use the tilde $\tilde{(\cdot)}$ to denote the weighted version of $(\cdot)$. For example, when $\mathbf{S}$ is the data covariance matrix, then $\tilde{\mathbf{S}}$ is the weighted data covariance matrix. More notations are listed in Table 1.

2. Related works

2.1. Spectral clustering and spectral classification

In this section, we first give a brief review on spectral clustering. There are several variants of spectral clustering (e.g., Refs. [14,15,21,22]). In the sequel, our focus will be on the normalized cuts method [14], which is one of the most effective spectral clustering methods. The same parlance can be referred to Ref. [16].

Given the data set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M\}$, with each $\mathbf{x}_i \in \mathbb{R}^d$. It can be represented by a weighted undirected graph $G = (\mathscr{V}, \mathscr{E})$,

where $\mathscr{V}$ (the set of vertices) contains all the $\mathbf{x}_i$'s and $\mathscr{E}$ (the set of edges) contains the pairwise similarities among $\mathbf{x}_i$'s. We first consider the simpler problem in which we want to separate the data into two (hard) clusters. This clustering problem is then equivalent to a graph partitioning problem, where one partitions $\mathscr{V}$ into two subsets, $\mathscr{V}_1$ and $\mathscr{V}_2$, such that $\mathscr{V}_1 \cap \mathscr{V}_2 = \phi$ and $\mathscr{V}_1 \cup \mathscr{V}_2 = \mathscr{V}$. Many criteria can be used to measure the quality of the partitioning result. In particular, the normalized cuts criterion is defined as [14]

$$\mathrm{Ncut}(\mathscr{V}_1, \mathscr{V}_2) = \frac{\mathrm{cut}(\mathscr{V}_1, \mathscr{V}_2)}{\mathrm{assoc}(\mathscr{V}_1, \mathscr{V})} + \frac{\mathrm{cut}(\mathscr{V}_1, \mathscr{V}_2)}{\mathrm{assoc}(\mathscr{V}_2, \mathscr{V})}, \tag{1}$$

where $\mathrm{cut}(\mathscr{V}_1, \mathscr{V}_2) = \sum_{u\in\mathscr{V}_1, v\in\mathscr{V}_2} a(u, v)$ and $\mathrm{assoc}(\mathscr{V}_i, \mathscr{V}) = \sum_{u\in\mathscr{V}_i, t\in\mathscr{V}} a(u, t)$, with $a(u, v)$ being the weight of the edge connecting vertices u and v. In general, instead of having only a bipartition, one can have a K-way cut [23], which partitions $\mathscr{V}$ into K subsets $\{\mathscr{V}_i\}_{i=1}^K$ such that $\mathscr{V}_i \cap \mathscr{V}_j = \phi$ for $i \neq j$ and $\cup_{i=1}^K \mathscr{V}_i = \mathscr{V}$. The desired partitioning can be represented by a $M \times K$ matrix $\mathbf{Q} = [\mathbf{Q}_{i\ell}]$, where $\mathbf{Q}_{i\ell} = 1$ if $i \in \mathscr{V}_\ell$ and 0 otherwise. The normalized cuts criterion in Eq. (1) then becomes

$$\mathrm{Ncut}(\mathscr{V}_1, \mathscr{V}_2, \ldots, \mathscr{V}_K) = \sum_{i=1}^{K} \frac{\mathrm{cut}(\mathscr{V}_i, \mathscr{V}\backslash\mathscr{V}_i)}{\mathrm{assoc}(\mathscr{V}_i, \mathscr{V})}, \tag{2}$$

where $\mathrm{cut}(\mathscr{V}_i, \mathscr{V}\backslash\mathscr{V}_i)$ measures the number of links escaping from $\mathscr{V}_i$, and $\mathrm{assoc}(\mathscr{V}_i, \mathscr{V})$ is the total number of connections from nodes in $\mathscr{V}_i$ to the other nodes in the graph.

Yu et al. [23] showed that minimization of Eq. (2) can be relaxed to the following optimization problem:

$$\begin{aligned} \max_{\mathbf{Z}} \quad & \varepsilon(\mathbf{Z}) = \frac{1}{K}\mathrm{trace}(\mathbf{Z}^{\mathrm{T}}\mathbf{A}\mathbf{Z}) \\ \text{s.t.} \quad & \mathbf{Z}^{\mathrm{T}}\mathbf{D}\mathbf{Z} = \mathbf{I}_K, \end{aligned} \tag{3}$$

where $\mathbf{Z} = \mathbf{Q}(\mathbf{Q}^{\mathrm{T}}\mathbf{D}\mathbf{Q})^{-1/2}$, $\mathbf{D} = \mathrm{diag}(d_{11}, d_{22}, \ldots, d_{MM})$ is the $M \times M$ degree matrix with $d_{ii} = \sum_{u\in\mathscr{V}} a(i, u)$, $\mathbf{A}$ is the $M \times M$ affinity matrix with $\mathbf{A}_{ij} = a(i, j)$, and $\mathbf{I}_K$ is the $K \times K$ identity matrix. Using the Rayleigh-Ritz theorem [24], the optimal $\mathbf{Z}$ can be obtained from the generalized eigensystem

$$\mathbf{A}\mathbf{Z} = \lambda\mathbf{D}\mathbf{Z}. \tag{4}$$

Each row of $\mathbf{Z}$ gives the embedding coordinates of the corresponding pattern, which can then be input to standard clustering methods. Besides using the rows of $\mathbf{Z}$ for (spectral) clustering, one can also input the extracted features to a standard classifier, leading to *spectral classification* [25].

However, as $\mathbf{Z}$ in Eq. (4) only involves the training set, a natural problem is how to perform feature extraction on the test data. To alleviate this problem, Ref. [25] assumes a transductive learning setting, where both the training and test data have to be available before feature extraction. Eigendecomposition is then performed on the matrices constructed from both the training and test sets. However, this transductive learning setting is obviously not always possible.

2.2. (*Kernel*) *principal component analysis*

The use of superfluous features often leads to inferior performance in pattern recognition. A general practical observation is that it is worth decreasing the dimensionality of the feature space while ensuring that the overall structure of the data points remains intact. A simple way to do this is by means of a transformation that linearly maps the initial feature space to a new one with fewer dimensions. A very popular technique is (PCA) [26], which chooses the basis vectors of the transformed space as those directions of the original space along which the data show a large variance. Denote the data matrix by $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M] \in \mathbb{R}^{d\times M}$. It is well-known that these basis vectors are given by the leading eigenvectors (i.e., eigenvectors with the largest eigenvalues) of the sample covariance matrix

$$\mathbf{S} = \frac{1}{M}\sum_{i=1}^{M}(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^{\mathrm{T}}, \tag{5}$$

where $\mathbf{m} = (1/M)\sum_{i=1}^{M}\mathbf{x}_i$ is the mean vector.

When linear transformations like PCA are not powerful enough, we can turn to nonlinear feature extraction methods. By performing PCA in some kernel-induced feature space, KPCA extracts features that are nonlinearly related to the input variables [19]. Denote the kernel function by k, the corresponding kernel-induced feature space by $\mathscr{F}$ (whose dimensionality may be infinite), and the corresponding nonlinear mapping by $\Phi : \mathbb{R}^d \mapsto \mathscr{F}$. Assuming that the mapped data have been centered in the feature space, i.e., $\sum_{i=1}^{M}\Phi(\mathbf{x}_i) = \mathbf{0}$, the covariance matrix is simply $\mathbf{C} = (1/M)\sum_{i=1}^{M}\Phi(\mathbf{x}_i)\Phi(\mathbf{x}_i)^{\mathrm{T}}$. Eigendecomposition of $\mathbf{C}$ gives $\lambda\mathbf{v} = \mathbf{C}\mathbf{v} = (1/M)\sum_{i=1}^{M}(\Phi(\mathbf{x}_i)^{\mathrm{T}}\mathbf{v})\Phi(\mathbf{x}_i)$, and so $\mathbf{v} \in \mathrm{span}\{\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \ldots, \Phi(\mathbf{x}_M)\}$. Let $\mathbf{v} = \sum_{i=1}^{M}\alpha_i\Phi(\mathbf{x}_i)$ for some $\alpha_i \in \mathbb{R}$, we have

$$\lambda\sum_{i=1}^{M}\alpha_i\Phi(\mathbf{x}_k)^{\mathrm{T}}\Phi(\mathbf{x}_i) = \frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{M}\alpha_j\Phi(\mathbf{x}_k)^{\mathrm{T}}\Phi(\mathbf{x}_i)\Phi(\mathbf{x}_i)^{\mathrm{T}} \times \Phi(\mathbf{x}_j). \tag{6}$$

Define the kernel matrix $\mathbf{K} = [\mathbf{K}_{ij}]$ with $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^{\mathrm{T}}\Phi(\mathbf{x}_j)$. Eq. (6) can then be written as $M\lambda\mathbf{K}\boldsymbol{\alpha} = \mathbf{K}^2\boldsymbol{\alpha}$ where $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_M]^{\mathrm{T}}$. As $\mathbf{K}$ is symmetric, it has a set of eigenvectors which spans the whole space, thus $M\lambda\boldsymbol{\alpha} = \mathbf{K}\boldsymbol{\alpha}$. The projection of the mapped data $\Phi(\mathbf{x}_i)$ on the kth principal component is

$$\begin{aligned} \Phi(\mathbf{x}_i)^{\mathrm{T}}\mathbf{v}^k = \Phi(\mathbf{x}_i)\cdot\sum_{j=1}^{M}\alpha_j^k\Phi(\mathbf{x}_j) &= \sum_{j=1}^{M}\alpha_j^k\Phi(\mathbf{x}_i)^{\mathrm{T}}\Phi(\mathbf{x}_j) \\ &= \sum_{j=1}^{M}\alpha_j^k\mathbf{K}_{ij}. \end{aligned}$$

Therefore, all these can be computed once the kernel function is known.

3. Spectral feature analysis

We have briefly reviewed traditional PCA and KPCA in the last section. Their basic assumption is that the training data are *independently and identically distributed* (i.i.d.), so that all the data points are treated equally (the contribution of each data point to the covariance matrix is $1/M$). However, this may not always be the case in real-world problems. For example, in face recognition, the face images are usually sampled from a video sequence, and this causes the obtained images to be related to each other. So it will be more reasonable if different weights can be assigned to the data points. Moreover, this weight should reflect the importance of each data point in the training set. In this section, we will first introduce such *weighted* versions of PCA and KPCA. Then, we will propose SFA, which can be viewed as a special case of *weighted* KPCA.

3.1. Weighted PCA and weighted KPCA

In Eq. (5), all the $\mathbf{x}_i$'s are implicitly assumed to have equal importance. Here, we consider the general case where each $\mathbf{x}_i$ carries a weight of $p_i > 0$. Without loss of generality, we assume $\sum_{i=1}^{M} p_i = 1$. The weighted covariance matrix is then

$$\tilde{\mathbf{S}} = \sum_{i=1}^{M} p_i (\mathbf{x}_i - \tilde{\mathbf{m}})(\mathbf{x}_i - \tilde{\mathbf{m}})^{\mathrm{T}}, \tag{7}$$

where $\tilde{\mathbf{m}} = \sum_{i=1}^{M} p_i \mathbf{x}_i$ is the weighted mean. Assume that the data have been centered (w.r.t. the weights p_i's), we have $\tilde{\mathbf{m}} = \mathbf{0}$ and $\tilde{\mathbf{S}} = \sum_{i=1}^{M} p_i \mathbf{x}_i \mathbf{x}_i^{\mathrm{T}} = \mathbf{X}\mathbf{P}\mathbf{X}^{\mathrm{T}}$, where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M]$ and $\mathbf{P} = \mathrm{diag}(p_1, p_2, \ldots, p_M)$. The desired principal components are the leading eigenvectors of $\tilde{\mathbf{S}}$.

The corresponding kernel extension is straightforward. The weighted covariance matrix in the kernel-induced feature space is $\tilde{\mathbf{C}} = \sum_{i=1}^{M} p_i \Phi(\mathbf{x}_i)\Phi(\mathbf{x}_i)^{\mathrm{T}}$. Again, assume that the data have been centered,[2] i.e., $\sum_{i=1}^{M} p_i \Phi(\mathbf{x}_i) = \mathbf{0}$. Eigendecomposition of $\tilde{\mathbf{C}}$ gives

$$\tilde{\lambda}\tilde{\mathbf{v}} = \tilde{\mathbf{C}}\tilde{\mathbf{v}} = \sum_{i=1}^{M} (w_i \Phi(\mathbf{x}_i)^{\mathrm{T}} \tilde{\mathbf{v}})(w_i \Phi(\mathbf{x}_i)),$$

where $w_i = \sqrt{p_i}$. Again, $\tilde{\mathbf{v}}$ lies in the span of $\{w_i \Phi(\mathbf{x}_i)\}_{i=1}^{M}$. Let

$$\tilde{\mathbf{v}} = \sum_{i=1}^{M} \tilde{\alpha}_i w_i \Phi(\mathbf{x}_i),$$

for some $\tilde{\alpha}_i \in \mathbb{R}$. Proceeding as in Section 2.2, we obtain the eigensystem

$$\tilde{\mathbf{K}}\tilde{\boldsymbol{\alpha}} = \tilde{\lambda}\tilde{\boldsymbol{\alpha}}, \tag{8}$$

where $\tilde{\mathbf{K}} = [\tilde{\mathbf{K}}_{ij}]$ with $\tilde{\mathbf{K}}_{ij} = (w_i \Phi(\mathbf{x}_i))^{\mathrm{T}}(w_j \Phi(\mathbf{x}_j))$. Alternatively, $\tilde{\mathbf{K}}$ can be written as

$$\tilde{\mathbf{K}} = \mathbf{W}\mathbf{K}\mathbf{W}, \tag{9}$$

where $\mathbf{W} = \mathrm{diag}(w_1, w_2, \ldots, w_M)$. We have to normalize the eigenvectors of the covariance matrix $\tilde{\mathbf{C}}$ through scaling the eigenvectors of $\tilde{\mathbf{K}}$ by $\hat{\boldsymbol{\alpha}}^i = \tilde{\boldsymbol{\alpha}}^i / \sqrt{\tilde{\lambda}_i}$, where $\tilde{\lambda}_i$ is the eigenvalue of $\tilde{\mathbf{K}}$ corresponding to the eigenvector $\tilde{\boldsymbol{\alpha}}^i$ [19].

Projection of the mapped training pattern $\Phi(\mathbf{x}_k)$ onto the jth weighted kernel principal component can then be computed as

$$\begin{aligned} \Phi(\mathbf{x}_k)^{\mathrm{T}} \tilde{\mathbf{v}}_j &= \sum_{i=1}^{M} \hat{\alpha}_i^j \Phi(\mathbf{x}_k)^{\mathrm{T}} (w_i \Phi(\mathbf{x}_i)) \\ &= w_k^{-1} (\tilde{\mathbf{K}} \hat{\boldsymbol{\alpha}}^j)_k \\ &= w_k^{-1} (\sqrt{\tilde{\lambda}_j}\, \tilde{\boldsymbol{\alpha}}^j)_k. \end{aligned} \tag{10}$$

Similarly, for the test set $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$, the projection of $\Phi(\mathbf{y}_k)$ onto the jth principal component is

$$\Phi(\mathbf{y}_k)^{\mathrm{T}} \tilde{\mathbf{v}}_j = \sum_{i=1}^{M} \hat{\alpha}_i^j \Phi(\mathbf{y}_k)^{\mathrm{T}} (w_i \Phi(\mathbf{x}_i)). \tag{11}$$

Define the kernel matrix $\tilde{\mathbf{K}}^t \in \mathbb{R}^{N \times M}$ between the training and test patterns, where $\tilde{\mathbf{K}}^t_{ij} = \Phi(\mathbf{y}_i)^{\mathrm{T}}(w_j \Phi(\mathbf{x}_j))$. Projections in Eq. (11) can then be computed as

$$\Phi(\mathbf{y}_k)^{\mathrm{T}} \tilde{\mathbf{v}}_j = (\tilde{\mathbf{K}}^t \hat{\boldsymbol{\alpha}}^j)_k. \tag{12}$$

After the projections of the training and test data are obtained, data analysis (e.g., classification and clustering) can be performed in the much lower-dimensional embedded space.

3.2. Spectral feature analysis

Recall that at the crux of spectral clustering is the generalized eigensystem (Eq. (4)). Define $\mathbf{Y} = \mathbf{D}^{1/2}\mathbf{Z}$, then Eq. (4) can be written as

$$\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}\mathbf{Y} = \lambda \mathbf{Y}. \tag{13}$$

It is easy to see that $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ in Eq. (13) is of the same form as $\tilde{\mathbf{K}}$ in Eq. (9). To be more specific, on setting the weight $p_i = 1/d_{ii} / \sum_i 1/d_{ii}$, where d_{ii}'s are the diagonal elements of the degree matrix $\mathbf{D}$, then we have $\mathbf{P} = (1/Z)\mathbf{D}^{-1}$, where $Z = \sum_i 1/d_{ii}$ is the normalization factor. If $\mathbf{A}$ in Eq. (13) is a kernel matrix, then the corresponding weighted kernel matrix is

$$\tilde{\mathbf{K}} = \mathbf{P}^{1/2}\mathbf{A}\mathbf{P}^{1/2} = \frac{1}{Z}\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}. \tag{14}$$

Since Z is constant, the eigenvectors of $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ and $\tilde{\mathbf{K}}$ are the same. Therefore, the jth embedded coordinate[3] of the kth data point $\mathbf{Y}_{kj}$ can be obtained by solving the eigensystem

$$\tilde{\mathbf{K}}\mathbf{Y} = \lambda \mathbf{Y}. \tag{15}$$

Comparing Eq. (15) with that for WKPCA in Eq. (10), we find that they have essentially the same form. The only difference

[2] The case when the data are not centered in $\mathscr{F}$ is discussed in Appendix A.

[3] Note that $\mathbf{Y}$ can also be used as the embedded coordinates. Interested readers are referred to Ref. [16].

Table 2
Spectral feature analysis

SFA using Gaussian kernel
Input: Training and testing set $\{\mathbf{x}_i\}_{i=1}^{M}, \{\mathbf{y}_i\}_{i=1}^{N} \in \mathbb{R}^d$
Variance of the RBF kernel σ^2, final dimensionality k
Output: Projections of the training and testing set
1. Construct the kernel matrix $\mathbf{K}_{ij} = \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/2\sigma^2)$ and the degree matrix $\mathbf{D}$
2. Construct the matrix $\tilde{\mathbf{K}} = \mathbf{D}^{-1/2}\mathbf{K}\mathbf{D}^{-1/2}$
3. Perform eigendecomposition on $\tilde{\mathbf{K}}$. Compute the projections of the training data on the first k
eigenvectors corresponding to the k largest eigenvalues by Eq. (10)
4. Calculate the affinity matrix between the testing set and the training set
5. Compute the projections of the testing data set on the first k eigenvectors calculated from step 3

is that in WKPCA, the jth coordinate of the kth data point in the embedded space is scaled by the factor $w_k^{-1}\sqrt{\tilde{\lambda}_j}$. This can be understood intuitively as follows. First, the factor $\sqrt{\tilde{\lambda}_j}$ only scales the jth axis of the embedded space, and does not change the structure of the embedded data. Second, in spectral clustering, the factor $w_k^{-1} = \sqrt{d_k}$ assigns large weights to points having high degrees (i.e., those that are more similar to the other points) and small weights to low-degree points. In such a way, densities of the different clusters can be leveraged since this weighting scheme makes the high-density clusters sparser and low-density clusters more compact. This will be beneficial to the final clustering results as the different densities of the different clusters can bias the clustering algorithm [27]. Such a scaling will not change the cluster properties of the coordinates. Thus, spectral clustering can be viewed as a particular instance of WKPCA, with the weights determined only by the training set.

While the original spectral clustering can only compute spectral features for the training set, this connection between spectral clustering and WKPCA allows spectral features to be obtained for the unseen testing data. The resultant feature extraction process will be called SFA (Table 2).

The way that SFA weights each data can be understood intuitively. Since $d_{ii} = \sum_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle$, this can reflect the combined similarity of $\mathbf{x}_i$ with the rest of the data set. Moreover, $p_i = 1/d_{ii}/\sum_i 1/d_{ii} \propto 1/d_{ii}$. Hence, the more similar is $\mathbf{x}_i$ to the other points, the less important $\mathbf{x}_i$ will be. This is a common principle in data analysis and pattern recognition. For example, in the support vector machines [28], we only use those data points that are close to the class boundary (i.e., the support vectors), while points lying around the mean vector are rarely adopted.

Another interesting property of SFA is that in the normalized cuts [14], the trailing eigenvector of the normalized Laplacian matrix is not used since it has no discriminative information. This is also true in SFA. The trailing eigenvector of the normalized Laplacian just corresponds to the leading eigenvector of the (weighted) covariance matrix $\tilde{\mathbf{C}} = (1/Z)\sum_{i=1}^{M} 1/d_{ii}\Phi(\mathbf{x}_i)\Phi(\mathbf{x}_i)^{\mathrm{T}}$. The following lemma shows that this eigenvector is indeed the mean $\boldsymbol{\mu}$ of the mapped data in the feature space and thus clearly contains no discriminative information.

Proposition 1. *The scaled mean vector in the feature space is the leading eigenvector of the (weighted) covariance matrix* $\tilde{\mathbf{C}}$.

Proof. Using the fact that if $\mathbf{u}$ is an eigenvector of $\mathbf{X}'\mathbf{X}$ (where $\mathbf{X}$ is a matrix), then $\mathbf{X}\mathbf{u}$ is an eigenvector of $\mathbf{X}\mathbf{X}'$. Now, the kernel matrix $\tilde{\mathbf{K}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ corresponds to the inner product $\mathbf{X}'\mathbf{X}$ where $\mathbf{X} = [w_1\Phi(\mathbf{x}_1), \ldots, w_M\Phi(\mathbf{x}_M)] = \boldsymbol{\Phi}\mathbf{D}^{-1/2}$, where $\boldsymbol{\Phi} = [\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \ldots, \Phi(\mathbf{x}_M)]$; while $\tilde{\mathbf{C}}$ corresponds to the outer product $\mathbf{X}\mathbf{X}' = \boldsymbol{\Phi}\mathbf{D}^{-1}\boldsymbol{\Phi}^{\mathrm{T}}$. As $\mathbf{D}^{1/2}\mathbf{1}$ is the leading eigenvector[4] of $\tilde{\mathbf{K}}$ [14], the leading eigenvector of $\tilde{\mathbf{C}}$ is thus $\mathbf{X}\mathbf{1} = \boldsymbol{\Phi}\mathbf{D}^{-1/2}\mathbf{D}^{1/2}\mathbf{1} = \boldsymbol{\Phi}\mathbf{1}$, which is $M\boldsymbol{\mu}$. □

4. Analysis and discussions

In this section, we present some theoretical analysis of SFA and discuss its connections to LDA and LPP.

4.1. Connections to LDA

LDA [29] is a feature extraction method that finds projection directions which are efficient for discrimination. Given the data set $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M]$ with K classes. Let n_i be the size of class i (with $M = \sum_{i=1}^{K} n_i$), $\mathbf{x}_j^{(i)}$ the jth pattern in class i, and $\mathbf{m}^{(i)} = (1/n_i)\sum_{j=1}^{n_i}\mathbf{x}_j^{(i)}$ the mean of class i. LDA seeks the direction $\mathbf{w}$ that maximizes the criterion:

$$\max_{\mathbf{w}} \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_b\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_w\mathbf{w}}, \tag{16}$$

where

$$\mathbf{S}_b = \sum_{i=1}^{K} n_i(\mathbf{m}^{(i)} - \mathbf{m})(\mathbf{m}^{(i)} - \mathbf{m})^{\mathrm{T}} \tag{17}$$

is the between-class scatter matrix and

$$\mathbf{S}_w = \sum_{i=1}^{K}\sum_{j=1}^{n_i}(\mathbf{x}_j^{(i)} - \mathbf{m}^{(i)})(\mathbf{x}_j^{(i)} - \mathbf{m}^{(i)})^{\mathrm{T}} \tag{18}$$

is the within-class scatter matrix.

Proposition 2. *Assume that the data set* $\mathbf{X}$ *is centered, then* LDA *can be viewed as a special case of* SFA *when the kernel matrix is*

$$\mathbf{K}_{ij}^{*} = \begin{cases} 1/n_l & \textit{both } \mathbf{x}_i \textit{ and } \mathbf{x}_j \textit{ belong to class } l, \\ 0 & \textit{otherwise.} \end{cases} \tag{19}$$

[4] If $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}\mathbf{u} = \lambda\mathbf{u}$, then let $\mathbf{v} = \mathbf{D}^{-1/2}\mathbf{u}$, we get $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{v} = \lambda\mathbf{D}^{1/2}\mathbf{v}$, which can be further transformed to $\mathbf{D}^{-1}\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. Since $\mathbf{D}^{-1}\mathbf{A}$ is a *Markov* matrix, then from the *Perron–Frobenius* theorem, we have the largest eigenvalue of the matrix $\mathbf{D}^{-1}\mathbf{W}$ is 1, with its corresponding eigenvector equivalent to $\mathbf{1}$. Thus the largest eigenvalue of the matrix $\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ is 1, with its corresponding eigenvector equivalent to $\mathbf{D}^{1/2}\mathbf{1}$.

Proof. Let $\mathbf{X}^{(i)} = [\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \ldots, \mathbf{x}_{n_i}^{(i)}]$ be the data matrix of class i, and $\mathbf{e}_i = (1, 1, \ldots, 1)^{\mathrm{T}}$ an n_i-dimensional vector with all its elements equal to 1. The contribution to the within-class scatter matrix from class i is

$$\begin{aligned}\mathbf{S}_w^{(i)} &= \sum_{j=1}^{n_i}(\mathbf{x}_j^{(i)} - \mathbf{m}^{(i)})(\mathbf{x}_j^{(i)} - \mathbf{m}^{(i)})^{\mathrm{T}} \\ &= \sum_{j=1}^{n_i}\left(\mathbf{x}_j^{(i)} - \frac{1}{n_i}\mathbf{X}^{(i)}\mathbf{e}_i\right)\left(\mathbf{x}_j^{(i)} - \frac{1}{n_i}\mathbf{X}^{(i)}\mathbf{e}_i\right)^{\mathrm{T}} \\ &= \left(\mathbf{X}^{(i)} - \frac{1}{n_i}\mathbf{X}^{(i)}\mathbf{e}_i\mathbf{e}_i^{\mathrm{T}}\right)\left(\mathbf{X}^{(i)} - \frac{1}{n_i}\mathbf{X}^{(i)}\mathbf{e}_i\mathbf{e}_i^{\mathrm{T}}\right)^{\mathrm{T}} \\ &= \mathbf{X}^{(i)}\left(\mathbf{I}^{(i)} - \frac{\mathbf{e}_i\mathbf{e}_i^{\mathrm{T}}}{n_i}\right)\left(\mathbf{I} - \frac{\mathbf{e}_i\mathbf{e}_i^{\mathrm{T}}}{n_i}\right)^{\mathrm{T}}\left(\mathbf{X}^{(i)}\right)^{\mathrm{T}} \\ &= \mathbf{X}^{(i)}\left(\mathbf{I}^{(i)} - \frac{\mathbf{e}_i\mathbf{e}_i^{\mathrm{T}}}{n_i}\right)(\mathbf{X}^{(i)})^{\mathrm{T}}, \end{aligned} \tag{20}$$

where the last equality comes from the symmetry and idempotency of $\mathbf{L}^{(i)} = \mathbf{I}^{(i)} - \mathbf{e}_i\mathbf{e}_i^{\mathrm{T}}/n_i$, and $\mathbf{I}^{(i)}$ is the $n_i \times n_i$ identity matrix. Hence, the total within-class scatter matrix in Eq. (18) can be written as

$$\mathbf{S}_w = \sum_{i=1}^{K}\mathbf{S}_w^{(i)} = \sum_{i=1}^{K}\mathbf{X}^{(i)}\mathbf{L}^{(i)}(\mathbf{X}^{(i)})^{\mathrm{T}}. \tag{21}$$

Thus $\mathbf{L} = \mathbf{I} - \mathbf{K}^*$, where $\mathbf{I}$ is the $M \times M$ identity matrix. It is then easy to obtain that

$$\mathbf{S}_w = \mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}. \tag{22}$$

Define the total data scatter matrix as

$$\mathbf{S} = \sum_{i=1}^{M}(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^{\mathrm{T}}. \tag{23}$$

Then, the total between-class scatter matrix becomes [29]

$$\mathbf{S}_b = \mathbf{S} - \mathbf{S}_w = \mathbf{S} - \mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}. \tag{24}$$

LDA seeks a direction $\mathbf{w}$ corresponding to the largest eigenvalue of the generalized eigenvalue problem

$$\mathbf{S}_b\mathbf{w} = \lambda\mathbf{S}_w\mathbf{w}. \tag{25}$$

Combining Eqs. (22), (24) and (16), and assuming that the data have already been centered in the feature space (so that $\mathbf{S} = \mathbf{X}\mathbf{X}^{\mathrm{T}}$), we obtain

$$\begin{aligned}\max_{\mathbf{w}}\frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_b\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_w\mathbf{w}} &= \max_{\mathbf{w}}\frac{\mathbf{w}^{\mathrm{T}}(\mathbf{S} - \mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}})\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}\mathbf{w}} \\ &= \max_{\mathbf{w}}\frac{\mathbf{w}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}\mathbf{w}} - 1 \\ &= \max_{\mathbf{w}}\frac{\mathbf{w}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{X}\mathbf{L}\mathbf{X}^{\mathrm{T}}\mathbf{w}}. \end{aligned} \tag{26}$$

Let $\mathbf{y} = \mathbf{X}^{\mathrm{T}}\mathbf{w}$, then Eq. (26) is equivalent to

$$\max_{\mathbf{y}}\frac{\mathbf{y}^{\mathrm{T}}\mathbf{y}}{\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{y}} = \min_{\mathbf{y}}\frac{\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{y}}{\mathbf{y}^{\mathrm{T}}\mathbf{y}}, \tag{27}$$

which is the reciprocal of a standard Rayleigh quotient. By the Perron–Frobenius theorem [24], the eigenvalues of $\mathbf{L}$ lie between 0 to 2. Thus, the maximization of $(\mathbf{y}^{\mathrm{T}}\mathbf{y})/(\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{y})$ corresponds to the minimization of $(\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{y})/(\mathbf{y}^{\mathrm{T}}\mathbf{y})$. Moreover, we discard the eigenvector corresponding to the eigenvalue 0 because of the reason stated in Ref. [14] and Proposition 1. From the Rayleigh quotient theorem [24], the solution of Eq. (27) is the eigenvector corresponding to the smallest eigenvalue of $\mathbf{L}$. In other words, we have to solve the following eigenvalue problem:

$$\mathbf{L}\mathbf{y} = \beta\mathbf{y}. \tag{28}$$

Define $\gamma = 1 - \beta$, and recall that $\mathbf{L} = \mathbf{I} - \mathbf{K}^*$. Then, Eq. (28) can be rewritten as

$$\mathbf{K}^*\mathbf{y} = \gamma\mathbf{y}. \tag{29}$$

Thus, the optimal $\mathbf{y}$ is the leading eigenvector of $\mathbf{K}^*$.

Comparing Eq. (29) with Eq. (13), we find that the main difference is the weighted matrix $\mathbf{D}^{-1/2}$. However, from the definition of $\mathbf{K}^*$ (Eq. (19)) we can easily infer that the corresponding degree matrix $\mathbf{D} = \mathbf{I}$, so $\mathbf{W} = \mathbf{D}^{-1/2}$ is also an identity matrix.

Since SFA requires $\tilde{\mathbf{K}}$ in Eq. (9) to be a kernel matrix, and from the analysis above we know that under the assumption of Proposition 2, $\tilde{\mathbf{K}} = \mathbf{K}^* = \mathbf{A}$. Thus, the only remaining issue is to prove that $\mathbf{K}^*$ is a kernel. This can be seen directly from the lemma presented in Appendix B.

Therefore, comparing the projection $\mathbf{y}$ with the projection equation from Eq. (10), we can find that the only difference is the scaling factor in Eq. (10). As discussed in Section 3.2, this will not affect the cluster characteristics of the data in the projected space. Thus LDA can be viewed as a special case of SFA. □

4.2. Connections to LPP

LPP [6] is a linear feature extraction method which aims at finding a direction under which the local structure of the dataset can be optimally preserved. This direction $\mathbf{w}$ is obtained by minimizing

$$\sum_{i,j}(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i - \mathbf{w}^{\mathrm{T}}\mathbf{x}_j)^2 a_{ij}, \tag{30}$$

where a_{ij} represents the similarity between $\mathbf{x}_i$ and x_j.

We will prove in Proposition 3 that SFA is equal to *kernel* LPP, which means that SFA can preserve the nonlinear structure underlying the data set.

Proposition 3. SFA *can be viewed as a special case of* kernel LPP *when the kernel matrix has full rank.*

Proof. If the data are first mapped to a high-dimensional feature space through a nonlinear mapping Φ, then the *kernel* LPP directions in this space can be obtained by solving

$$\mathbf{\Phi}\boldsymbol{L}\mathbf{\Phi}^{\mathrm{T}}\mathbf{w} = \nu\mathbf{\Phi}\boldsymbol{D}\mathbf{\Phi}^{\mathrm{T}}\mathbf{w}, \tag{31}$$

where $\mathbf{\Phi} = [\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_M)]$, and the optimal $\mathbf{w}$'s are the eigenvectors of the above generalized eigensystem corresponding to the smallest eigenvalues. Clearly, $\mathbf{w}$ is the linear combination of $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_M)$. Assuming

$$\mathbf{w} = \sum_{i=1}^{M} \alpha_i \Phi(\mathbf{x}_i) = \mathbf{\Phi}\boldsymbol{\alpha}. \tag{32}$$

Define the kernel matrix as $\mathbf{K} = (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j))$. If we multiply $\mathbf{\Phi}^{\mathrm{T}}$ on both sides of Eq. (31), and then combine Eqs. (31) and (32), we obtain

$$\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\mathbf{L}\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\boldsymbol{\alpha} = v\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\mathbf{D}\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\boldsymbol{\alpha} \Leftrightarrow \mathbf{K}\mathbf{L}\mathbf{K}\boldsymbol{\alpha} = v\mathbf{K}\mathbf{D}\mathbf{K}\boldsymbol{\alpha}. \tag{33}$$

In general, $\mathbf{K}$ is invertible (e.g., when $\mathbf{K}$ is a Gaussian kernel [28]). Therefore, we can multiply $\mathbf{K}^{-1}$ on both sides of Eq. (33) and obtain

$$\mathbf{L}\mathbf{K}\boldsymbol{\alpha} = v\mathbf{D}\mathbf{K}\boldsymbol{\alpha}. \tag{34}$$

Recall that $\mathbf{L} = \mathbf{D} - \mathbf{A}$, then

$$\mathbf{L}\mathbf{K}\boldsymbol{\alpha} = (\mathbf{D} - \mathbf{A})\mathbf{K}\boldsymbol{\alpha} = v\mathbf{D}\mathbf{K}\boldsymbol{\alpha} \Rightarrow \mathbf{A}\mathbf{K}\boldsymbol{\alpha} = (1 - v)\mathbf{D}\mathbf{K}\boldsymbol{\alpha}. \tag{35}$$

The projections of the training data are

$$\mathbf{z} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{w} = \mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\boldsymbol{\alpha} = \mathbf{K}\boldsymbol{\alpha}. \tag{36}$$

Let $\gamma = 1 - v$, thus Eq. (35) becomes

$$\mathbf{A}\mathbf{z} = \gamma\mathbf{D}\mathbf{z}, \tag{37}$$

which is equivalent to Eq. (4), and the optimal $\mathbf{z}$'s are the eigenvectors of the above generalized eigensystem corresponding to its largest eigenvalues. Thus *kernel* LPP can be viewed as a special case of SFA. □

5. Experiments

In this section, we use the proposed SFA algorithm for face manifold analysis and face recognition. Unless otherwise stated, the Gaussian kernel $\exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2/(2\sigma^2))$ will always be used to define the similarity matrix.

5.1. Face manifold analysis

In this section, we applied SFA to the face data set used in Ref. [12]. It has 1965 images taken from a video sequence, with the pose and expression of the faces vary slowly and smoothly. Each image is of size 20×28, and with 256 gray levels. In our experiment, we randomly use 1955 images for training and the remaining 10 images for testing.

Fig. 1 shows the projections of all the images on the 2-dimensional space defined by the second and third coordinates of the spectral features. Some representative faces are also shown. As can be seen, the embedded space is roughly divided into two parts. On the left, the faces have closed mouths, while on the right, the faces have open mouths. Moreover, it can be clearly seen that the pose and expression of the faces change continuously and smoothly, from top to bottom, and from left to right. For example, the face expression changes from angry to happy from left to right. On the right, the face pouts gradually from top to bottom. A particular mode of variability in the face expression is illustrated by the image sequence inside the lower green ellipse, which corresponds to points along the red path (linked by red solid line) in the upper green ellipse.

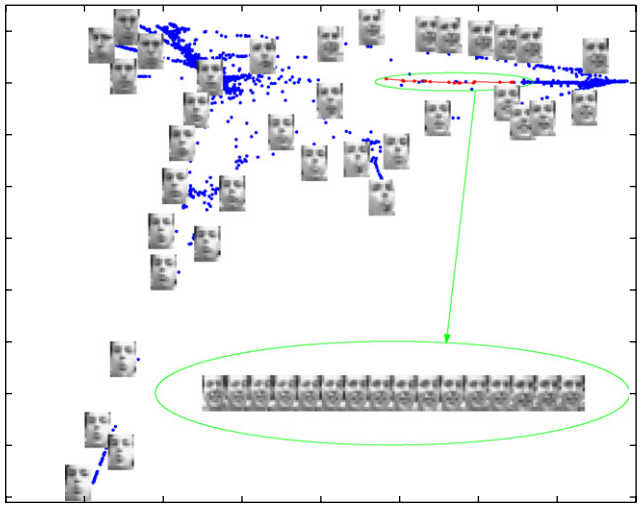

Fig. 1. The 2-dimensional embedding of face images obtained by spectral feature analysis. The embedded space is roughly divided into two parts. On the left, the faces have closed mouths, while on the right, the faces have open mouths. Moreover, the pose and expression of the faces change continuously and smoothly, from top to bottom, and from left to right. For example, the face expression changes from angry to happy from left to right. On the right, the face pouts gradually from top to bottom. A particular mode of variability in the face expression is illustrated by the image sequence inside the lower ellipse, which corresponds to points along the path (linked by a solid line) in the upper ellipse.

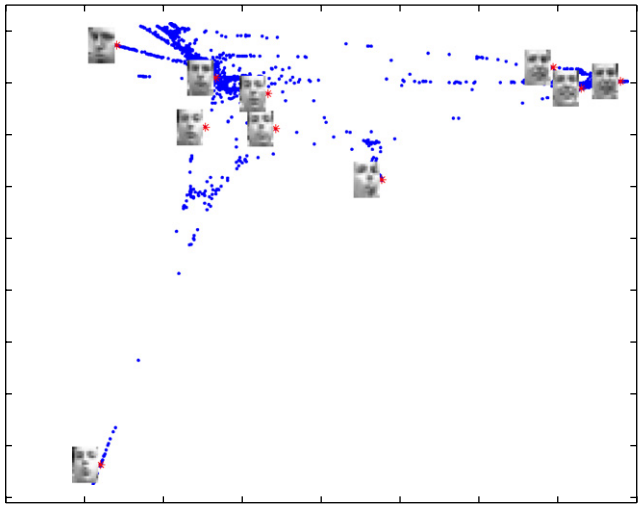

Fig. 2. Embeddings of the 10 test images obtained by spectral feature analysis. Comparing with Fig.1, we can easily see that these testing samples can find their optimal coordinates which reflect their intrinsic properties, i.e., pose and expression.

The 10 test samples can be easily located in this lower-dimensional space by SFA, which are shown in red in Fig. 2.

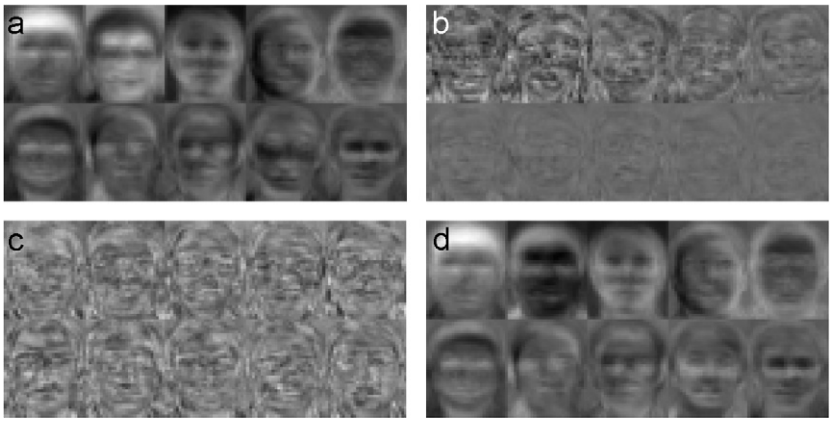

Fig. 3. "Faces" obtained by the different methods. (a) Eigenfaces. (b) Fisherfaces. (c) Laplacianfaces. (d) Spectralfaces.

As can be seen, these testing samples find their optimal coordinates which reflect their intrinsic properties, i.e., pose and expression. This shows that SFA is able to capture the intrinsic face manifold structure.

5.2. Face representation using linear SFA

Since SFA is a kernel-based method, the data lie in a very high-dimensional feature space after the Φ-mapping. Moreover, usually we do not know the explicit form of Φ. However, if we use the linear kernel [28], then SFA is just a special case of *weighted* PCA. Thus, we can show the eigenvectors $\tilde{\mathbf{v}}$ of the weighted covariance matrix $\tilde{\mathbf{C}}$ as ordinary images. We will call these images *spectralfaces*. To be consistent with the nomenclature of these standard methods, we will also call SFA the *spectralface* method in the rest of this paper. As an illustration, we use the ORL data set [30] as our training set and show its first 10 *eigenfaces*, *Fisherfaces*, *Laplacianfaces* and *spectralfaces* in Fig. 3. We can see that the spectralfaces are very similar to the eigenfaces, since with the linear kernel SFA is just a special version of WPCA.

5.3. Face recognition

In this section, we will investigate the performance of our proposed *Spectralface* method for face recognition on three benchmark databases. The results of the following standard face recognizers are also provided for comparison: (1) *Eigenface* (PCA) [4]; (2) *Fisherface* (LDA) [5]; and (3) *Laplacianface* (LPP) [6].

5.3.1. ORL face database

In this experiment, we perform face recognition by using the ORL face database [30]. There are 10 images for each of the 40 human subjects. For some subjects, the images were taken at different times, with varying lighting, facial expressions (open/closed eyes, smiling/not smiling) and facial details (glasses/no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). The original images (with 256 gray levels) have size 92×112, which are downsampled to 23×28 for efficiency. Fig. 4 shows the 10 images of one particular subject.

Fig. 4. Sample images from the ORL face database.

We performed experiments with different numbers of training samples. As each subject has 10 images, k of them were randomly selected for training and the remaining $10-k$ were used for testing. The dimensionality of the projected space was set to 10. The nearest-neighbor classifier, with the Euclidean metric, was adopted in the projected space. The variance of the Gaussian function (used in the *spectralface* and *Laplacianface* methods) was adjusted for best performance. For each value of k, 50 independent runs were performed. Fig. 5 shows the average recognition accuracies. Notice that SFA achieves higher accuracy than the other methods.

Next, we experimented with different dimensionalities of the projected space. A random subset with six images per subject (and thus 240 images in total) was used for training, and the remaining images were used for testing. Note that for the *Fisherface* method, there are at most $K-1$ nonzero generalized eigenvalues (where K is the number of subjects) [5], and thus the dimensionality of the projected space will be less than K. The recognition results are shown in Fig. 6, where the accuracy values were averaged over 50 random splits. Table 3 shows the best performance for each method and the corresponding dimensionalities of the projected spaces. Again, *spectralface* clearly outperforms the others.

5.3.2. UMIST face database

The UMIST face database [31] consists of 564 images of 20 people. The number of images of each subject vary from 19

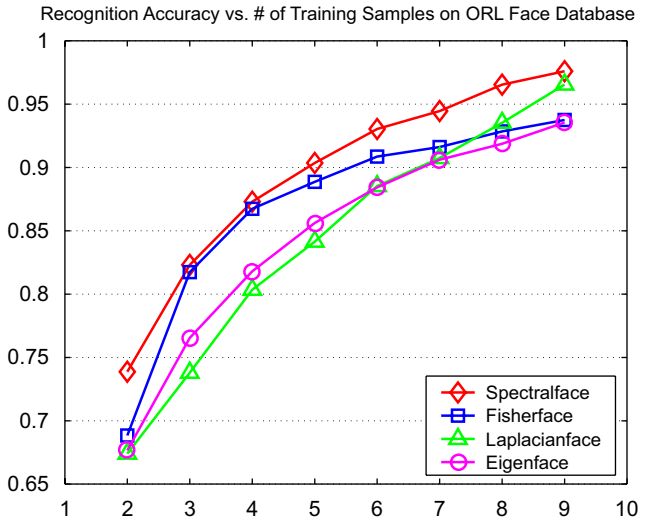

Fig. 5. Recognition accuracies on the ORL face database. The abscissa is the number of training images per subject and the ordinate is the average recognition accuracy.

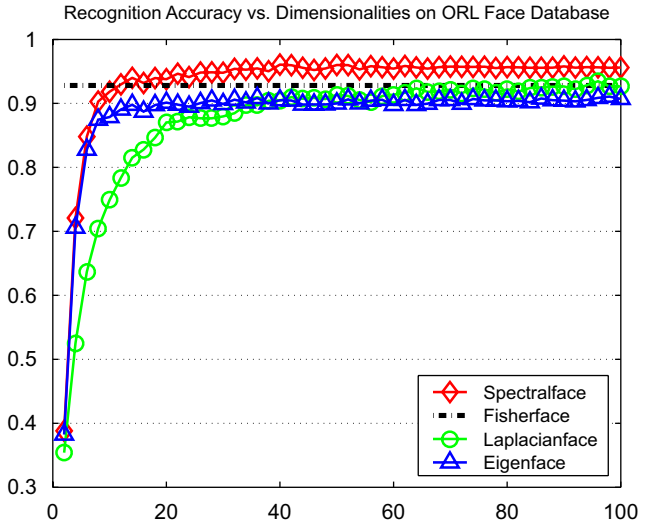

Fig. 6. Average recognition accuracy on the ORL face database with different dimensionalities of the projected space. The abscissa is the dimensionality and the ordinate is the average recognition accuracy.

Table 3
Best average recognition accuracy on the ORL face database

	Dimensionality	Accuracy (%)
Eigenface	98	91.06
Fisherface	39	92.81
Laplacianface	96	93.47
Spectralface	40	96.07

to 36. The subjects cover a range of race/sex/appearance. Each image has 256 shades of gray, and is of size 92 × 112, which is again down-sampled to 23 × 28 for efficiency. Fig. 7 shows the 10 images of a particular subject.

Fig. 7. Sample images from the UMIST face database.

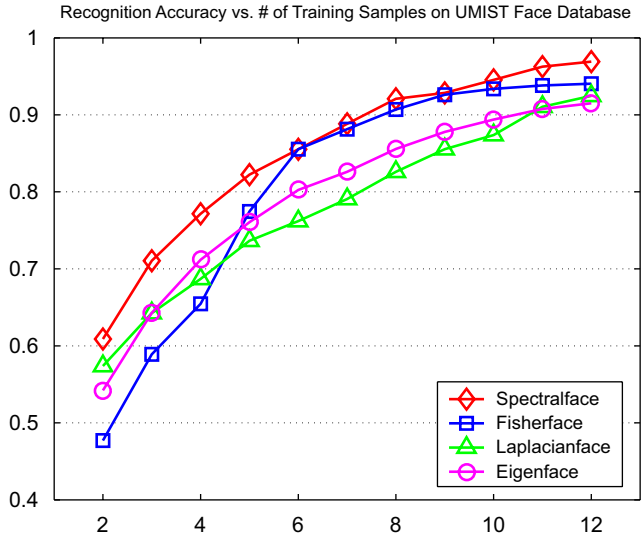

Fig. 8. Average recognition accuracies on the UMIST face database. The abscissa is the number of training images per subject and the ordinate is the average recognition accuracy.

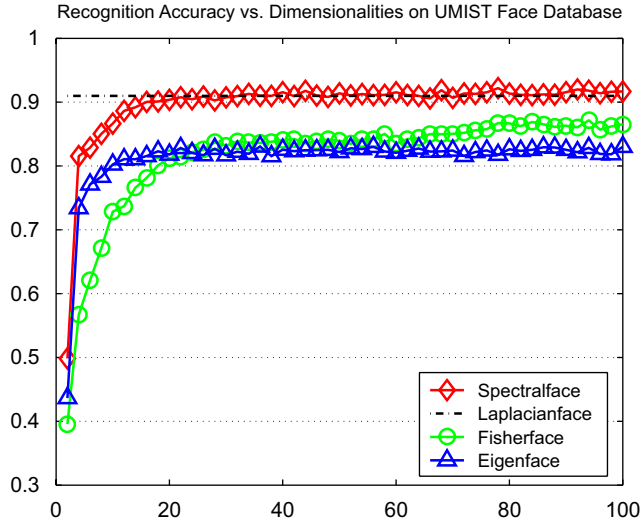

Fig. 9. Average recognition accuracies on the UMIST face database with different dimensionalities of the projected space. The abscissa is the dimensionality and the ordinate is the average recognition accuracy.

From each subject, we randomly select k images for training and the remaining for testing. The dimensionality of the projected space is set to 10. The other experimental setups are the same as those in Section 5.3.1. Results are shown in Fig. 8. Clearly, the *spectralface* method is superior.

Table 4
Best average recognition accuracy on the UMIST face database

	Dimensionality	Accuracy (%)
Eigenface	36	83.05
Fisherface	17	90.98
Laplacianface	94	87.14
Spectralface	78	92.19

Fig. 10. Sample images from the Yale face database.

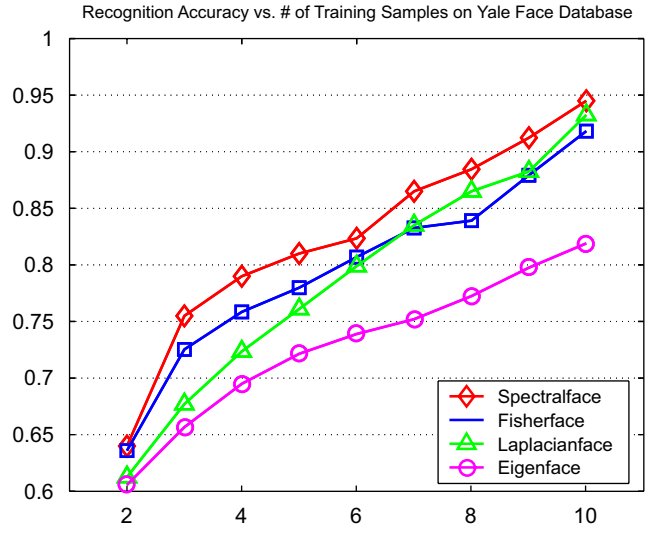

Fig. 11. Average recognition accuracies on the Yale face database. The abscissa is the number of training images per subject and the ordinate is the average recognition accuracy.

Next, we experimented with different dimensionalities of the projected space. Six face images per subject were randomly selected for training, and the remaining were used for testing. Fig. 9 shows the results averaged over 50 trials. Finally, Table 4 shows the best performance for each method and the corresponding dimensionalities of the projected spaces. Again, our *spectralface* method is clearly the best.

5.3.3. Yale database

The Yale face database [32] contains 11 grayscale images for each of the 15 individuals. The images demonstrate variations in lighting condition (left-light, center-light, right-light), facial expression (normal, happy, sad, sleepy, surprised, and wink), and with/without glasses. In our experiment, the images were also resized to 23×28. Fig. 10 shows 10 images of a particular object.

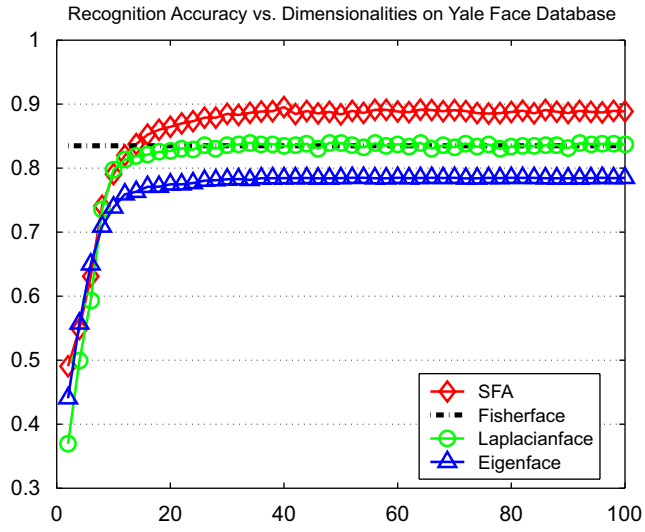

Fig. 12. Average recognition accuracies on the Yale face database with different dimensionalities of the projected space. The abscissa is the dimensionality and the ordinate is the average recognition accuracy.

Table 5
Best recognition accuracy on the Yale face database

	Dimensionality	Accuracy (%)
Eigenface	35	78.56
Fisherface	14	83.22
Laplacianface	38	85.04
spectralface	40	89.26

For each individual, a random subset of k images was used for training, and the rest were used for testing. The other experimental setups are the same as those in Sections 5.3.1 and 5.3.2. Fig. 11shows the results, and the *spectralface* also performs best. Next, we experimented with different dimensionalities of the projected spaces, with the size of the training set fixed to 90, and with six images per subject. The average recognition accuracies over 50 independent runs are shown in Fig. 12. The best performance and the corresponding dimensionalities of the projected spaces for each method are shown in Table 5. Clearly, our method is more advantageous.

6. Conclusion and discussion

Inspired by spectral clustering, we introduce in this paper a novel feature extraction method called *spectral feature analysis* (SFA). Theoretical analysis shows that it has many appealing properties including the immunity from the small-sample-size problem, and the abilities of extracting discriminative information and exploiting the nonlinear manifold structure hidden in the data set. Experimental results on face manifold analysis and face recognition showed the effectiveness of our method.

Despite these advantages, there are still some problems to be solved. For example, the weighting factor $w_i = 1/\sqrt{d_{ii}}$ used in Section 3.2 seems a little unreasonable. Since $d_{ii} = \sum_j \mathbf{A}_{ij}$ is the total similarity between pattern i and the remaining patterns,

so the further is pattern i away from the other patterns, the smaller is d_{ii}, and consequently the larger is w_i. This may make SFA sensitive to outliers. Therefore, how to make SFA more robust and how to make the weights more reasonable is an important future direction. Moreover, automatic tuning of the parameters in SFA (e.g., the variance of the Gaussian kernel) is also a challenging and interesting task since the tuning of the parameters in kernel-based methods is still a hard problem in machine learning research.

Acknowledgments

The authors would like to thank the comments of the anonymous editors and reviewers. The work is supported by Chinese Natural Science Foundation (60675009).

Appendix A.

In Section 3.1, we assume that all the mapped data are centered in the feature space. Now we will drop this assumption. Generally speaking, the sample mean of a data set is the best representative in the sense that the total squared distance between all the data objects and the mean is the smallest. More precisely, if $\boldsymbol{\mu} = (1/M)\sum_{i=1}^{M}\mathbf{x}_i$, then $\boldsymbol{\mu} = \arg\ \min_{\varepsilon}\sum_{i=1}^{M} \|\mathbf{x}_i - \varepsilon\|^2$. Extending this idea, we can define the sample mean in the weighted case as

$$\mathbf{m} = \frac{\sum_{i=1}^{M} p_i\mathbf{x}_i}{\sum_{i=1}^{M} p_i} = \arg\min_{\boldsymbol{\epsilon}} \sum_{i=1}^{M} p_i\|\mathbf{x}_i - \boldsymbol{\epsilon}\|^2. \tag{38}$$

It is easy to see that if we set all the weights of the data objects to be one, then $\mathbf{m} = \boldsymbol{\mu}$. According to Eq. (38), since we assume $\sum_i p_i = 1$, we can define the weighted mean of the data objects in the feature space as $\mathbf{m} = \sum_{i=1}^{M} p_i\Phi(\mathbf{x}_i)$. Let $\mathbf{P} = \mathrm{diag}(p_1, p_2, \ldots, p_M) = \mathbf{W}^2$, then $\mathbf{m} = \boldsymbol{\Phi}\mathbf{P}\mathbf{e}$, where $\boldsymbol{\Phi} = [\Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2), \ldots, \Phi(\mathbf{x}_M)]$ and $\mathbf{e}$ is the M-dimensional column vector with all ones. Let the centered weighted kernel matrix be $\tilde{\mathbf{A}}^c$, then

$$\begin{aligned}\tilde{\mathbf{A}}^c &= ((\boldsymbol{\Phi} - \mathbf{m}\mathbf{e}^{\mathrm{T}})\mathbf{W})^{\mathrm{T}}((\boldsymbol{\Phi} - \mathbf{m}\mathbf{e}^{\mathrm{T}})\mathbf{W})\\ &= (\boldsymbol{\Phi}\mathbf{W} - \boldsymbol{\Phi}\mathbf{P}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W})^{\mathrm{T}}(\boldsymbol{\Phi}\mathbf{W} - \boldsymbol{\Phi}\mathbf{P}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W})\\ &= (\boldsymbol{\Phi}\mathbf{W}(\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W}))^{\mathrm{T}}(\boldsymbol{\Phi}\mathbf{W}(\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W}))\\ &= (\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W})(\boldsymbol{\Phi}\mathbf{W})^{\mathrm{T}}(\boldsymbol{\Phi}\mathbf{W})(\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W})\\ &= (\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W})\tilde{\mathbf{A}}(\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W}).\end{aligned} \tag{39}$$

For the test set $\boldsymbol{\Phi}_t = [\Phi(\mathbf{y}_1), \Phi(\mathbf{y}_2), \ldots, \Phi(\mathbf{y}_N)]$, the centered matrix of $\tilde{\mathbf{A}}^t$ is

$$\begin{aligned}\tilde{\mathbf{A}}^{tc} &= (\boldsymbol{\Phi}_t - \mathbf{m}\mathbf{e}_t^{\mathrm{T}})^{\mathrm{T}}((\boldsymbol{\Phi} - \mathbf{m}\mathbf{e}^{\mathrm{T}})\mathbf{W})\\ &= (\boldsymbol{\Phi}_t - \boldsymbol{\Phi}\mathbf{P}\mathbf{e}\mathbf{e}_t^{\mathrm{T}})^{\mathrm{T}}(\boldsymbol{\Phi}\mathbf{W} - \boldsymbol{\Phi}\mathbf{P}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W})\\ &= \tilde{\mathbf{A}}_t(\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W}) - \mathbf{e}_t\mathbf{e}^{\mathrm{T}}\mathbf{W}\tilde{\mathbf{A}}(\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W})\\ &= (\tilde{\mathbf{K}}_t - \mathbf{e}_t\mathbf{e}^{\mathrm{T}}\mathbf{W}\tilde{\mathbf{A}})(\mathbf{I} - \mathbf{W}\mathbf{e}\mathbf{e}^{\mathrm{T}}\mathbf{W}),\end{aligned} \tag{40}$$

where $\mathbf{e}_t = [1, 1, \ldots, 1]^{\mathrm{T}}_{1\times N}$, and N is the number of testing data.

Assume that $\tilde{\mathbf{v}}_j$ is the jth eigenvector of the covariance matrix

$$\tilde{\mathbf{C}}' = \sum_{i=1}^{M} p_i(\Phi(\mathbf{x}_i) - \mathbf{m})(\Phi(\mathbf{x}_i) - \mathbf{m})^{\mathrm{T}}, \tag{41}$$

then we can calculate the projections of the training and testing data as

$$\begin{aligned}(\Phi(\mathbf{x}_k)\cdot\tilde{\mathbf{v}}_j) &= \sum_{i=1}^{M}\hat{\gamma}_i^j(\Phi(\mathbf{x}_k)\cdot w_i\Phi(\mathbf{x}_i))\\ &= w_k^{-1}(\tilde{\mathbf{A}}^c\hat{\boldsymbol{\gamma}}^j)_k,\end{aligned} \tag{42}$$

$$\begin{aligned}(\Phi(\mathbf{y}_k)\cdot\tilde{\mathbf{v}}_j) &= \sum_{i=1}^{M}\hat{\gamma}_i^j(\Phi(\mathbf{y}_k)\cdot w_i\Phi(\mathbf{x}_i))\\ &= (\tilde{\mathbf{A}}^{tc}\hat{\boldsymbol{\gamma}}^j)_k,\end{aligned} \tag{43}$$

where $\hat{\boldsymbol{\gamma}}^i = \tilde{\boldsymbol{\gamma}}^i/\sqrt{\delta_i}$, and $(\delta_i, \tilde{\boldsymbol{\gamma}}^i)$ is the eigenvalue–eigenvector pair of $\tilde{\mathbf{K}}^c$.

Appendix B.

Lemma 1. $\mathbf{K}^*$ (*defined by Eq.* (19)) *is a kernel.*

Proof. Define the M-dimensional indication vector of class i as

$$\mathbf{q}_j^{(i)} = \begin{cases}1 & j \in \text{class } i,\\ 0 & \text{otherwise.}\end{cases} \tag{44}$$

then $\mathbf{K}^* = \sum_{i=1}^{K}(1/n_i)\mathbf{q}^{(i)}(\mathbf{q}^{(i)})^{\mathrm{T}}$. Let $\mathbf{K}_i^* = (1/n_i)\mathbf{q}^{(i)}(\mathbf{q}^{(i)})^{\mathrm{T}}$, then

$$\begin{aligned}(\mathbf{K}^*)^2 &= \frac{\mathbf{q}^{(i)}(\mathbf{q}^{(i)})^{\mathrm{T}}}{n_i}\cdot\frac{\mathbf{q}^{(i)}(\mathbf{q}^{(i)})^{T}}{n_i}\\ &= \frac{\mathbf{q}^{(i)}(\mathbf{q}^{(i)})^{\mathrm{T}}}{n_i}\\ &= \mathbf{K}^*.\end{aligned} \tag{45}$$

Thus, $\mathbf{K}_i^*$ is idempotent and its eigenvalues are either 0 or 1 [24]. In addition, $\mathbf{K}^*$ is symmetric, and therefore it is positive semidefinite (Example 2.4 in Ref. [28]). Let $\mathbf{S}_+^M$ be the set of all $M \times M$ symmetric positive semidefinite kernels, then $\mathbf{S}_+^M$ is a convex cone [33]. So, $\mathbf{K}^*$ is positive semidefinite. From Definition 2.5 of Ref. [28], we then know that $\mathbf{K}$ is a kernel. □

References

[1] R. Brunelli, T. Poggio, Face recognition: features versus templates, IEEE Trans. Pattern Anal. Mach. Intell. 15 (10) (1993) 1042–1052.
[2] F.S. Samaria. Face recognition using hidden Markov models, Ph.D. Thesis, University of Cambridge, 1994.
[3] L. Wiskott, J. Fellous, N. Kruger, C. Malsburg, Face recognition by elastic bunch graph matching, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 775–779.
[4] M.A. Turk, A.P. Pentland, Eigenfaces for recognition, J. Cognitive Neurosci. 3 (1) (1991) 71–96.
[5] P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, Eigenface vs. Fisherface: recognition using class specific linear projection, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 711–720.

[6] X. He, S. Yan, Y. Hu, P. Niyogi, H. Zhang, Face recognition using Laplacianfaces, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 328–340.
[7] Y. Adini, Y. Moses, S. Ullman, Face recognition: the problem of compensating for changes in illumination direction, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 721–732.
[8] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Oxford, 1995.
[9] X. Wang, X. Tang, Random sampling LDA for face recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2004.
[10] A.M. Martínez, A.C. Kak, PCA versus LDA, IEEE Trans. Pattern Anal. Mach. Intell. 23 (2) (2001) 228–233.
[11] Y. Chang, C. Hu, M. Turk, Manifold of facial expression, in: Proceedings of the IEEE International Workshop on Analysis and Modeling of Faces and Gestures, 2003.
[12] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science (2000) 290.
[13] H.S. Seung, D.D. Lee, The manifold ways of perception, Science (2000) 290.
[14] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[15] Y. Weiss, Segmentation using eigenvectors: a unifying view, in: Proceedings of the IEEE International Conference on Computer Vision, 1999, pp. 975–982.
[16] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, Adv. Neural Inf. Process. Syst. (2002) 14.
[17] F. Chung, Spectral Graph Theory, in: CMBS Regional Conference Series in Mathematics, vol. 92, American Mathematical Society, Providence, RI, 1997.
[18] C. Alpert, A. Kahng, S. Yao, Spectral partitioning: the more eigenvectors, the better, Discrete Appl. Math. 90 (1999) 3–26.
[19] B. Schölkopf, A. Smola, K. Müler, Nonlinear component analysis as a kernel eigenvalue problem, Neural Comput. 10 (1998) 1299–1319.
[20] H. Chang, D.-Y. Yeung, Robust locally linear embedding, Pattern Recognition 39 (2006) 1053–1065.
[21] B. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, The Bell Syst. Tech. J. 49 (2) (1970) 291–307.
[22] P. Chan, M. Schlag, J. Zien, Spectral k-way ratio cut partitioning, IEEE Trans. CAD-Integrated Circuits Syst. 13 (1994) 1088–1096.
[23] S.X. Yu, J. Shi, Multiclass spectral clustering, in: Proceedings of IEEE International Conference on Computer Vision, 2003.
[24] G.H. Golub, C.F. Van Loan, Matrix Computation, second ed., Baltimore, 1989.
[25] S. Kamvar, D. Klein, C. Manning, Spectral learning, in: Proceedings of International Joint Conference on Artificial Intelligence, 2003.
[26] I.T. Jolliffe, Principal Component Analysis, second ed., Springer, New York, 2002.
[27] D. Zhou, B. Schölkopf, Learning from labeled and unlabeled data using random walks, in: Proceedings of 26th DAGM Symposium, C.E. Rasmussen, H.H. Bülthoff, M.A. Giese, B. Schölkopf, (Eds.) Springer, Berlin, Germany, 2004, pp. 237–244.
[28] B. Schölkopf, A. Smola, Learning with kernels, The MIT Press, Cambridge, MA, London, England, 2002.
[29] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley, New York, 2001.
[30] F. Samaria, A. Harter, Parameterisation of a stochastic model for human face identification, in: Second IEEE Workshop on Applications of Computer Vision, 1994.
[31] D.B. Graham, N.M. Allinson. Characterizing virtual eigensignatures for general purpose face recognition, in: H. Wechsler, P.J. Phillips, V. Bruce, F. Fogelman-Soulie, T.S. Huang (Eds.), Face Recognition: From Theory to Applications, NATO ASI Series F, Computer and Systems Sciences, vol. 163, 1998, pp. 446–456.
[32] Yale University Face Database, ⟨http://cvc.yale.edu/projects/yalefaces/yalefaces.html⟩, 2002.
[33] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, 2004.

About the Author—FEI WANG is a PhD candidate of grade four in Department of Automation, Tsinghua University, Beijing, PR China. He has now published several papers in top conferences of machine learning and pattern recognition, such as CVPR and ICML.

About the Author—JINGDONG WANG is a PhD candidate in Department of Computer Science, Hong Kong University of Science and Technology.

About the Author—CHANGSHUI ZHANG is a professor in Department of Automation, Tsinghua University, Beijing, PR China. He is an associated editor of Pattern Recognition Journal.

About the Author—JAMES KWOK is an associate professor in Department of Computer Science, Hong Kong University of Science and Technology.