Linear and Nonlinear Dimensionality Reduction Techniques

Ioan Buciu
Department of Electronics
Faculty of Electrical Engineering and Information Technology
University of Oradea, Romania
E-mail: ibuciu@uoradea.ro

Ioan Naforniță
Electronics and Communications Faculty
Politehnica University of Timisoara
300223 Timisoara, Romania
Bd. Vasile Parvan, nr.2
E-mail: ioan.nafornita@etc.upt.ro

Abstract: This overview aims at describing the most representative existing standard as well as new linear and nonlinear data dimensionality reduction methods. Finding new dimensionality reduction algorithms for preserving as much as possible the underlying structure of data are still challenging topics from the computer vision field. The paper presents the most classical and advanced techniques used for data dimensionality reduction.

Keywords: dimensionality reduction, image processing, computer vision, pattern recognition, artificial intelligence.

Ioan Buciu received the Diploma of Electrical Engineering in 1996 and the Master of Science degree in microwaves in 1997, both from the University of Oradea. From 1997 to 2000, he served as a Teaching Assistant with the Department of Applied Electronics, University of Oradea. During 2001 - 2005, he was a researcher and a PhD student at the Artificial Intelligence and Information Analysis Lab, Department of Informatics, Aristotle University of Thessaloniki, Greece. Currently he is with the Department of Applied Electronics, Electrical Engineering and Information Technology, University of Oradea. His current research interests lie in the areas of signal, image processing, pattern recognition, machine learning, and artificial intelligence. Also, his area of expertise includes face analysis, support vector machines, and image representation and decomposition.

Ioan Naforniță (M'68) received his BS, MEE, and PhD in electronics in 1965, 1968, and 1981 respectively, from "Politehnica" University, Timisoara, Romania. He is currently a Professor, leading the Communications Department of the Electronics and Telecommunications Faculty at the "Politehnica" University, Timisoara, Romania. He is also working as a PhD advisor in Engineering and Telecommunications. He is a Correspondent Member of the Romanian Academy of Technical Sciences, section 5, Information Technology and Communications "Computers and Communications" since 2001. His current research interests are in the area of statistical signal processing and time-frequency representations.

1. Introduction

The issue of dimensionality reduction concerns plenty of researchers from various fields, such as image processing, machine learning, data compression, computer vision, artificial intelligence and pattern recognition or even neuroscience. Therefore, not surprisingly, these topics knew a growing interest in the last few years taking into account its interdisciplinary character. One of the main reasons for performing dimensionality reduction is provided by the so-called "curse of dimensionality". As many times the final goal of an image processing task is to classify the samples from the data as pertaining to some certain classes invoking pattern recognition approaches, avoiding the curse of dimensionality is a necessary step. It is well known and widely accepted that many classifiers perform poorly in a high dimensional space with a limited number of samples. The classification methods emerging from statistics cannot be accurately modeled when the amount of available samples is small compared with its dimension, as usually occurs, for instance, in image processing and classification. Another reason for carrying out this step is provided by the huge computational time required for high dimensional data processing. Even if the classification process leads to satisfactory results, the procedure can be prohibitively time consuming. Dimensionality reduction is a research topic from both the computer vision and neuroscience fields. To have a simple example of the importance of dimensionality reduction let us consider we have an image of an acceptable quality of 256x256 pixels leading to a large dimension of 65536 pixels. It is highly unlikely that the image does not contain redundant information. As consequence, it is commonly accepted that the intrinsic dimensionality of the image space is much lower than the original image space.

1.1. General Model

The problem can be stated as follows. Let us assume we have a p - dimensional manifold M embedded in a m - dimensional space, with $p < m$. Suppose further that we have a differentiable

function f having the rank p. Then, the function $f : R^p \rightarrow \Re^m$ is called an *embedding*. Usually $\Re^m$ forms the input (data) space and R^p refers to the feature (or latent) space. Given a set of *n* observations (realizations) $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ of the m - dimensional random vector $\mathbf{x} = (x_1, x_2, ..., x_m)^T$, the *dimensionality reduction* refers to the estimation of the unknown lower (intrinsic) p - dimensional vector $\mathbf{y}_i$, $i = 1, ..., n$ such that $\mathbf{x}_i = f(\mathbf{y}_i) + \varepsilon_i$, with ε denoting the noise. The vector p is often known as *latent* or *hidden* variable (component). The sample mean and the $m \times m$ sample covariance matrix of the variable $\mathbf{x}$ are denoted by:

$$\mu = (\mu_1, \mu_2, ..., \mu_m)^T, \tag{1}$$

and

$$\Sigma = E\{(\mathbf{x} - \mu)(\mathbf{x} - \mu^T)\}, \tag{2}$$

respectively, where E denotes the expectation. Assuming that the data are stored in the columns of the $m \times n$ matrix $\mathbf{X}$, the dimensionality reduction techniques attempt to decompose the data into a $m \times p$ matrix $\mathbf{B}$ where each column of the input data $\mathbf{X}$ is a linear (or nonlinear) combination of the elements in $\mathbf{B}$, such that

$$\mathbf{X} = \mathbf{BY}, \tag{3}$$

where the matrix $\mathbf{Y}$ of size $p \times n$ convey in its columns the new n p - dimensional data variables (vectors).

The linear dimensionality reduction techniques can only retrieve the linear structure of the subspace. In this case, each latent component y_i is a linear combination of original vector x_i:

$$y_{ji} = \sum_{o=1}^{m} a_{jo} x_{oi} \tag{4}$$

with $i = 1, ..., n$, $j = 1, ..., p$, and $o = 1, ..., m$. In a matrix form this is written:

$$\mathbf{Y} = \mathbf{AX}, \tag{5}$$

where $\mathbf{A}$ is the linear transformation weight matrix. Obviously, $\mathbf{A} = \mathbf{B}^{-1}$. However, in many cases, the assumption that the input space can be represented as linear combination of the feature subspaces does not always account for expected results from the real-world scenarios. For instance, translating an object on a uniform background cannot be represented as a linear function of the image pixels. Therefore, undertaken a nonlinear decomposition of the input space can lead to more appropriate subspace representation.

The paper reviews current state-of-the art linear as well as nonlinear dimensionality reduction approaches. Linear approaches described in this overview include Principal Component Analysis, Linear Discriminant Analysis, and Independent Component Analysis. New techniques that allow only non-negative values in their decomposition factors such as Non-negative Matrix Factorization, Local Non-negative Matrix Factorization and Discriminant Non-negative Matrix Factorization method are also discussed. The nonlinear approaches will be represented by the kernel-based techniques such as the Kernel Principal Component Analysis, Kernel Linear Discriminant Analysis and Kernel Independent Component Analysis as nonlinear extension for their linear parts.

2. Linear Dimensionality Reduction

2.1. Principal Component Analysis

One of the oldest linear dimensionality techniques is the well known *Principal Component Analysis* (PCA) [1]. Its simplicity and large area of applications ranked the PCA (also know as Karhunen-Loeve transform from the communication theory) as one of the most popular approaches, despite its shortcomings. PCA is a technique that linearly transforms the data by finding a set of p orthogonal axes (principal components or eigenvectors) that accounts for the maximum data's variance. The first component points out to the direction with the maximum variance. From relations (3) and (5) and, since the principal components (PCs) are orthogonal (i.e., $\mathbf{B}^{-1} = \mathbf{B}^T$), the latent variable $\mathbf{y}_1$ is found by projecting the original variable onto the weight matrix, $\mathbf{y}_1 = \mathbf{B}^T(\mathbf{x}_1 - \mu)$ such that:

$$\mathbf{b} = \arg\max_{\|b\|=1} E\{(\mathbf{b}^T\mathbf{x})^2\} \tag{6}$$

The PCs are comprised in the columns of the matrix $\mathbf{B}$. The second component lies in the subspace perpendicular to the first, the third component goes along with the maximum variance direction in the subspace perpendicular to the first two, and so on. The orthogonality principle between the eigenvectors is the only constraint imposed in expressing the eigenvectors. The eigenvectors $\mathbf{b}_j$ (PCs) and the corresponding eigenvalues λ_j can be computed by solving the equation:

$$\Sigma\mathbf{b}_j = \lambda_j\mathbf{b}_j \, j = 1,..., p, \tag{7}$$

or equivalently, the following characteristic equation:

$$\left|\Sigma - \lambda\mathbf{I}\right| = 0 \tag{8}$$

where $\mathbf{I}$ is the identity matrix and the $|.|$ denotes the determinant of the matrix. By ordering the eigenvectors in the order of descending eigenvalues (largest first), one can create an ordered orthogonal basis with the first eigenvector having the direction of largest variance of the data. In this way, we can find directions in which the data set has the most significant amounts of energy. PCA optimally minimizes reconstruction error under the L_2 norm (Euclidean distance) as cost function:

$$\sum\left\|\mathbf{X} - \mathbf{BY}\right\|^2 \tag{9}$$

Once the eigenvectors are rearranged according to the decreasing value of their eigenvalues, an issue is given by the number p of eigenvectors (PCs) to be retained. The number of PCs to be kept depends on the application. A common way to select an appropriate number of PCs is provided by computing the cumulative energy proportion of the variance comprised by the first p PCs:

$$\sum_{j=1}^{p}\lambda_j / \text{trace}(\Sigma) \tag{10}$$

Next, the number of PCs is selected so that the cumulative energy is above a certain threshold, for instance, 95 %. The rest of the eigenvectors having low energy are discarded.

An impressive number of applications exist for PCA from different scientific fields such as Statistics, Computer Vision, Image Processing, Pattern Recognition, Chemistry, Astronomy, etc. Here we present PCA as graphical representation for a known database called the Iris database [2] containing 3 classes of 50 instances each, where each class refers to a type of iris plant. Each plant is described by four attributes:

sepal length, sepal width, petal length and petal width. Therefore, each plant can be described as 5-dimensional vector, i.e., 4 attributes plus one for the label that encodes the classes. One class is linearly separable from the other 2; the latter are not linearly separable from each other. The classes are coded as follows: Iris Setosa - 0, Iris Versicolour - 1, and Iris Virginica - 2. PCA can be performed to reduce the dimensionality from 4 to 2 in order to visualize the data. Figure 1 depicts the graphical representation of the projection of the Iris database on the two first principal components.

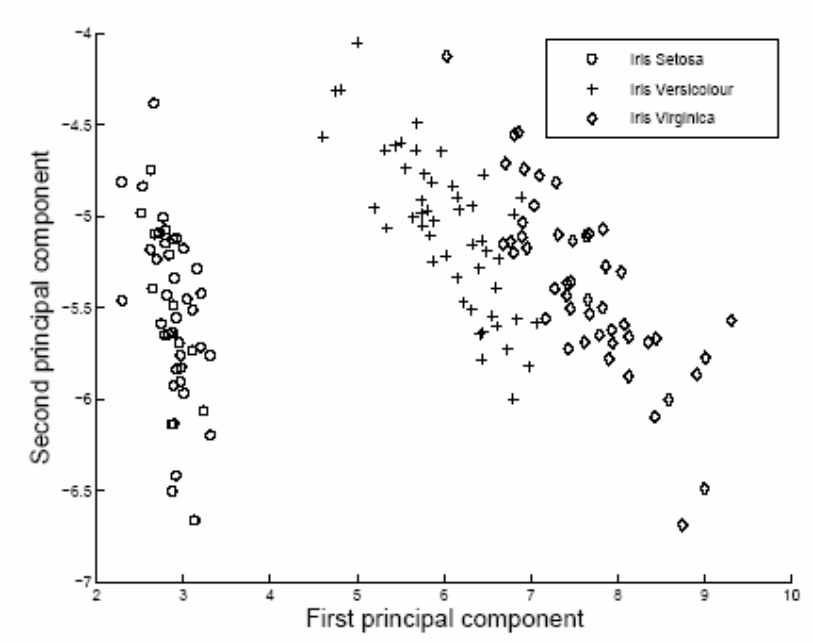

Figure 1. Graphical representation of the projection of the Iris database on the two first principal components.

The PCA projection reveals the structure of the original data described by the three formed clusters corresponding to the three iris plant types.

2.2. Linear Discriminant Analysis

Let us suppose now that the input data are formed by samples that are annotated according to some certain labels (classes). While the most techniques for dimensionality reduction are based on unsupervised approach (i.e. the cost function does not take into account the class information), several approaches to incorporate class information exist. Closely related to PCA, and justified by the fact that, when it comes to classification task, the principal components do not necessary convey discriminant information (since PCA is not optimized for classification), a class-specific linear projection for dimensionality reduction is provided by the *Linear Discriminant Analysis* (LDA) or Fisher's linear discriminant (FDA) [3]. Let us suppose now that we have Q distinctive classes and let n_c be the number of samples in class $Q, c = 1,...,Q$. The total number of coefficient vectors is $n = \sum_{c=1}^{Q} n_c$. We denote the mean coefficient vector of class c by $\mu_c = \frac{1}{n_c}\sum_{l=1}^{n_c} x_{cl}$ and the global mean coefficient vector by $\mu = \frac{1}{n}\sum_{c=1}^{Q}\sum_{l=1}^{n_c} x_{cl}$. Within-class scatter matrix is defined as $S_w = \sum_{c=1}^{Q}\sum_{l=1}^{n_c}(x_{cl} - \mu_c)(x_{cl} - \mu_c)^T$ and represents the scatter of the samples corresponding to the class around their mean. The dispersion of samples that belong to the same class around their corresponding mean should be as small as possible. The between-class scatter matrix is defined as $S_b = \sum_{c=1}^{Q}(\mu_c - \mu)(\mu_c - \mu)^T$ and represents the scatter of the class mean around the global mean μ. Each cluster formed by the samples that belong to the same class must be as far as possible from the other

clusters. Therefore, S_b should be as large as possible. This technique projects the sample variables into a subspace in which the classes are maximally separated by maximizing the between class scatter matrix and minimizing the within class scatter matrix in the same time. The cost function is described by:

$$B = \arg\max_B \frac{\left\| B^T S_b B \right\|}{\left\| B^T S_w B \right\|} \tag{11}$$

When $n \square m$, the within-class scatter matrix S_w is singular. In this case, a common approach is to apply PCA prior to LDA. Accordingly, B_{PCA} is first computed as $B_{PCA} = \text{argmax}_B \left\| B^T S_T B \right\|$, where $S_T = S_B + S_W$ is the total class scatter matrix. Then:

$$B_{LDA} = \arg\max_B \frac{\left\| B^T B_{PCA}^T S_b B_{PCA} B \right\|}{\left\| B^T B_{PCA}^T S_w B_{PCA} B \right\|} \tag{12}$$

is calculated, and finally, the optimum LDA projection direction is given by $B_{opt} = B_{LDA} \cdot B_{PCA}$. LDA has been used widely in many applications such as image retrieval [4] microarray data classification [5], speaker recognition [6], etc. As LDA was specially designed to cope with pattern recognition issues, any application that involves classification task can benefit of it. One of the most popular applications of LDA refers to face recognition [7] where the LDA's recognition performance was compared with PCA. The YALE face database [8] containing human subjects posing under variation in both facial expression and lighting was used in experiments. The results are presented in Table 1 for different methods along with the reduced feature space.

"Leaving-One-Out" of Yale Database			
Method	Reduced Space	Close Crop Error rate (%)	Full Face Error rate (%)
Eigenface	30	24.4	19.4
Eigenface w/o 1st 3	30	15.3	10.8
Correlation	160	23.9	20.0
Linear Subspace	48	21.6	15.6
Fisherface	15	7.3	0.6

Tabel 1. Performance of the algorithms used in [7] when applied to the Yale database which contains variation in facial expression and lighting. The performances were obtained using "leave-one-out strategy" [9]. "Reduced Space" refers to the new feature vectors lower dimension. "Close Crop" refers to images that include internal structures such as the brow, eyes, nose, mouth and chin but did not extend to the occluding contour. For details regarding the other methods see [7].

2.3. Independent Component Analysis

Since PCA relies on the second order statistics of the data, in general, it yields uncorrelated components. When the data have a Gaussian distribution (see Appendix), the uncorrelated components are independent as well. However, if the data is a mixture of non-Gaussian components, PCA fails to extract the components having a non-Gaussian distribution. On the contrary, *Independent Component Analysis* (ICA) [10], which can be viewed as an extension of PCA, takes into account the higher order statistics of the data in the attempt to recover the non-Gaussian components. ICA is able to compute projections which lead to the least Gaussian projected data. This happens because, in the case of non-Gaussian data, the optimization of an independence criterion is equivalent to the maximization of a non-Gaussian criterion

due to the central limit theorem. ICA is not typically used as a dimensionality reduction method in itself. However, in most of applications of ICA, PCA is used as a preprocessing step, in which the newly generated dimensions are ordered by their importance. Based on the PCA transformed data matrix, ICA further transforms the data into independent components. Therefore, using PCA as a preprocessing step, ICA can be viewed as a dimensionality reduction technique, where the resulting latent components $\mathbf{y}$ are as independent as possible. Usually, two preprocessing steps are accomplished before applying ICA. The first is *centering*, i.e., transforming $\mathbf{x}$ into zero-mean variables. The second is *whitening* (or *sphering*), which means that $\mathbf{x}$ is linearly transformed, such that its components are uncorrelated and their variance equals unity (see Appendix). This step is performed through PCA. The noise-free ICA model for the m-dimensional random $\mathbf{x}$ seeks to estimate the components of the p-dimensional $\mathbf{y}$ and mixing matrix $\mathbf{B}$ (eq. 3) assuming $\mathbf{y}_i$ are as independent as possible. Technically, this task recasts in determining an estimation $\mathbf{U}$ of the independent latent components by finding a demixing matrix $\mathbf{W}$:

$$\mathbf{U} = \mathbf{WX} = \mathbf{WAY}, \tag{13}$$

We usually call the columns of $\mathbf{U}$ (and implicitly the columns of $\mathbf{Y}$) *independent sources*. The columns of $\mathbf{X}$ are measurements from a number of sensors that capture the sources. Usually, the number of observed components equals the number of independent components ($m = p$). There is ICA methods that cope with the case of $m < p$ or $m > p$, called *overcomplete* or *undercomplete* ICA, respectively. Estimation of the ICA model consists of two steps, choosing the cost function and applying an optimizing algorithm to minimize or maximize the cost function. There are several ICA implementations. The *InfoMax* algorithm performs ICA based on the information maximization approach proposed by Bell and Sejnowski [11]. This approach relies on the maximization of the entropy of the joint distribution $f(\mathbf{u})$. The demixing matrix $\mathbf{W}$ is updated through an iterative process. At iteration $k+1$, $\mathbf{W}$ is updated according to:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta[\mathbf{I} + (1 - 2\mathbf{z}_k)\mathbf{u}_k^T]\mathbf{W}_k, \tag{14}$$

where η is the learning rate controlling the convergence speed of the algorithm, 1 is a $n \times 1$ vector of ones, $\mathbf{I}$ is the identity matrix of size $n \times n$, and $\mathbf{z}$ is a $n \times 1$ vector having elements:

$$z_i = g(u_i) \qquad i = 1, ..., n \tag{15}$$

with $g(.)$ being a component-wise nonlinearity applied to all elements of the demixer output $\mathbf{u}$, at each iteration k. The form of the nonlinearity must be chosen to match the cumulative distribution function of the input. In the Infomax algorithm [11], this non-linearity is approximated by the logistic transfer function:

$$g(u_i) = 1/(1 + e^{-u_i}) \qquad i = 1, ..., n. \tag{16}$$

The algorithm performs satisfactory when the mixture comprises super-Gaussian components, but fails to extract the components having a sub-Gaussian distribution if such components exist in the mixture of non-Gaussians. Therefore, Lee et al. have extended the InfoMax algorithm to the *extended-InfoMax* approach by using a new learning rule that is able to separate both sub- and super-Gaussian distributions [12]. The learning rule that is able to switch between these distributions iteratively updates the demixing matrix as follows:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \eta[\mathbf{I} - \Xi \tanh(\mathbf{u}_k)\mathbf{u}_k^T - \mathbf{u}_k\mathbf{u}_k^T]\mathbf{W}_k, \tag{17}$$

where Ξ is a $n \times n$ diagonal matrix whose ii-th element, ξ_{ii}, takes the value 1 for a super-Gaussian source and the value -1 for a sub-Gaussian one, and $\tanh()$ denotes the hyperbolic tangent function that is applied to the elements of $\mathbf{u}_k$ in a component-wise fashion. The adaptation of ξ_{ii} is given by:

$$\xi_{ii} = \text{sign}(E\{\text{sech}^2(u_{ki})\}E\{u_{ki}^2\} - E\{[\tanh(u_{ki})]u_{ki}\}), \quad i = 1,...,n \tag{18}$$

where u_{ki} is the i-th element of $\mathbf{u}_k$, and $\text{sign}()$ and $\text{sech}()$ denote the sign and hyperbolic secant functions, respectively.

Other linear ICA implementations include *Joint Approximate Diagonalization of Eigen-matrices* (JADE) proposed by Cardoso and Souloumiac [13] and *fastICA* developed by Hyvärinen [14]. The main advantage of JADE is the fact that it does not need a learning step for its tuning. Its drawback is the relatively small number of components that can be extracted, making it inadequate for a large number of mixture components. The fastICA algorithm is an approach that maximizes the negentropy. A major advantage of fastICA is its speed, making it even 100 times faster than the previously described approaches. Major ICA applications include blind source separation, blind deconvolution, and feature extraction [11]. Figure 2 presents independent feature extraction by using ICA applied to human face images for classifying facial actions [15]. The features are comprised in the columns of basis images $\mathbf{B}$, while $\mathbf{Y}$ represents a matrix containing the new p low dimensional vectors in the reduced space.

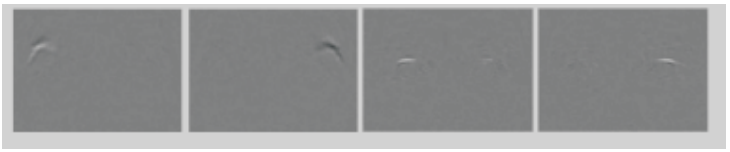

Figure 2. Basis images $\mathbf{B}$ retrieved by ICA from a database containing human face expressing different expressions [15]. Reprinted with the permission from ©1999 IEEE.

2.4. Non-negative Matrix Factorization

Non-negative Matrix Factorization (NMF) has been recently proposed as a dimensionality reduction technique, where both the decomposition factors ($\mathbf{B}$, $\mathbf{Y}$), are constrained to comprise only non-negative values [16]. Allowing only addition for recombining the original data is justified by the intuitive notion of combining parts to form the whole. Another argument for applying non-negativity comes from the neuroscience field where such constrained are related to the non-negative firing rate of the neural receptive fields. Also, these constraints arise in many real image processing applications where the pixels in a gray scale image have non-negative intensity values. Two cost functions were proposed in [17]: Euclidean distance between $\mathbf{X}$ and $\mathbf{BY}$ and Kullback-Leibler (KL) divergence [18]. The KL is expressed as:

$$\sum_{o,i} (x_{oi} \ln \frac{x_{oi}}{\sum_j b_{oj} y_{ji}} + \sum_j b_{oj} y_{ji} - x_{oi}). \tag{19}$$

Recall $o = 1,..,m$, $j = 1,..,p$, and $i = 1,..,n$. The above expression can be minimized and the factors $\mathbf{B}$ and $\mathbf{Y}$ can be determined by applying multiplicative update rules subject to $\mathbf{B},\mathbf{Y} \geq 0$, in a similar way to the Expectation-Maximization algorithm. Although relatively novel, NMF has been already applied on a variety of applications, such as image classification [19], chemometry [20], sound recognition [21], musical audio separation [22] or extraction of summary excerpts from audio and video [23].

2.5. Local Non-negative Matrix Factorization

NMF techniques was extended by Li et al. [24], leading to a new algorithm named *Local Non-negative Matrix Factorization* (LNMF). While, in theory, when NMF is applied to images it should conduct to sparse and local features retrieved by **B** (named here basis images), this is not always the case. LNMF imposes more constraints on the cost function that are related to spatial localization, thus improving the basis images sparseness. Besides the common non-negativity constraints impose to the factors **B** and **Y**, three additional conditions are required:

(1) $\sum_j u_{jj} \longrightarrow \min$, (2) $\sum_{j \neq i} u_{ji} \longrightarrow \min$, and (3) $\sum_j v_{jj} \longrightarrow \max$, with $[\mathbf{u}_{ij}] = \mathbf{U} = \mathbf{B}^T\mathbf{B}$ and $[\mathbf{v}_{ij}] = \mathbf{V} = \mathbf{Y}\mathbf{Y}^T$. The first condition guarantees the generation of more localized features on the basis images $\mathbf{B}$ than those resulting from NMF. The second condition enforces basis orthogonality in order to minimize the redundancy between image bases, and, finally, the total "activity" on each retained component $\mathbf{y}$ (total squared projection coefficients summed over all training images) is maximized through the third condition. The LNMF objective function to be minimized is given by:

$$\sum_{o,i}(x_{oi}\ln\frac{x_{oi}}{\sum_j b_{oj}y_{ji}} + \sum_j b_{oj}y_{ji} - x_{oi}) + \alpha\sum_{oi} u_{oi} - \beta\sum_o v_{oo}, \tag{20}$$

where α, $\beta > 0$ are constants. A solution for the minimization of relation (20) can be found in [24], by employing a strategy analog for finding the NMF's factors.

2.6. Discriminant Non-negative Matrix Factorization

Similar to the idea developed in LDA, LNMF algorithm was further modified so that to incorporate class information in its decomposition. The new class-dependent algorithm called *Discriminant Non-negative Matrix Factorization* (DNMF) was proposed in [25] to optimize the classification procedure. However, the difference between DNMF and LDA is fundamental: whilst FLD preserves the class discriminatory information on the original data $\mathbf{X}$, DNMF performs on the lower dimensional data $\mathbf{Y}$ (named here the decomposition coefficient matrix). Analog to LDA, and besides the conditions borrowed from the LNMF algorithm, DNMF tries to minimize the within-class scatter matrix and to maximize the between-class scatter matrix associated to the new variables $\mathbf{y}_i$. Recall, from LDA, that we have Q distinctive classes and n_c is the number of samples in class $Q, c = 1,...,Q$. The total number of coefficient vectors is $n = \sum_{c=1}^{Q} n_c$. Each image from the image database corresponding to one column of matrix $\mathbf{X}$, belongs to one of these classes. Therefore, each column of the $p \times n$ matrix $\mathbf{Y}$ can be expressed as image representation coefficients vector $\mathbf{y}_{cl}$, where $c = 1,...,Q$ and $l = 1,...,n_c$. Further, the notations are similar with the one defined for LDA, where the original variables $\mathbf{x}$ are replaced by $\mathbf{y}$. Thus, the DNMF cost function to be minimized is expressed as follows:

$$\sum_{o,i}(x_{oi}\ln\frac{x_{oi}}{\sum_j b_{oj}y_{ji}} + \sum_j b_{oj}y_{ji} - x_{oi}) + \alpha\sum_{oi} u_{oi} - \beta\sum_o v_{oo} + \gamma\sum_{c=1}^{Q}\sum_{l=1}^{n_c}(y_{cl}-\mu_c)(y_{cl}-\mu_c)^T$$
$$-\delta\sum_{c=1}^{Q}(\mu_c - \mu)(\mu_c - \mu)^T \tag{21}$$

NMF, LNMF, and DNMF were applied for subspace image representation, face recognition [24] and facial expression classification tasks [25].

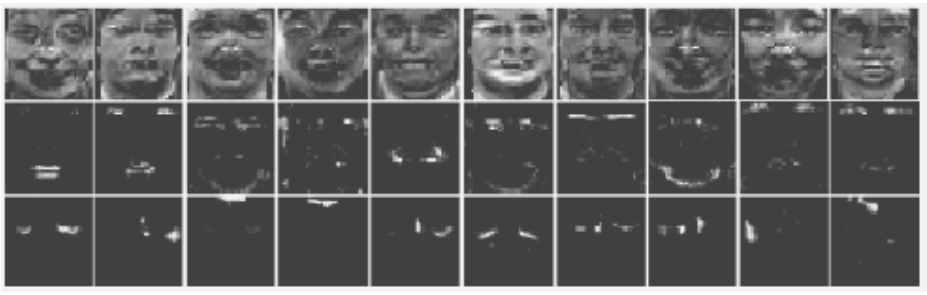

Figure 3. A set of learned basis images using (from left to right) NMF, DNMF, and LNMF [25]. The algorithms were applied to the Cohn-Kanade facial expression database [26].

Every NMF-based dimensionality reduction techniques conduct to different subspace image representations. It can be noticed by visual inspection from Figure 3 that the basis images retrieved by DNMF are not as sparse as those extracted by LNMF but are more sparse than the basis images found by NMF. Also, almost all features found by the DNMF's basis images are represented by the salient face features such as eyes, eyebrows or mouth features, while, the features retrieved by LNMF have rather random positions.

3. Nonlinear Dimensionality Reduction and Manifold Learning Techniques

3.1. Kernel Principal Component Analysis

The *Kernel Principal Component Analysis* (KPCA) [27] is based on the idea of translating the original data $\mathbf{x}_i$ to a higher-dimensional space. At the first glance, by mapping the original data onto a higher space seems to be contrary to the dimensionality reduction principle. However, by employing the same analysis as for the linear PCA and using the so-called "kernel trick" [28] the KPCA computation can be performed in a lower-dimensional space. As aforementioned, the input space where the original data reside is transformed by a nonlinear map onto a feature space, $\varphi : \Re^m \to F^q$, $q > m$. The main rationale of developing such an approach comes from the statistical pattern recognition field and is given by the fact that, in a higher-dimension space the data are (hopefully) more separable (thus improving the classification performance) than in the original variable space. Another reason is that, by employing a nonlinear PCA, one can retrieve higher-order correlations between variables, in contrast to linear PCA which accounts only for second-order variable dependencies. By applying the map φ, the sample covariance matrix (2) is recast into:

$$\Sigma_{KPCA} = E\{\varphi(\mathbf{x}-\mu)\varphi(\mathbf{x}-\mu^T)\}, \tag{22}$$

The eigenvectors $\mathbf{b}_j$ and the corresponding eigenvalues λ_j can be computed by solving the equation:

$$\Sigma_{KPCA}\mathbf{b}_j = \lambda_j \mathbf{b}_j \, j = 1,..., p. \tag{23}$$

Defining a $m \times m$ matrix $\mathbf{K}$ (kernel operator) by:

$$K_{ij} = (\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)) \tag{24}$$

Hence, the relation (3) translates into finding the eigenvectors and eigenvalues of the dot product matrix $\mathbf{K}$ by solving:

$$n\lambda\alpha = \mathbf{K}\alpha \tag{25}$$

With the eigenvectors normalized and eigenvalues computed, a new test variable is then projected onto the eigenvectors to reduce its dimensionality, as follows:

$$(\mathbf{B}^k \varphi(\mathbf{x}_i)) = \sum_{i=1}^{n} \alpha_i^k (\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x})) \tag{26}$$

The above projection refers to the nonlinear PCs associated to φ. There are several kernels available where ones of the most used are represented by the linear $\mathbf{x}_i^T \mathbf{x}_i$, polynomial $(\mathbf{x}_i^T \mathbf{x}_i)^d$, and the exponential radial basis function (ERBF) $\exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_1}{2\sigma^2})$, with d the polynomial degree and σ - ERBF parameter, respectively. KPCA applications include face recognition [29], object and character recognition [27]. An example of the utility of KPCA for handwritten character recognition (on the US postal service -USPS- database) [27] is presented below. The database contains 9300 examples of

dimensionality 256. As one can see from Table 2 the polynomial kernel PCA followed by a linear Support Vector classifier leads to performance superior to PCA. Also, since KPCA allows processing in a higher dimensionality space than the original one (256), the experiments were conducted for the extraction of up to 2048 nonlinear PCs, while, for PCA, the size of PCs was limited to 256.

# of components	Test Error Rate for degree						
	1	2	3	4	5	6	7
32	9.6	8.8	8.1	8.5	9.1	9.3	10.8
64	8.8	7.3	6.8	6.7	6.7	7.2	7.5
128	8.6	5.8	5.9	6.1	5.8	6.0	6.8
256	8.7	5.5	5.3	5.2	5.2	5.4	5.4
512	n.a.	4.9	4.6	4.4	5.1	4.6	4.9
1024	n.a.	4.9	4.3	4.4	4.6	4.8	4.6
2048	n.a.	4.9	4.2	4.1	4.0	4.3	4.4

Table 2. Test error rates for the USPS handwritten digit database when linear Support Vector machines (used as classifiers) are trained on nonlinear principal components extracted by PCA with polynomial kernel for degree 1 to 7 [27]. Here "# of. components" refers to p from this paper.

3.2. Kernel Discriminant Analysis

Employing the same kernel theory as in KPCA, the LDA approach can be generalized, leading to nonlinear data dimensionality reduction. *Kernel Discriminant Analysis* (KDA), also known as Kernel Fisher Discriminant (KFD) [30] performs by computing a cost function in the new feature space, described as:

$$B^{\varphi} = \arg\max_{B} \frac{\left\| B^T S_b^{\varphi} B \right\|}{\left\| B^T S_w^{\varphi} B \right\|} \tag{27}$$

which is a natural extension of (14), and φ is the associated nonlinear mapping. For a two-class problem, $S_w^{\varphi} = \sum_{c=1}^{Q} \sum_{l=1}^{n_c} (\varphi(x_{cl}) - \mu_c^{\varphi})(\varphi(x_{cl}) - \mu_c^{\varphi})^T$ and

$S_b^{\varphi} = \sum_{c=1}^{Q} (\mu_c^{\varphi} - \mu^{\varphi})(\mu_c^{\varphi} - \mu^{\varphi})^T$, where $Q \in \{Q_1, Q_2\}$ and $n_c \in \{n_{c1}, n_{c2}\}$. The Fisher's linear discriminant cost function is expressed by: $\alpha = \text{argmax}_{\alpha} \frac{\alpha^T M \alpha}{\alpha^T N \alpha}$, where the notations are defined as: $M = (\mathbf{M}_1 - \mathbf{M}_2)(\mathbf{M}_1 - \mathbf{M}_2)^T$, $(\mathbf{M}_i)_Q = 1/Q \sum_{k=1}^{textslQ} k(\mathbf{x}_Q, \mathbf{x}_k^i)$, $N = \sum_{Q=1,2} K_Q (\mathbf{I} - 1_n) K_Q^T$. K_Q is the kernel matrix for the class Q (see [30] for details). As typical application, KFD was used in pattern recognition. The work of Mika et al. [30] presents a performance comparison between KFD and some other approaches for 13 artificial and real datasets from several benchmark repositories (see [30] for details) employed in classification task. The results are depicted in Table 3, where for the KFD a Gaussian kernel was used (see [30] for details).

	RBF	AB	AB_R	SVM	KFD
Banana	**10.8±0.6**	12.3±0.7	*10.9±0.4*	11.5±0.7	**10.8±0.5**
B.Cancer	27.6±4.7	30.4±4.7	26.5±4.5	*26.0±4.7*	**25.8±4.6**
Diabetes	24.3±1.9	26.5±2.3	23.8±1.8	*23.5±1.7*	**23.2±1.6**
German	24.7±2.4	27.5±2.5	24.3±2.1	**23.6±2.1**	*23.7±2.2*
Heart	17.6±3.3	20.3±3.4	16.5±3.5	**16.0±3.3**	*16.1±3.4*
Image	3.3±0.6	**2.7±0.7**	**2.7±0.6**	*3.0±0.6*	4.8±0.6
Ringnorm	1.7±0.2	1.9±0.3	*1.6±0.1*	1.7±0.1	**1.5±0.1**
F.Sonar	34.4±2.0	35.7±1.8	34.2±2.2	**32.4±1.8**	*33.2±1.7*
Splice	*10.0±1.0*	10.1±0.5	**9.5±0.7**	10.9±0.7	10.5±0.6
Thyroid	4.5±2.1	*4.4±2.2*	4.6±2.2	4.8±2.2	**4.2±2.1**
Titanic	23.3±1.3	*22.6±1.2*	*22.6±1.2*	**22.4±1.0**	23.2±2.0
Twonorm	2.9±0.3	3.0±0.3	*2.7±0.2*	3.0±0.2	**2.6±0.2**
Waveform	10.7±1.1	10.8±0.6	**9.8±0.8**	*9.9±0.4*	*9.9±0.4*

Table 3. Comparison between KFD, a single RBF classifier, AdaBoost (AB), regularized AdaBoost (AB_R) and Support Vector Machine (SVM) [30]. Best methods are illustrated as bold letters, the second best are emphasized.

3.3. Kernel Independent Component Analysis

Kernel Independent Component Analysis (KICA) has been proposed to replace classical ICA when the components are mixed using nonlinear functions [31]. A kernel Hilbert space is used to extract such sources that are nonlinearly mixed. Two contrast functions based on canonical correlations in this reproducing space have been defined namely the *kernel ICA-KCCA* (where KCCA stands for Kernel Canonical Correlation Analysis) and the *ICA-KGV* (where KGV stands for Kernel Generalized Variance). Kernel ICA-KCCA minimizes the first kernel canonical correlation that depends on the data $\mathbf{x}_i$, $i = 1,...,n$ only through the centered Gram matrices for l ICs. Kernel ICA-KGV minimizes the kernel generalized variance. Both contrast functions are related to a generalized eigenvector problem $K_\kappa \alpha = \lambda D_\kappa \alpha$, where κ is a regularization parameter and K_κ and D_κ are block matrices constructed from the Gram matrices. Kernel ICA-KCCA deals with the minimal eigenvalue of the aforementioned problem while kernel ICA-KGV deals with the entirespectrum. More theoretical details can be found in [31]. KICA have been used for face recognition [32] under several conditions, along with various subspace methods for comparison. However, as the results are tabulated in Table 4, KICA does not seem to be very efficient for this task, being outperformed by techniques such as KPCA, FDA (LDA) or KFDA (KDA).

Recognition Rate (%)	PCA	KPCA	ICA	KICA	FDA	KFDA	PPCA
Pose	92.2	90.1	88.2	89	89.1	95.3	81.4
Facial Expression	61	61	53	46	72	73	68
Illumination	57	61	57	59	75	65	72

Table 4. Face recognition rate for various subspace methods including KICA and under different recording conditions of faces [32].

4. Discussions

Several linear and nonlinear dimensionality techniques have been presented in this overview along with one representative application. Although they were successfully applied to a variety of many other applications and fields as mentioned, each method has its own shortcomings. One of the PCA's drawbacks is its computational complexity for large data dimensions and number of samples (hundreds or several thousands). This issue comes from the computation of the sample covariance matrix. A way to avoid this computation was proposed by Roweis who developed a PCA variant based on EM algorithm [33] where no covariance matrix is needed to be calculated. Another limitation of the classical PCA is driven by a batch computation step. This means that, when a new training sample comes this new sample

must be added to the entire training data and the PCA training procedure has to be rerun for this new formed training set. Several incremental methods for the computation of the eigenvectors have been introduced to overcome the limitation, where the learning step is performed on-line as each new training sample is added [34], [35]. As far as the recognition issue is concerned, PCA is very sensitive to illumination changes that can dramatically influence the recognition performance. Usually, to increase the PCA's robustness to this factor, the first three principal components are discarded prior to apply the recognition step. In spite the fact that LDA has been designed to cope with recognition tasks, it may fail to provide satisfactory results under some certain circumstances. One of them is the violation of the linearity assumption, and another one is the limited number of samples [36]. NMF and its relatives (LNMF and DNMF) suffer from the lack of a global algorithmic convergence. To converge, the method has to be run several times to assure the finding of a local minimum.

Nonlinear dimensionality reduction approaches have their own disadvantages. One of their drawbacks is the computational complexity and the necessary processing time that is, generally higher than the one allocated for the linear approaches. A direct analytical comparison can be performed between KPCA, KDA, and KICA, versus their linear parts, PCA, LDA, and ICA, respectively. The difference resides in computing the kernel matrix, which, for large number of samples may be highly timely consuming. Also, the lack of theoretical rationale for choosing a specific kernel along with its optimum parameter (some empirical approaches such as cross-validation, etc. exists for tuning, though) is another shortcoming of these techniques.

REFERENCES

1. JOLLIFFE, T., **Principal Component Analysis**, (2nd ed.), New York: Springer-Verlag, 2002.

2. ANDERSON, E., **The Irises of the Gaspe Peninsula**, Bulletin of the American Iris Society, Vol. 59, 1935, pp. 2–5.

3. FISHER, R. A., **The use of Multiple Measures in Taxonomic Problems**, Ann. Eugenics, Vol. 7, 1936, pp. 179–188.

4. SWETS, D. L., J. WENG, **Using Discriminant Eigenfeatures for Image Retrieval**, Trans. on Pattern Analysis and Machine Intelligence, Vol. 18, No. 8, 1996, pp. 831–836.

5. DUDOIT, S., J. FRIDLYAND, T. P. SPEED, **Comparison of Discrimination Methods for the Classification of Tumors Using Gene Expression Data**, Journal of the American Statistical Association, Vol. 97, No. 457, 2002, pp. 77–87.

6. JIN, Q., A. WAIBEL, **Application of LDA to Speaker Recognition**, in Proc. of International Conference on Spoken Language Processing, Vol. 2, 2000, pp. 250–253.

7. BELHUMEUR, P. N., J. P. HESPANHA, D. J. KRIEGMAN, **Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection**, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 19, No. 7, 1997, pp. 711-720.

8. http://cvc.yale.edu.

9. DUDA, R. O., P. E. HART, D. G. STORK, **Pattern Classification**, Second Edition, Wiley-Interscience, 2000.

10. HYVÄRINEN, A., J. KARHUNEN, E. OJA, **Independent Component Analysis**, John Wiley & Sons, Toronto, 2001.

11. BELL, A. J., T. J. SEJNOWSKI, **An Information-maximization Approach to Blind Separation and Blind Deconvolution,** Neural Computation, Vol. 7, No. 6, 1995, pp. 1129–1159.

12. . LEE, T.-W., M. GIROLAMI, T. J. SEJNOWSKI, **Independent Component Analysis Using an Extended Infomax Algorithm for Mixed Sub-Gaussian and Super-Gaussian Sources,** Neural Computation, Vol. 11, No. 2, 1999, pp. 417–441.

13. CARDOSO, J. F., A. SOULOUMIAC, **Blind Beamforming for Non Gaussian Signals**, IEE Proceedings-F, Vol. 140, No. 6, 1993, pp. 362-370.

14. HYVÄRINEN, A., **Fast and Robust Fixed-point Algorithms for Independent Component Analysis**, IEEE Trans. Neural Networks, Vol. 10, No. 3, 1999, pp. 626–634.

15. DONATO, G., M. S. BARTLETT, J. C. HAGER, P. EKMAN, T. J. SEJNOWSKI, **Classifying Facial Actions**, IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 21, No. 10, 1999, pp. 974–989.

16. LEE, D. D., H. S. SEUNG, **Learning the Parts of the Objects by Non-negative Matrix Factorization**, Nature, Vol. 401, 1999, pp. 788–791.

17. LEE, D D., H. S. SEUNG, **Algorithms for Non-negative Matrix Factorization**, Advances Neural Information Processing Systems, Vol. 13, 2001, pp. 556–562.

18. KULLBACK, S., R. LEIBLER, **On Information and Sufficiency**, Annals of Mathematical Statistics, No. 22, 1951, pp. 79–86.

19. GUILLAMET, D., B. SCHIELE, J. VITRI, **Analyzing Non-negative Matrix Factorization for Image Classification**, in Proc. of 16th Int. Conf. on Pattern Recognition, Vol. II, 2002, pp. 116–119.

20. PAATERO, P., U. TAPPER, **Positive Matrix Factorization: A Non-negative Factor Model with Optimal Utilization of Error Estimates of Data Values**, Environmetric*s*, Vol. 5, 1994, pp. 111–126.

21. KAWAMOTO, T., K. HOTTA, T. MISHIMA, J. FUJIKI, M. TANAKA, T. KURITA, **Estimation of Single to Nes from Chord Sounds Using Non-negative Matrix Factorization**, in Neural Network World, Vol. 3, 2000, pp. 429–436.

22. WANG, B., M. D. PLUMBLEY**, Musical Audio Stream Separation by Non-negative Matrix Factorization**, in Proc. of DMRN Summer Conference, Glasgow, 2005.

23. COOPER M., J. FOOTE, **Summarizing Video Using Non-negative Similarity Matrix Factorization**, in Proc. IEEE Workshop on Multimedia Signal Processing, 2002, pp. 25–28.

24. LI, S. Z., X. W. HOU, H. J. ZHANG, **Learning Spatially Localized, Parts-based Representation**, Int. Conf. Computer Vision and Pattern Recognition, 2001, pp. 207–212.

25. BUCIU, I., I. PITAS, **A New Sparse Image Representation Algorithm Applied to Facial Expression Recognition**, in IEEE Workshop on Machine Learning for Signal Processing, 2004, pp. 539–548.

26. KANADE, T., J. COHN, Y. TIAN, **Comprehensive Database for Facial Expression Analysis**, in Proc. IEEE Inter. Conf. on Face and Gesture Recognition, 2000, pp. 46–53.

27. SCHÖLKOPF, B., A. J. SMOLA, K.-R. MÜLLER, **Nonlinear Component Analysis as a Kernel Eigenvalue problem**, Neural Computation, Vol. 10, No. 5, pp. 1299–1319, 1998.

28. TAYLOR, J. S., N. CRISTIANINI, **Kernel Methods for Pattern Analysis**, Cambridge University Press, 2004.

29. YANG, M.-H., **Kernel Eigenfaces vs. Kernel Fisherfaces: Face Recognition Using Kernel Methods**, in Proc. IEEE International Conference on Automatic Face and Gesture Recognition, 2002, pp. 215–220.

30. MIKA, S., G. RÄTSCH, J. WESTON, B. SCHÖLKOPF, K.-R. MÜLLER, **Fisher Discriminant Analysis with Kernels**, in Neural Networks for Signal Processing *IX*, 1998, pp. 41–48.

31. BACH, F. R., M. J. JORDAN, **Kernel Independent Component Analysis**, Machine Learning Research, Vol. 3, 2002, pp. 1–48.

32. LI, J., S. ZHOU, C. SHEKHAR, **A Comparison of Subspace Analysis for Face Recognition**, in IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 3, 2003, pp. 121–124.

33. ROWEIS, S., **EM Algorithms for PCA and SPCA**, Neural Information Processing Systems, Vol. 10, 1997, pp. 626–632.

34. CHANDRASEKARAN, S., B. S. MANJUNATH, Y. F. WANG, J. WINKELER, H. ZHANG, **An Eigenspace Update Algorithm for Image Analysis**, Graphical models and image processing: GMIP, Vol. 59, No. 5, 1997, pp. 321–332.

35. ARTAC, M., M. JOGAN, A. LEONARDIS, **Incremental PCA for On-line Visual Learning and Recognition**, in Int. Conf. on Pattern Recognition, Vol. 3, 2002, pp. 781–784.

36. MARTINEZ, A. M., A. C. KAK, **PCA Versus LDA**, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 23, No. 2, 2001, pp. 228–233.