International Scholarly Research Network
ISRN Artificial Intelligence
Volume 2012, Article ID 847305, 19 pages
doi:10.5402/2012/847305

Review Article

Neural Network Implementations for PCA and Its Extensions

Jialin Qiu,[1] Hui Wang,[1] Jiabin Lu,[2] Biaobiao Zhang,[1] and K.-L. Du[1,3]

[1] *Enjoyor Labs, Enjoyor Inc., Hangzhou 310030, China*
[2] *Faculty of Electromechanical Engineering, Guangdong University of Technology, Guangzhou 510006, China*
[3] *Department of Electrical and Computer Engineering, Concordia University, Montreal, QC, Canada H3G 1M8*

Correspondence should be addressed to K.-L. Du, kldu@ece.concordia.ca

Received 8 April 2012; Accepted 14 June 2012

Academic Editors: C. Kotropoulos and B. Schuller

Many information processing problems can be transformed into some form of eigenvalue or singular value problems. Eigenvalue decomposition (EVD) and singular value decomposition (SVD) are usually used for solving these problems. In this paper, we give an introduction to various neural network implementations and algorithms for principal component analysis (PCA) and its various extensions. PCA is a statistical method that is directly related to EVD and SVD. Minor component analysis (MCA) is a variant of PCA, which is useful for solving total least squares (TLSs) problems. The algorithms are typical unsupervised learning methods. Some other neural network models for feature extraction, such as localized methods, complex-domain methods, generalized EVD, and SVD, are also described. Topics associated with PCA, such as independent component analysis (ICA) and linear discriminant analysis (LDA), are mentioned in passing in the conclusion. These methods are useful in adaptive signal processing, blind signal separation (BSS), pattern recognition, and information compression.

1. Introduction

In information processing such as pattern recognition, data compression and coding, image processing, high-resolution spectrum analysis, and adaptive beamforming, feature extraction or feature selection is necessary to deal with the large storage of raw data. Feature extraction is a dimensionality-reduction technique, mapping high-dimensional patterns onto a lower-dimensional space by extracting the most prominent features using orthogonal transforms. The extracted features do not have any physical meaning. In contrast, feature selection decreases the size of the feature set or reduces the dimension of the features by discarding the raw information according to a criterion.

Orthogonal decomposition is a well-known technique to eliminate ill-conditioning. The Gram-Schmidt orthonormalization (GSO) is suitable for feature selection. This is due to the fact that the physically meaningless features in Gram-Schmidt space can be linked back to the same number of variables of the measurement space, thus resulting in no dimensionality reduction. The GSO procedure starts with QR decomposition of the transpose of the full feature matrix, $\mathbf{X}^T$, where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$. QR decomposition can be performed by using the Householder transform or the Givens rotation [1], which is suitable for hardware implementation. The GSO transform can be used for feature subset selection; it inherits the compactness of the orthogonal representation and at the same time provides features that retain their original meaning.

An orthogonal transform can decompose the correlations among the candidate features so that the significance of the individual features can be evaluated independently. Principal component analysis (PCA) is a well-known orthogonal transform that is used for dimensionality reduction. Another popular technique for feature extraction is linear discriminant analysis (LDA), also known as Fisher's discriminant analysis [2, 3]. Taking all the data into account, PCA computes vectors that have the largest variance associated with them. The generated PCA features do not have clear physical meanings. In contrast, LDA searches for those vectors in the underlying space that best discriminate among the classes rather than those that best describe the data.

For a J_1-dimensional data set $\{\mathbf{x}_i\}$ of size N, PCA [4] generates a J_2-dimensional feature set $\{\mathbf{y}_i\}$ of the same size, $J_1 > J_2$, by using the linear transformation $\mathbf{y}_i = \mathbf{W}^T\mathbf{x}_i$. The weight matrix $\mathbf{W}$ can be solved under different criteria such

as the output variance maximization or MSE minimization. In comparison with the GSO transform, PCA generates each of its features based on the covariance matrix of all the N vectors $\mathbf{x}_i$, $i = 1, \ldots, N$. Dimensionality reduction is achieved by dropping the variables with insignificant variances. PCA is often used to select inputs, but it is not always useful, since the variance of a signal is not always related to the importance of the signal, for non-Gaussian signals. An improvement on PCA is provided by nonlinear generalizations of PCA, which extend the ability of PCA to incorporate nonlinear relationships in the data. Two-dimensional PCA [5] is designed for image feature extraction.

In this paper, we give a state-of-the-art introduction to various neural network implementations and algorithms for PCA and its extensions. This paper is organized as follows. In Section 2, we introduce the Hebbian learning rule and Oja's learning rule. Section 3 defines the PCA problem. Various PCA networks and algorithms are treated in Sections 4–7. PCA algorithms based on the Hebbian rule are expanded in Section 4. In Section 5, least means squared error-based PCA methods are dealt with. Other optimization-based PCA methods are described in Section 6. PCA based on the anti-Hebbian rule is treated in Section 7. Nonlinear PCA is addressed in Section 8. Section 9 is dedicated to minor component analysis (MCA). In Section 10, we describe various localized PCA strategies. Section 11 extends all the methods to the complex-valued domain. Some other generalizations of PCA such as constrained PCA, generalized EVD, and two-dimensional PCA are described in Section 12. In Section 13, the cross-correlational PCA asymmetric network and SVD are described. Canonical correlation analysis is described in Section 14. A simulation example of PCA is given in Section 15. A brief summary is given in Section 16, and independent component analysis (ICA) and linear discriminant analysis (LDA) are also mentioned in passing in this section.

2. Hebbian Learning Rule and Oja's Learning Rule

The stochastic approximation theory [6], introduced by Robbins and Monro in 1951, is an important tool for analyzing stochastic discrete-time systems including the classical gradient-descent method. Given a stochastic discrete-time system of the form

$$\mathbf{z}(t+1) = \mathbf{z}(t) + \eta(t)(\mathbf{f}(\mathbf{z}, t) + \mathbf{n}(t)), \tag{1}$$

where $\mathbf{z}$ is the state vector, $\mathbf{f}(\mathbf{z}, t)$ a finite nonzero vector that uses functions as entries, and $\mathbf{n}(t)$ an unbiased noisy term. Assuming that $\{\eta(t)\}$ is a sequence of positive numbers satisfying the Robbins-Monro conditions [6]

$$\sum_{t=1}^{\infty} \eta(t) = \infty, \qquad \sum_{t=1}^{\infty} \eta^2(t) < \infty, \tag{2}$$

then the stochastic system (1) can be transformed into a deterministic differential equation

$$\frac{d\mathbf{z}}{dt} = \mathbf{f}(\mathbf{z}, t). \tag{3}$$

If (3) converges to a fixed point $\mathbf{z}^*$, then (1) also converges to $\mathbf{z}^*$ as $t \to \infty$ with probability one. The Robbins-Monro conditions [6] require $\eta(t) \to 0$ as $t \to \infty$. Typically, one can select $\eta(t) = 1/(\alpha + t)$, $\alpha \geq 0$ being a constant, or $\eta(t) = 1/t^{\beta}$, $1/2 \leq \beta \leq 1$ [7, 8].

The Hebbian learning rule was introduced in [9]. For a single neuron, the Hebbian rule is written as

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta y(t) \mathbf{x}_t, \tag{4}$$

where the learning rate $\eta > 0$, $\mathbf{w}$ is the weight vector from the input to the neuron, $\mathbf{x}_t$ is an input vector presented at time t, and the output of the neuron $y(t)$ is defined by

$$y(t) = \mathbf{w}^T(t) \mathbf{x}_t. \tag{5}$$

For a stochastic input vector $\mathbf{x}$, applying an analysis on (4) using the stochastic approximation theory, we get the only equilibrium state $\mathbf{w} = \mathbf{0}$ by maximizing the criterion $E_{\text{Hebb}} = E[y^2]$, where $E[\cdot]$ is the expectation operator [10, 11]. The solution $\mathbf{w} = \mathbf{0}$ is unstable, which drives $\mathbf{w}$ to infinite magnitude, with a direction parallel to that of the eigenvector of $\mathbf{C} = E[\mathbf{x}\mathbf{x}^T]$ corresponding to the largest eigenvalue [11]. To prevent the divergence of the Hebbian rule, $\mathbf{W}$ can be normalized after each iteration [12, 13], and this leads to the normalized Hebbian rule. Other methods, such as Oja's rule [14], Yuille's rule [15], Linsker's rule [16], and Hassoun's rule [11], add a weight-decay term to the Hebbian rule to stabilize the algorithm.

Oja's rule introduces a weight decay term into the Hebbian rule [14] to prevent instability

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta y(t) \mathbf{x}_t - \eta y^2(t) \mathbf{w}(t). \tag{6}$$

Oja's rule converges to a state that maximizes E_{Hebb} subject to $\|\mathbf{w}\| = 1$. The solution is the principal eigenvector of $\mathbf{C}$ [14]. For small η, Oja's rule is proved to be equivalent to the normalized Hebbian rule [14]. Based on stochastic approximation theory, the stable solutions exist only at $\mathbf{w} = \pm \mathbf{c}_1$, if $\lambda_1 \neq \lambda_2$, where $\mathbf{c}_1$ is the eigenvector corresponding to λ_1, and λ_1 and λ_2 are the two largest eigenvalues of $\mathbf{C}$ [10, 11]. Thus, Oja's rule always converges to the principal component of $\mathbf{C}$.

The Robbins-Monro conditions are not practical for implementation, especially for learning nonstationary data. Zufiria [17] has proposed to convert the stochastic discrete-time algorithms into their deterministic discrete-time formulations that characterize their average evolution from a conditional expectation perspective. Analysis based on this method guarantees the convergence of Oja's rule by selecting some constant learning rate. Oja's rule almost always converges exponentially to the unit eigenvector associated with the largest eigenvalue of $\mathbf{C}$, starting from points in an invariant set [18]. A constant learning rate for fast convergence is suggested as $\eta = 0.618/\lambda_1$ [18].

3. Principal Component Analysis

PCA is based on the spectral analysis of the second-order moment matrix called correlation matrix that statistically characterizes a random vector. In the zero-mean case, this

matrix becomes the covariance matrix. In the area of image coding, PCA is known as Karhunen-Loeve transform (KLT) [4], which exploits correlation between neighboring pixels or groups of pixels for data compression.

PCA allows the removal of the second-order correlation among given random processes. By calculating the eigenvectors of the covariance matrix of the input vector, PCA linearly transforms a high-dimensional input vector into a low-dimensional one whose components are uncorrelated. PCA is directly related to singular value decomposition (SVD), and the most common way to perform PCA is via SVD of the data matrix. However, the capability of SVD is limited for vey large data set.

PCA is often derived by optimizing some information criterion, such as the maximization of the variance of the projected data or the minimization of the reconstruction error. The objective of PCA is to extract m orthonormal directions $\overline{\overline{\mathbf{w}}}_i \in R^n$, $i = 1, 2, \ldots, m$, $m < n$, in the input space that account for as much of the data's variance as possible. Subsequently, an input vector $\mathbf{x} \in R^n$ may be transformed into an m-dimensional space without losing essential intrinsic information. The vector $\mathbf{x}$ can be represented by being projected onto the m-dimensional subspace spanned by $\overline{\overline{\mathbf{w}}}_i$ using the inner products $\mathbf{x}^T\overline{\overline{\mathbf{w}}}_i$, hence achieving dimensionality reduction.

PCA finds those unit directions $\overline{\overline{\mathbf{w}}} \in R^n$, along which the projections of the input vectors, known as the principal components (PCs), $y = \mathbf{x}^T\overline{\overline{\mathbf{w}}}$, have the largest variance

$$E_{\mathrm{PCA}}(\mathbf{w}) = E[y^2] = \overline{\overline{\mathbf{w}}}^T\mathbf{C}\overline{\overline{\mathbf{w}}} = \frac{\mathbf{w}^T\mathbf{C}\mathbf{w}}{\|\mathbf{w}\|^2}, \tag{7}$$

where $\overline{\overline{\mathbf{w}}} = \mathbf{w}/\|\mathbf{w}\|$. Based on an analysis using the stochastic approximation theory [10, 11], when $\mathbf{w} = \alpha\mathbf{c}_1$, where α is a scalar, $E_{\mathrm{PCA}}(\mathbf{w})$ takes its maximum value. When $\alpha = 1$, $\mathbf{w}$ becomes a unit vector.

By repeating maximization of $E_{\mathrm{PCA}}(\mathbf{w})$ but limiting $\mathbf{w}$ to be orthogonal to $\mathbf{c}_1$, the maximum of $E_{\mathrm{PCA}}(\mathbf{w})$ is equal to λ_2 at $\mathbf{w} = \alpha\mathbf{c}_2$. Following this deflation procedure, all the m principal directions $\overline{\overline{\mathbf{w}}}_i$ can be derived [11]. The projections $y_i = \mathbf{x}^T\overline{\overline{\mathbf{w}}}_i$, $i = 1, 2, \ldots, m$, are the PCs of $\mathbf{x}$.

A linear least squares (LS) estimate $\hat{\mathbf{x}}$ can be constructed for the original input $\mathbf{x}$ as $\hat{\mathbf{x}}_t = \sum_{i=1}^{m} a_i(t)\overline{\overline{\mathbf{w}}}_i$. This is a data reconstruction process. The reconstruction error $\mathbf{e}$ is the difference between the original and reconstructed data

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}} = \sum_{i=m+1}^{n} a_i\overline{\overline{\mathbf{w}}}_i. \tag{8}$$

Naturally, $\mathbf{e}$ is orthogonal to $\hat{\mathbf{x}}$. Each principal component a_i is a Gaussian with zero mean and variance $\sigma_i^2 = \lambda_i$.

4. Hebbian Rule-Based Principal Component Analysis

Neural PCA originates from the seminal work by Oja [14]. For a single neuron, the output is given by

$$y = \mathbf{w}^T\mathbf{x}, \tag{9}$$

where the weights to the neuron $\mathbf{w} = (w_1, \ldots, w_{J_1})^T$.

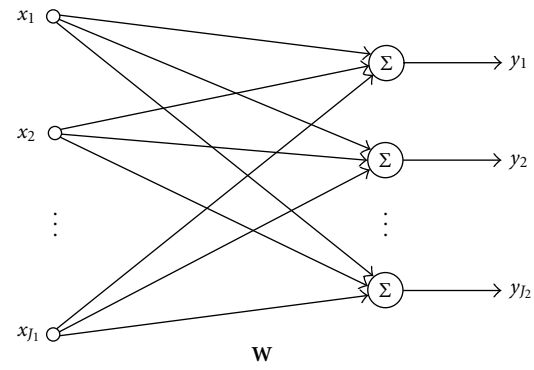

Figure 1: Architecture of the PCA network.

The single-neuron model was extended to a J_1-J_2 feed-forward network model to extract the first J_2 PCs [7]. The architecture of the PCA network is shown in Figure 1. The output of the network is given by

$$\mathbf{y} = \mathbf{W}^T\mathbf{x}, \tag{10}$$

where $\mathbf{y} = (y_1, y_2, \ldots, y_{J_2})^T$, $\mathbf{x} = (x_1, x_2, \ldots, x_{J_1})^T$, $\mathbf{W} = [\mathbf{w}_1 \ \mathbf{w}_2 \cdots \mathbf{w}_{J_2}]$, and $\mathbf{w}_i = (w_{1i}, w_{2i}, \ldots, w_{J_1 i})^T$, w_{jk} being the weight from the jth input to the kth neuron.

4.1. Subspace Learning Algorithms. By using Oja's rule (6), $\mathbf{w}$ will converge to a unit eigenvector of $\mathbf{C}$, and the variance of y is maximized. For zero-mean input data, this extracts the first PC [14, 19]. In order to keep the algorithm convergent, $0 < \eta(t) < 1/1.2\lambda_1$ is required [7]. If $\eta(t) \geq 1/\lambda_1$, $\mathbf{w}$ will not converge to $\pm\mathbf{c}_1$ even if it is initially close to the target [20]. One can select $\eta(t) = 0.5[\mathbf{x}_t^T\mathbf{x}_t]$ at the beginning and gradually decrease η [7].

The symmetrical subspace learning algorithm (SLA) [7] is a learning algorithm for the PCA network. The SLA is based on Oja's rule and is given by [7]

$$\begin{aligned} \mathbf{w}_i(t+1) &= \mathbf{w}_i(t) + \eta(t)y_i(t)[\mathbf{x}_t - \hat{\mathbf{x}}_t], \\ \hat{\mathbf{x}}_t &= \mathbf{W}\mathbf{y}. \end{aligned} \tag{11}$$

After the algorithm converges, $\mathbf{W}$ is roughly orthonormal and the columns of $\mathbf{W}$, namely, $\mathbf{w}_i$, $i = 1, \ldots, J_2$, converge to some linear combination of the first J_2 principal eigenvectors of $\mathbf{C}$ [7, 21], which is a rotated basis of the dominant eigenvector subspace. This is called principal subspace analysis (PSA). The value of $\mathbf{w}_i$ is dependent on the initial conditions and training samples. The corresponding eigenvalues λ_i approximate $E[y_i^2]$ and can be adaptively estimated by

$$\hat{\lambda}_i(t+1) = \left(1 - \frac{1}{t+1}\right)\hat{\lambda}_i(t) + \frac{1}{t+1}y_i^2(t+1). \tag{12}$$

Weighted SLA introduces asymmetry into the SLA [22, 23]:

$$\begin{aligned} \mathbf{w}_i(t+1) &= \mathbf{w}_i(t) + \eta(t)y_i(t)[\mathbf{x}_t - \gamma_i\hat{\mathbf{x}}_t], \\ \hat{\mathbf{x}}_t &= \mathbf{W}\mathbf{y}, \end{aligned} \tag{13}$$

for $i = 1, \ldots, J_2$, where the coefficients γ_i satisfy $0 < \gamma_1 < \gamma_2 < \cdots < \gamma_{J_2}$. In the weighted SLA, $\mathbf{w}_i$ almost surely converges to the eigenvectors of $\mathbf{C}$. The weighted SLA can perform PCA; however, norms of the weight vectors are not equal to unity.

SLA and weighted SLA are nonlocal algorithms, and they rely on the calculation of the errors and the backward propagation of the values between the layers. A PCA algorithm is obtained by adding a term to SLA [7] so as to rotate the basis vectors in the principal subspace toward the principal eigenvectors [24]. The adaptive learning algorithm (ALA) [20] is also a PCA algorithm that is based on SLA. In ALA, each neuron adaptively updates its learning rate by $\eta_i(t) = \beta(t)/\widehat{\lambda}_i(t)$, where $\widehat{\lambda}_i(t)$ is the estimated eigenvalue and can be estimated using (12), $\beta(t)$ is set to be smaller than $2(\sqrt{2}-1)$ and decreases to zero as $t \to \infty$. All $\mathbf{w}_i(t)$ will quickly converge, at nearly the same rate, to $\mathbf{c}_i$ in the order of descending eigenvalues. The performance is better than that of the generalized Hebbian algorithm (GHA) [8]. SLA has been extended in [25] so as to extract a noise robust projection.

4.2. Generalized Hebbian Algorithm. By combining Oja's rule and the GSO procedure, Sanger proposed GHA for extracting the first J_2 PCs [8]. GHA can extract the first J_2 eigenvectors in the order of decreasing eigenvalues.

The GHA is given by [8]

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \eta_i(t) y_i(t) [\mathbf{x}_t - \widehat{\mathbf{x}}_i(t)], \tag{14}$$

$$\widehat{\mathbf{x}}_i(t) = \sum_{j=1}^{i} \mathbf{w}_j(t) y_j(t), \tag{15}$$

for $i = 1, \ldots, J_2$. GHA becomes a local algorithm by rewriting the summation term in (15) in a recursive form

$$\widehat{\mathbf{x}}_i(t) = \widehat{\mathbf{x}}_{i-1}(t) + \mathbf{w}_i(t) y_i(t), \quad i = 1, \ldots, J_2, \tag{16}$$

where $\widehat{\mathbf{x}}_0(t) = 0$. Usually $\eta_i(t)$ is selected as the same for all neurons. In GHA, the mth neuron converges to the mth PC, and all neurons tend to converge together. $\mathbf{w}_i \to \mathbf{c}_i$ and $E[y_i^2] \to \lambda_i$, as $t \to \infty$.

Both SLA [7, 22] and GHA [8] employ implicit or explicit GSO to decorrelate the connection weights. The weighted SLA [22] performs well for extracting less dominant components.

4.3. Other Hebbian Rule-Based Algorithms. In addition to the popular SLA, weighted SLA and GHA algorithm, there are some other Hebbian rule-based PCA algorithms such as local LEAP (learning machine for adaptive feature extraction via principal component analysis) [26], the nonlocal dot-product-decorrelation (DPD) rule [27], and the local invariant-norm PCA [28].

The LEAP algorithm [26] is a local PCA algorithm for extracting all the J_2 PCs and their corresponding eigenvectors. Unlike SLA [7] and GHA [8], whose stability analysis is based on the stochastic approximation theory [6], the stability analysis of LEAP is based on Lyapunov's first theorem, and as such η can be selected as a small positive constant [26]. Due to a constant learning rate, LEAP is capable of tracking nonstationary processes. LEAP can satisfactorily extract PCs even for ill-conditioned autocorrelation matrices [26].

The DPD algorithm is a nonlocal PCA algorithm [27]. It moves $\mathbf{w}_i$, $i = 1, \ldots, J_2$, towards the J_2 principal eigenvectors $\mathbf{c}_i$, ordered arbitrarily. The algorithm induces the norms of the weight vectors towards the corresponding eigenvalues, that is, $\|\mathbf{w}_i(t)\| \to \lambda_i(t)$, as $t \to \infty$. The algorithm breaks the symmetry in its learning process by the difference in the norms of the weight vectors while keeping the symmetry in its structure. The algorithm is as fast as the GHA [8], weighted SLA [22], and least mean squared error reconstruction (LMSER) [29] algorithms.

5. Least Mean Squared Error-Based Principal Component Analysis

Existing PCA algorithms including the Hebbian rule-based algorithms can be derived by optimizing an objective function using the gradient-descent method. The least mean squared error- (LMSE-) based methods are derived from the modified MSE function

$$E(\mathbf{W}) = \sum_{t_1=1}^{t} \mu^{t-t_1} \left\| \mathbf{x}_{t_1} - \mathbf{W}\mathbf{W}^T \mathbf{x}_{t_1} \right\|^2, \tag{17}$$

where $0 < \mu \leq 1$ is a forgetting factor used for nonstationary observation sequences, and t is the current instant. Many adaptive PCA algorithms actually optimize (17) by using the gradient-descent method [29, 30] and the RLS method [30–34].

The gradient-descent or Hebbian rule-based algorithms are highly sensitive to η. RLS-based algorithms such as adaptive principal components extraction (APEX) [35], Kalman-type RLS [31], projection approximation subspace tracking (PAST) [30], PAST with deflation (PASTd) [30], and the robust RLS algorithm (RRLSA) [33] can overcome the drawback. All RLS-based PCA algorithms exhibit fast convergence and high tracking accuracy and are suitable for slowly varying nonstationary vector stochastic processes. However, RLS method may cause instability in certain cases. All these algorithms correspond to a three-layer J_1-J_2-J_1 linear autoassociative network model, and they can extract all the J_2 PCs in the descending order of the eigenvalues, where a GSO-like orthonormalization procedure is used.

In [36], a regularization term $\mu^t \vec{\mathbf{w}}^T \mathbf{P}_0^{-1} \vec{\mathbf{w}}$ is added to (17), where $\vec{\mathbf{w}}$ is a stack vector of $\mathbf{W}$ and $\mathbf{P}_0$ is a diagonal matrix with dimension $J_1 J_2 \times J_1 J_2$. As t is sufficiently large, this term is negligible. This term ensures that the entries of $\mathbf{W}$ do not become too large. The Gauss-Seidel recursive PCA and Jacobi recursive PCA algorithms are derived in [36].

The LMSER algorithm is derived on the MSE criterion using the gradient-descent method [29]. LMSER reduces to Oja's SLA algorithm when $\mathbf{W}(t)$ is orthonormal, namely, $\mathbf{W}^T(t)\mathbf{W}(t) = \mathbf{I}$. Oja's algorithm can thus be treated as an approximate stochastic gradient rule to minimize the MSE. LMSER [29] has been compared with the weighted SLA [22] and GHA [8] in [37]. LMSER [29] uses nearly twice as much computation as weighted SLA [22] and GHA [8], for each update of the weight. However, it leads to a smaller

asymptotic MSE and faster convergence for the minor eigenvectors [37].

PASTd [30] is a well-known subspace tracking algorithm updating the signal eigenvectors and eigenvalues. PASTd is based on PAST. Both PAST and PASTd are derived for complex-valued signals. Both PAST and PASTd have linear computational complexity, that is, $O(J_1J_2)$ operations every update, as in the cases of SLA [14], GHA [8], LMSER [29], and novel information criterion (NIC) [32]. PAST computes an arbitrary basis of the signal subspace, while PASTd is able to update the signal eigenvectors and eigenvalues. Both the algorithms produce nearly orthonormal, but not exactly orthonormal, subspace basis or eigenvector estimates. If perfectly orthonormal eigenvector estimates are required, an orthonormalization procedure is necessary.

Kalman-type RLS [31] combines the basic RLS algorithm with the GSO procedure. Kalman-type RLS and PASTd are exactly identical if the inverse of the covariance of the output of the ith neuron in Kalman-type RLSA takes a special value. In the one-unit case, both PAST and PASTd reduce to Oja's learning rule [14]. Both PAST and PASTd provide much more robust estimates than eigenvalue decomposition (EVD) and converge much faster than SLA [14]. PASTd has been extended for tracking both the rank and the subspace by using some information theoretic criteria [38].

RRLSA [33] is more robust than PASTd [30]. RRLSA can be implemented in a sequential or parallel form. RRLSA has the flexibility as Kalman-type RLS [31], PASTd [30], APEX [35] in that increasing the number of neurons does not affect the previously extracted principal components. RRLSA naturally selects the inverse of the output energy to be the adaptive learning rate for the Hebbian rule. The Hebbian and Oja rules are closely related to the RRLSA algorithm by suitable selection of the learning rates [33]. RRLSA [33] is also robust to the error accumulation from the previous components, which exists in the sequential PCA algorithms like Kalman-type RLS [31] and PASTd [30]. RRLSA converges fast, even if the eigenvalues extend over several orders of magnitude. According to the empirical results [33], RRLSA provides the best performance in terms of the convergence speed as well as the steady-state error, whereas Kalman-type RLS and PASTd have similar performance, which is inferior to that of RRLSA, and ALA [20] exhibits the poorest performance.

6. Other Optimization-Based Principal Component Analysis

PCA can be derived by any optimization method based on a proper objective function. This leads to many other algorithms, including gradient-descent based algorithms [15, 16, 39, 40], the conjugate gradient (CG) method [41], and the quasi-Newton method [42, 43]. The gradient-descent method usually converges to a local minimum. Some adaptive algorithms derived from the gradient descent, conjugate direction, and Newton-Raphson methods, whose simulation results are better than that of the gradient-descent method [29], have also been proposed in [44]. Second-order algorithms such as the CG [41] and quasi-Newton methods [42] typically converge much faster than first-order methods but have a computational complexity of $O(J_1^2J_2)$ per iteration.

The infomax principle [16] was first proposed by Linsker to describe a neural network algorithm. The principal subspace is derived by maximizing the mutual information criterion.

The NIC algorithm [32] is obtained by applying the gradient-descent method to maximize the NIC, a cost function that is very similar to the mutual information criterion [16, 45] but integrates a soft constraint on the weight orthogonalization. The NIC has a steep landscape along the trajectory from a small weight matrix to the optimum one. E_{NIC} has a single global maximum, and all the other stationary points are unstable saddle points. At the global maximum, $\mathbf{W}$ yields an arbitrary orthonormal basis of the principal subspace. The NIC algorithm has a computational complexity of $O(J_1^2J_2)$ for each iteration.

The NIC algorithm is a PSA method. It can extract the principal eigenvectors when the deflation technique is incorporated. The NIC algorithm converges much faster than SLA [22] and LMSER [29] and is able to globally converge to the PSA solution from almost any weight initialization. Reorthonormalization can be applied so as to perform true PCA [30, 32]. An RLS version of the NIC algorithm is given in [32]. The PAST algorithm [30] is a special case of the NIC algorithm when η takes unity. The weighted information criterion (WINC) [34] is obtained by adding to the NIC a weight to break the symmetry in the NIC. Two WINC algorithms are derived by using gradient ascent and RLS, respectively. The gradient ascent-based WINC algorithm can be viewed be a weighted SLA [23] with an adaptive step size, leading to a much faster convergence speed. The RLS-based WINC algorithm not only provides fast convergence and high accuracy but also has low computational complexity.

Most popular PCA or MCA algorithms do not consider eigenvalue estimates in the update equations of the weights, and they suffer from the stability-speed problem. The convergence speed of a system depends on the eigenvalues of its Jacobian. In PCA algorithms, the eigenmotion depends on the principal eigenvalue of the covariance matrix, while in MCA algorithms on all the eigenvalues [46]. Coupled learning rules can be derived by applying the Newton method to a common information criterion. In coupled PCA/MCA algorithms [46], both the eigenvalues and the eigenvectors are simultaneously adapted. The Newton method yields averaged systems with identical speed of convergence in all eigendirections.

In order to extract multiple PCs, one has to apply an orthonormalization procedure, like GSO, or its first-order approximation as used in SLA [7, 22], or deflation as in GHA [8]. In the coupled learning rules, multiple PCs are simultaneously estimated by a coupled system of equations. In the coupled learning rules a first-order approximation of GSO is superior to the standard deflation procedure in terms of the orthonormality error and the quality of the eigenvectors and eigenvalues generated [47].

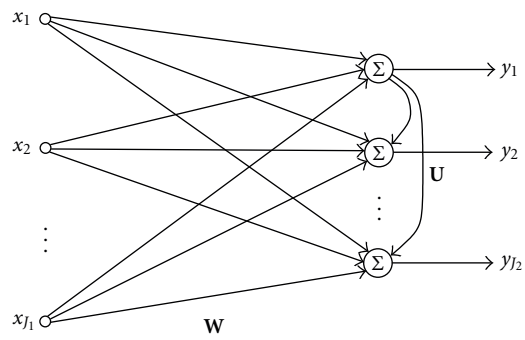

Figure 2: Architecture of the PCA network with hierarchical lateral connections. The lateral weight matrix **U** is an upper triangular matrix with zero diagonal elements.

7. Anti-Hebbian Rule-Based Principal Component Analysis

The anti-Hebbian learning rule updates a synaptic weight by the same amount as that in the Hebbian rule [9] but in the opposite direction. The anti-Hebbian rule can be used to remove correlations between units receiving correlated inputs [13, 48, 49]. The anti-Hebbian rule is inherently stable [13, 49].

Anti-Hebbian rule-based PCA algorithms can be derived by using a J_1-J_2 feedforward architecture with lateral connections among the output units [13, 48, 49]. The lateral connections can be in a symmetrical or hierarchical topology. The hierarchical lateral connection topology is illustrated in Figure 2, based on which Rubner-Tavan PCA [13, 49] and APEX [50] algorithms are proposed. In [48], the local PCA algorithm is based on a symmetrical lateral connection topology. In addition to the feedforward weights **W**, the lateral weight matrix $\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_{J_2}]$ is a $J_2 \times J_2$ matrix, where $\mathbf{u}_i = (u_{1i}, u_{2i}, \ldots, u_{J_2 i})^T$ contains all the lateral weights connected to neuron i and u_{ji} denotes the lateral weight from unit j to unit i.

The Rubner-Tavan PCA algorithm is based on the PCA network with hierarchical lateral connection topology [13, 49]. The algorithm extracts the first J_2 PCs in the decreasing order of the eigenvalues.

The weights $\mathbf{w}_i$ are trained by Oja's rule, while the lateral weights $\mathbf{u}_i$ are updated by the anti-Hebbian rule. This is a nonlocal algorithm. During the training process, the outputs of the neurons are gradually uncorrelated and the lateral weights approach zero. The network should be trained until the lateral weights $\mathbf{u}_i$ are below a specified level. The PCA algorithm proposed in [48] has the same form as Rubner-Tavan PCA, but **U** is a full matrix.

The APEX algorithm is used to adaptively extract the PCs [50]. The algorithm is recursive and adaptive, namely, given $i - 1$ PCs, it can produce the ith PC iteratively. The hierarchical structure of lateral connections serves the purpose of weight orthogonalization and also allows the network to grow or shrink without retraining the old units. APEX is proved to have the property of exponential convergence [50].

Assuming that the correlation matrix **C** has distinct eigenvalues arranged in the decreasing order as $\lambda_1 > \lambda_2 > \cdots > \lambda_{J_2}$ with the corresponding eigenvectors $\mathbf{w}_1, \ldots, \mathbf{w}_{J_2}$, the algorithm is given as [35, 50]

$$\begin{aligned} \mathbf{y} &= \mathbf{W}^T \mathbf{x}, \\ y_i &= \mathbf{w}_i^T \mathbf{x} + \mathbf{u}^T \mathbf{y}, \end{aligned} \tag{18}$$

where $\mathbf{y} = (y_1, \ldots, y_{i-1})^T$ is the output vector, $\mathbf{u} = (u_{1i}, u_{2i}, \ldots, u_{(i-1)i})^T$, and $\mathbf{W} = [\mathbf{w}_1 \cdots \mathbf{w}_{i-1}]$ is the weight matrix of the first $i - 1$ neurons. The iteration is given as [35, 50]

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \eta_i(t)\left[y_i(t)\mathbf{x}_t - y_i^2(t)\mathbf{w}_i(t)\right], \tag{19}$$

$$\mathbf{u}(t+1) = \mathbf{u}(t) - \eta_i(k)\left[y_i(t)\mathbf{y}(t) + y_i^2(t)\mathbf{u}(t)\right]. \tag{20}$$

Equation (19) is the Hebbian part, and (20) the anti-Hebbian part. y_i tends to be orthogonal to all the previous components due to the anti-Hebbian rule, also called orthogonalization rule. APEX can also be derived from the RLS method using the MSE criterion. Based on the RLS method, an optimum learning rate in terms of convergence speed is given by $\eta_i(t) = (1 - \mu)/\lambda_i$, where $0 < \mu \leq 1$ is a forgetting factor [35].

A desirable number of neurons can be decided during the learning process. When the environment is changing over time, a new PC can be added to compensate for the change without affecting the previously computed PCs, and the network structure can be expanded if necessary. $\mathbf{w}_i$ converges to the eigenvector of the correlation matrix **C** corresponding to the ith largest eigenvalue, and **u** converges to zero.

For growing each additional PC, APEX requires $O(J_1)$ per iteration, while GHA requires $O(J_1 J_2)$ per iteration.

A class of learning algorithms, called ψ-APEX, is presented based on a criterion optimization [51, 52]. ψ can be selected as an arbitrary function that guarantees the stability of the network. Some members in the class have better numerical performance and require less computational effort compared to that of both GHA and APEX.

Most existing linear complexity methods including GHA [8], SLA [7], and PCA with the lateral connections [13, 35, 48–50] require a computational complexity of $O(J_1 J_2)$ per iteration.

8. Nonlinear Principal Component Analysis

PCA is based on the Gaussian assumption for data distribution, and the optimality of PCA results from taking into account only the second-order statistics. For non-Gaussian data distributions, PCA is not able to capture complex nonlinear correlations, and nonlinear processing of the data is usually more efficient. Nonlinearities introduce higher-order statistics into the computation in an implicit way. Higher-order statistics, defined by cumulants, are needed for a good characterization of non-Gaussian data. PCA can be generalized to distributions of the exponential family [53].

When the feature space is nonlinearly related to the input space, we need to use nonlinear PCA. The outputs

of nonlinear PCA networks are usually more independent than their respective linear cases. For non-Gaussian input data, a nonlinear PCA permits the extraction of higher-order components and provides a sufficient representation.

Kernel PCA [3, 54] is a special, linear algebra-based nonlinear PCA, which introduces kernel functions into PCA. The kernel PCA first maps the original input data into a high-dimensional feature space using the kernel method and then calculates PCA in the high-dimensional feature space. It is much more complicated and may sometimes be caught more easily in local minima. PCA needs to deal with an eigenvalue problem of a $J_1 \times J_1$ matrix, while kernel PCA needs to solve an eigenvalue problem of an $N \times N$ matrix. Sparse approximation methods can be applied to reduce the computational cost [3].

In order to increase the robustness of PCA against outliers, a robust version of the covariance matrix based on the M-estimator can be used. Several popular PCA algorithms have been generalized into robust versions by applying statistical physics approach [55], where the defined objective function can be regarded as a soft generalization of an M-estimator [56]. Robust PCA can be defined so that the optimization criterion grows less than quadratically with the same constraint conditions as those for PCA, which are based on the quadratic criterion [57]. This usually leads to mildly nonlinear algorithms, in which the nonlinearities appear at selected places only and at least one neuron produces the linear response $y_i = \mathbf{x}^T\mathbf{w}_i$. When all the neurons generate nonlinear responses $y_i = \phi(\mathbf{x}^T\mathbf{w}_i)$, the algorithm is referred to as nonlinear PCA. The robust or nonlinear PCA algorithms are derived by using the gradient-descent method [57]. They can be treated as generalization of SLA [7, 22] and the GHA [8]. Robust and nonlinear PCA algorithms have better stability properties than the corresponding PCA algorithms if the (odd) nonlinearity $\varphi(x)$ grows less than linearly, namely, $|\varphi(x)| < |x|$ [57]. On the contrary, nonlinearities growing faster than linearly cause stability problems easily and are not recommended. These extensions are also introduced in [29, 58, 59].

The multilayer perceptron (MLP) can be used to perform nonlinear dimensionality reduction and hence nonlinear PCA. Both the input and output layers of the MLP have J_1 units, and one of its hidden units, known as the bottleneck or representation layer, have J_2 units, $J_2 < J_1$. The network is trained to reproduce its input vectors themselves. This kind of networks is called the autoassociative network. After the network is trained, it performs a projection onto the J_2-dimensional subspace spanned by the first J_2 PCs of the data. The vectors of weights leading to the hidden units form a basis set which spans the principal subspace, and data compression therefore occurs in the bottleneck layer. Many applications of the autoassociative MLP for PCA are available in the literature [60–63].

Kramer's nonlinear PCA network [62] is a five-layer autoassociative MLP. It has J_1 input and J_1 output nodes. The third layer has J_2 nodes. Nonlinear activation functions such as the sigmoidal functions are used in the second and fourth layers, while the nodes in the bottleneck and output layers usually have linear activation functions, although they can be nonlinear. The network is trained by the backpropagation (BP) algorithm [10, 64]. Kramer's nonlinear PCA fits a lower-dimensional surface through the training data. Usually, the data compression achieved in the bottleneck layer in such networks is somewhat better than that provided by the PCA solution [65]. However, the BP algorithm is prone to local minima and often requires excessive time for convergence.

A hierarchical nonlinear PCA network composed of a number of independent subnetworks can extract ordered nonlinear PCs [66]. Each subnetwork extracts one PC and has at least five layers. The subnetworks can be selected as Kramer's nonlinear PCA network and are hierarchically arranged and trained. This network constructs the extraction functions in the order of the reconstruction efficiency as to the objective data. The number of PCs to be extracted is not required to be known in advance.

A hybrid hetero/autoassociative network [67] is constructed with a set of autoassociative outputs and a set of heteroassociative outputs. Both sets of output nodes are fully connected to the same bottleneck layer. The improvement over an autoassociative network or the PCA can be attributed to the reorganization done by the network in the representation layer space. The self-organizing map (SOM) [68] is a competitive learning-based neural network. It is capable of performing dimensionality reduction on the input. The SOM is inherently nonlinear and is viewed as a nonlinear PCA [69]. The adaptive subspace SOM (ASSOM) [70, 71] can be treated as a hybrid of VQ and PCA.

9. Minor Component Analysis

MCA, as a variant of PCA, is to find the smallest eigenvalues and their corresponding eigenvectors of the autocorrelation matrix $\mathbf{C}$ of the signals. MCA is closely associated with the curve and surface fitting under the total least squares (TLSs) criterion [72]. MCA provides an alternative solution to the TLS problem [73]. The TLS technique achieves a better global optimal objective than the LS technique [1]. Both the TLS and LS problems can be solved by SVD. However, the TLS technique is computationally much more expensive than the least squares (LSs) technique [74]. MCA is useful in many fields including spectrum estimation, optimization, TLS parameter estimation in adaptive signal processing, and eigen-based bearing estimation.

The anti-Hebbian learning rule and its normalized version can be used for MCA [75], but both may lead to infinite magnitudes of weights [76]. To avoid this, one can renormalize the weight vector at each iteration. The constrained anti-Hebbian learning algorithm [73, 77] has a simple structure and requires a low computational complexity per update. However, the convergence of the magnitudes of the weights cannot be guaranteed either unless the initial weights take special values. The total least mean squares (TLMS) algorithm [74] is a random adaptive algorithm for extracting the MC, which has an equilibrium point under persistent excitation conditions. The TLMS requires about $4J_1$ multiplications per iteration, which is twice the complexity of the LMS [78]. An adaptive step-size learning

algorithm [79] has been derived for extracting the MC by introducing information criterion. The algorithm globally converges asymptotically to the MC and its corresponding eigenvector. The algorithm outperforms the TLMS [74] in terms of both the convergence speed and the estimation accuracy.

Minor components (MCs) can be extracted in ways similar to that for PCs. A simple idea is to reverse the sign of the PCA algorithms. This is because in many algorithms PCs correspond to the maximum of a cost function, while MCs correspond to the minimum of the same cost function. However, this idea does not work in general and has been discussed in [22]. Oja's minor subspace analysis (MSA) algorithm can be formulated by reversing the sign of the learning rate of the SLA. This algorithm requires the assumption that the smallest eigenvalue of the autocorrelation matrix $\mathbf{C}$ is less than unity. However, Oja's MSA algorithm is known to diverge [22, 80, 81]. The bigradient PSA algorithm [82] is a modification to SLA [7] and is obtained by introducing an additional bigradient term embodying the orthonormal constraints of the weights, and it can be used for MSA by reversing the sign of η.

A general algorithm that can extract, in parallel, principal and minor eigenvectors of arbitrary dimensions is derived based on the natural-gradient method [83]. The difference between PCA and MCA lies in the sign of the learning rate. The MCA algorithm proposed in [83] suffers from a marginal instability, and thus it requires intermittent normalization such that $\|\mathbf{w}_i\| = 1$ [80]. A self-stabilizing MCA algorithm has been proposed in [80], such that none of $\|\mathbf{w}_i(t)\|$ deviates significantly from unity. It diverges for PCA when $-\eta$ is changed to $+\eta$.

The orthogonal Oja (OOja) algorithm consists of Oja's MSA [7] plus an orthogonalization of $\mathbf{W}(t)$ at each iteration [84], $\mathbf{W}^T(t)\mathbf{W}(t) = \mathbf{I}$. In this case, the above algorithms given in [7, 80, 83] are equivalent. The OOja is numerically very stable. By reversing the sign of η, we extract J_2 PCs. The normalized Oja (NOja) is derived by optimizing the MSE criterion subject to an approximation to the orthonormal constraint [81]. This leads to the optimal learning rate. The normalized orthogonal Oja (NOOja) is an orthogonal version of the NOja such that the orthonormal constraint is perfectly satisfied [81]. Both algorithms offer, as compared to Oja's SLA, a faster convergence, orthogonality, and a better numerical stability with a slight increase in the computational complexity. By switching the sign of η in given learning algorithms, both the NOja and the NOOja can be used for the estimation of minor and principal subspaces of a vector sequence.

The above algorithms including Oja's MSA [7], the natural-gradient-based method [83], self-stabilizing MCA [80], OOja, NOja, and NOOja have a complexity of $O(J_1J_2)$ [80, 84]. OOja, NOjia, and NOOjia require less computation load than the natural-gradient-based method [83], self-stabilizing MCA [80, 81, 84].

By using the Rayleigh quotient as an energy function, invariant-norm MCA [85] is analytically proved to converge to the first MC of the input signals. However, invariant-norm MCA [85] leads to divergence in finite time [76], and this drawback can be eliminated by renormalizing the weight vector at each iteration. In [86], an MCA algorithm for extracting multiple MCs is described by using the idea of sequential addition; a conversion method between MCA and PCA is also discussed. Based on a generalized differential equation for the generalized eigenvalue problem, a class of algorithms can be obtained for extracting the first PC or MC by selecting different parameters and functions [87]. Many existing PCA and MCA algorithms are special cases of this class. All the algorithms of this class have the same order of convergence speed and are robust to implementation error. A rapidly convergent quasi-Newton method has been applied to extract multiple MCs in [88]. The proposed algorithm has a complexity of $O(J_1^2J_2)$ but with a quadratic convergence. It makes use of the implicit orthogonalization procedure that is built into it through an inflation technique.

10. Localized Principal Component Analysis

The nonlinear PCA problem can be solved by partitioning the data space into a number of disjunctive regions and then estimating the principal subspace within each partition by linear PCA. This is the so-called localized PCA. The distribution is collectively modeled by a collection or a mixture of linear PCA models, each characterizing a partition. Most natural data sets have large eigenvalues in only a few eigendirections, while the variances in other eigendirections are so small as to be considered as noise. The localized PCA method provides an efficient means to decompose high-dimensional data compression problems into low-dimensional data compression problems. The localized PCA method is commonly used in image compression [8]. An image is often first transformation coded by PCA, and then the coefficients are quantized.

VQ-PCA [65] is a locally linear model that uses VQ to define the Voronoi regions for localized PCA. The algorithm builds a piecewise linear model of the data. It performs better than the global models implemented by PCA model and Kramer's nonlinear PCA and is significantly faster than Kramer's nonlinear PCA [65]. Adaptive combination of PCA and VQ is given in [89], where an autoassociative network is used to perform PCA and simple competitive learning is used to perform VQ. The error between the input and output of the autoassociative network is fed to the VQ network. The network produces better results than by using the two algorithms successively. An online localized PCA algorithm [90] is developed by extending the neural gas method [91]. Instead of the Euclidean distance measure, a combination of a normalized Mahalanobis distance and the squared reconstruction error guides the competition between the units. Weighting between the two measures is determined from the residual variance in the minor subspace of each submodel. The unit centers are updated as in neural gas, while subspace learning is based on the RRLSA algorithm [33].

Similar to localized PCA, localized ICA is used to characterize nonlinear ICA. Clustering is first used for an overall coarse nonlinear representation of the underlying data and linear ICA is then applied in each cluster so as to describe local features of the data [92]. This leads to a

better representation of the data than in linear ICA. ASSOM [93] is another localized PCA for unsupervised extraction of invariant local features from the input data. ASSOM associates a subspace instead of a single weight vector to each node of the SOM. The subspaces in ASSOM can be formed by applying ICA [94].

11. Extending to Complex Domain

Complex PCA is a generalization of PCA in complex-valued data sets [95]. Complex PCA has been widely applied to complex-valued data and two-dimensional vector fields. Complex PCA employs the same neural network architecture as that of PCA, but with complex weights. The objective functions for PCA can also be adapted to complex PCA by changing the transpose into the Hermitian transpose. For example, for complex PCA, complex PCs can be extracted by minimizing the MSE function

$$E = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{z}_i - \mathbf{W}\mathbf{W}^H\mathbf{z}_i\right\|^2, \tag{21}$$

where $\mathbf{z}_i$ is the ith input complex vector.

In [96], a complex-valued neural network model is developed for nonlinear complex PCA. Nonlinear complex PCA has the ability to extract nonlinear features missed by PCA. It uses the architecture of Kramer's nonlinear PCA network [62], but with complex weights and biases. For a similar number of model parameters, the nonlinear complex PCA model captures more variance of a data set than the alternative real approach, where each complex variable is replaced by two real variables and is applied to Kramer's nonlinear PCA. The complex nonlinear transfer function is selected as $\tanh(z)$ with $|z| < \pi/2$ [97]. To limit the modulus of the net input of a neuron $|\mathbf{w}^H\mathbf{z}|$ to be less than $\pi/2$, one can initialize the algorithm with weights and biases of small magnitudes and use a weight penalty in the objective function. A complex-valued BP or quasi-Newton algorithm can be used for training.

There are many other complex PCA algorithms. Both PAST and PASTd are, respectively, the PSA and PCA algorithms derived for complex-valued signals [30]. A heuristic complex extension of GHA [8] and APEX [35] is, respectively, given in [98, 99]. The robust complex PCA algorithms have also been derived in [100] for hierarchically extracting PCs of complex-valued signals based on a robust statistics-based loss function. Concerning complex MCA, the constrained anti-Hebbian algorithm [73, 77] has been extended for the complex-valued TLS problem [77] and has been applied to adaptive FIR and IIR filtering. The adaptive invariant-norm MCA algorithm [85] has been generalized to the case for complex-valued input signal vector $\mathbf{x}(t)$. For ICA algorithms, FastICA has been applied to complex signals [101]. The ψ-APEX algorithms and GHA are, respectively, extended to the complex-valued case [102, 103]. Based on a suitably selected nonlinear function, these algorithms can be used for BSS of complex-valued circular source signals.

12. Other Generalizations of PCA

Simple neural models, described by differential equations, are derived in [104, 105] to calculate the largest and smallest eigenvalues as well as their corresponding eigenvectors of any real symmetric matrix. Supervised PCA [106, 107] is achieved by augmenting the input of the PCA with the class label of the data set. Given a sample covariance matrix, we examine the problem of maximizing the variance explained by a linear combination of the input variables while constraining the number of nonzero coefficients in this combination. This is known as *sparse PCA* [108]. Constrained PCA, generalized EVD, and the two-dimensional PCA are three important generalizations to PCA.

12.1. Constrained Principal Component Analysis. When certain subspaces are less preferred than others, this yields constrained PCA [109]. The optimality criterion for constrained PCA is variance maximization, as in PCA, but with an external subspace orthogonality constraint that extracted PCs are orthogonal to some undesired subspaces. PCA usually obtains the best fixed-rank approximation to the data in the LS sense. On the other hand, constrained PCA allows specifying metric matrices that modulate the effects of rows and columns of a data matrix. This actually is the weighted LS estimation. Constrained PCA first decomposes the data matrix by projecting the data matrix onto the spaces spanned by matrices of external information and then applies PCA to decomposed matrices, which involves the generalized SVD. The APEX algorithm has been applied to recursively solve the constrained PCA problem [35].

The constrained PAST algorithm [110] is for tracking the signal subspace recursively. Based on an interpretation of the signal subspace as the solution of a constrained minimization task, it guarantees the orthonormality of the estimated signal subspace basis at each update, hence avoiding orthonormalization process. To reduce the computational complexity, fast constrained PAST is introduced which has $O(J_1J_2)$ complexity. A signal subspace rank estimator is employed to track the number of sources.

12.2. Generalized Eigenvalue Decomposition. Generalized EVD is a statistical tool extremely useful in feature extraction, pattern recognition as well as signal estimation and detection. The generalized EVD problem involves the matrix equation

$$\mathbf{R}_1\mathbf{w}_i = \lambda_i\mathbf{R}_2\mathbf{w}_i, \quad i = 1, \ldots, J_2, \tag{22}$$

where $\mathbf{R}_1$, $\mathbf{R}_2 \in R^{J_1\times J_1}$, and λ_i, $\mathbf{w}_i$ are, respectively, the ith generalized eigenvalue and its corresponding generalized eigenvector. For real symmetric and positive definite matrices, all the generalized eigenvectors are real and the corresponding generalized eigenvalues are positive.

Generalized EVD achieves simultaneous diagonalization of $\mathbf{R}_1$ and $\mathbf{R}_2$:

$$\mathbf{W}^T\mathbf{R}_1\mathbf{W} = \vec{\Lambda}, \quad \mathbf{W}^T\mathbf{R}_2\mathbf{W} = \mathbf{I}, \tag{23}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_{J_2}]$ and $\vec{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_{J_2})$. Typically, $\mathbf{R}_1$ and $\mathbf{R}_2$ are, respectively, the full covariance matrices of zero-mean stationary random signals $\mathbf{x}_1$, $\mathbf{x}_2 \in R^{J_1}$. In this case, iterative generalized EVD algorithms can be obtained by using two PCA steps. When $\mathbf{R}_2$ becomes an identity matrix, the generalized EVD reduces to PCA.

Any generalized eigenvector $\mathbf{w}_i$ is a stationary point of the criterion function

$$E_{\text{GEVD}}(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{R}_1 \mathbf{w}}{\mathbf{w}^T \mathbf{R}_2 \mathbf{w}}. \tag{24}$$

The LDA problem is a typical generalized EVD problem. The three-layer LDA network [111] is obtained by concatenating two Rubner-Tavan PCA subnetworks, each being trained by the Rubner-Tavan PCA algorithm [13, 49]. Based on the Rubner-Tavan PCA network architecture, online local learning algorithms for LDA and generalized EVD are given in [112]. There are also a number of adaptive methods for generalized EVD such as LDA-based gradient descent [113, 114], a quasi-Newton type generalized EVD [115], an RLS-like fixed-point generalized EVD algorithm [116], error-correction learning [112], and Hebbian learning [112]. All these algorithms first extract the principal generalized eigenvector and then estimate the minor generalized eigenvectors using a deflation procedure.

A recurrent network with invariant B-norm [117] computes the largest or smallest generalized eigenvalue and the corresponding eigenvector of any symmetric positive pair, which can be simply extended to compute the second largest or smallest generalized eigenvalue and the corresponding eigenvector. In [118], the proposed unconstrained quartic cost function based on the weighted rule has a unique global minimum, which corresponds to the principal generalized eigenvectors.

12.3. Two-Dimensional PCA. Two-dimensional PCA [5] is especially designed for image representation. An image covariance matrix is constructed directly using the original image matrices instead of the transformed vectors, and its eigenvectors are derived for image feature extraction. For $m \times n$ images, the size of the image covariance (scatter) matrix using two-dimensional PCA is $n \times n$, whereas, for PCA, the size is $mn \times mn$. This results in considerable computational advantage in two-dimensional PCA. Two-dimensional PCA evaluates the covariance matrix more accurately over PCA. When used for face recognition, two-dimensional PCA results in a better recognition accuracy. Two-dimensional PCA is a row-based PCA, and it only reflects the information between rows. Diagonal PCA [119] improves two-dimensional PCA by defining the image scatter matrix as the covariances between the variations of the rows and those of the columns of the images and is shown to be more accurate than PCA and two-dimensional PCA. L_1-norm-based two-dimensional PCA [120] is a iterative two-dimensional generalization of L_1-norm based PCA.

Bidirectional PCA [121] reduces the dimension in both column and row directions for image feature extraction. The feature dimension obtained is much less than that of two-dimensional PCA. Two-dimensional PCA can be regarded as a special bidirectional PCA. Bidirectional PCA has to be performed in batch mode. With the concepts of tensor, k-mode unfolding and matricization, an SVD-revision-based incremental learning method of bidirectional PCA [122] gives a close approximation to bidirectional PCA, but using less time.

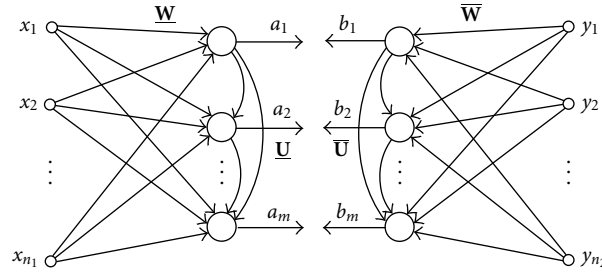

Figure 3: Architecture of the cross-correlation APCA network. The APCA network is composed of two hierarchical PCA networks. The connections with solid arrows denote feedforward connections, and the connections with hollow arrows denote lateral connections.

The uncorrelated multilinear PCA algorithm [123] is used for unsupervised subspace learning of tensorial data. It is a multilinear extension of PCA. Through successive variance maximization, uncorrelated multilinear PCA seeks a tensor-to-vector projection that captures most of the variation in the original tensorial input while producing uncorrelated features. It is the only multilinear extension of PCA that can produce uncorrelated features in a fashion similar to that of PCA, in contrast to other multilinear PCA extensions, such as two-dimensional PCA [5] and multilinear PCA (MPCA) [124].

13. Singular Value Decomposition

Given two sets of random vectors with zero mean, $\{\mathbf{x}_t \in R^{n_1}\}$ and $\{\mathbf{y}_t \in R^{n_2}\}$, the cross-correlation matrix is defined by

$$\mathbf{C}_{xy} = E\left[\mathbf{x}_t \mathbf{y}_t^T\right] = \sum_{i=1}^{n} \sigma_i \mathbf{v}_i^x \left(\mathbf{v}_i^y\right)^T, \tag{25}$$

where $\sigma_i > 0$ is the ith singular value, $\mathbf{v}_i^x$ and $\mathbf{v}_i^y$ are its corresponding left and right singular vectors, and $n = \min\{n_1, n_2\}$. The cross-correlation asymmetric PCA/MCA networks can be used to extract the singular values of the cross-correlation matrix of two stochastic signal vectors or to implement SVD of a general matrix.

Cross-correlation asymmetric PCA (APCA) network consists of two sets of neurons that are laterally hierarchically connected [125]. The topology of the network is shown in Figure 3. $\mathbf{x}$ and $\mathbf{y}$ are, respectively, the n_1-dimensional and n_2-dimensional input signals, the $n_1 \times m$ matrix $\underline{\mathbf{W}} = [\underline{\mathbf{w}}_1 \cdots \underline{\mathbf{w}}_m]$ and the $n_2 \times m$ matrix $\overline{\mathbf{W}} = [\overline{\mathbf{w}}_1 \cdots \overline{\mathbf{w}}_m]$ are the feedforward weights, while the $n_2 \times m$ matrices $\underline{\mathbf{U}} = [\underline{\mathbf{u}}_1 \cdots \underline{\mathbf{u}}_m]$ and $\overline{\mathbf{U}} = [\overline{\mathbf{u}}_1 \cdots \overline{\mathbf{u}}_m]$ are the lateral connection weights, where $\underline{\mathbf{u}}_i = (\underline{u}_{1i}, \ldots, \underline{u}_{mi})^T$, $\overline{\mathbf{u}}_i = (\overline{u}_{1i}, \ldots, \overline{u}_{mi})^T$, and $m \leq \min\{n_1, n_2\}$. This model performs SVD of $\mathbf{C}_{xy}$ [125].

The network has the following relations:

$$\mathbf{a} = \underline{\mathbf{W}}^T\mathbf{x}, \qquad \mathbf{b} = \overline{\mathbf{W}}^T\mathbf{y}, \tag{26}$$

where $\mathbf{a} = (a_1, \ldots, a_m)^T$ and $\mathbf{b} = (b_1, \ldots, b_m)^T$.

The objective function for extracting the first principal singular value of the covariance matrix is given by

$$E_{\text{APCA}}(\underline{\mathbf{w}}, \overline{\mathbf{w}}) = \frac{E[a_1(t)b_1(t)]}{||\underline{\mathbf{w}}||||\overline{\mathbf{w}}||} = \frac{\underline{\mathbf{w}}^T\mathbf{C}_{xy}\overline{\mathbf{w}}}{||\underline{\mathbf{w}}||||\overline{\mathbf{w}}||}. \tag{27}$$

It is an indefinite function. When $\mathbf{y} = \mathbf{x}$, it reduces to PCA [14]. After the principal singular component is extracted, a deflation transformation is introduced to nullify the principal singular value so as to make the next singular value principal. Thus, $\mathbf{C}_{xy}$ in the criterion (27) can be replaced by a transformed form so as to extract the next principal singular component.

Using a deflation transformation, the two sets of neurons are trained with the cross-coupled Hebbian learning rules, which are given in [125, 126]. $\underline{\mathbf{w}}_i$ and $\overline{\mathbf{w}}_i$ approximate the ith left and right principal singular vectors of $\mathbf{C}_{xy}$, respectively, and σ_i approximates its corresponding criterion E_{APCA}, as $t \to \infty$. The algorithm extracts the first m principal singular values in the descending order and their corresponding left and right singular vectors. Like APEX, the APCA algorithm incrementally adds nodes without retraining the learned nodes. Exponential convergence has been demonstrated by simulation [125].

Based on the APCA network, the principal singular component of $\mathbf{C}_{xy}$ can be efficiently extracted by using a modification to the cross-coupled Hebbian rule with global asymptotic convergence [127]. This algorithm is extended for extracting the principal singular component of a general matrix $\mathbf{A} \in R^{n_1 \times n_2}$ by replacing the cross-correlation matrix by a general nonsquare matrix [127]. Based on this and a deflation transformation, one can extract multiple principal singular components for the nonsquare matrix $\mathbf{A}$ [127]. The algorithm can efficiently perform SVD of an ill-posed matrix. It can be used to solve the smallest singular component of the general matrix $\mathbf{A}$ and is especially useful for TLS problems. Some adaptive SVD algorithms for subspace tracking of a recursively updated data matrix have been surveyed and proposed in [128].

Coupled learning rules for SVD produce better results than Hebbian learning rules. Combined with first-order approximation of GSO, precise estimates of singular vectors and singular values with only small deviations from orthonormality are produced. Double deflation is clearly superior to standard deflation but inferior to first-order approximation of GSO, both with respect to orthonormality and diagonalization errors. Coupled learning rules converge faster than Hebbian learning rules, and the first-order approximation of GSO produces more precise estimates and better orthonormality than standard deflation [129]. Many SVD algorithms are reviewed in [129].

Tucker decomposition [130] decomposes a three-dimensional signal directly using three-dimensional PCA, which is a multilinear generalization of SVD to multidimensional data. For video frames, this higher-order SVD decomposes the dynamic texture as a multidimensional signal (tensor) without unfolding the video frames on column vectors. This is a more natural and flexible decomposition, since it permits us to perform dimension reduction in the spatial, temporal, and chromatic domains between the pixels of the video sequence, leading to an important decrease in model size, while standard SVD allows for temporal reduction only. The analysis part is more expensive, but the synthesis has the same cost as existing algorithms [131].

14. Canonical Correlation Analysis

CCA [132], proposed by Hotelling in 1936, is a multivariate statistical technique. It makes use of two views of the same set of objects and projects them onto a lower-dimensional space in which they are maximally correlated. CCA seeks prominently correlated projections between two views of data, and it has been long known to be equivalent to LDA when the data features are used in one view and the class labels are used in the other view [133, 134]. In other words, LDA is a special case of CCA. CCA is equivalent to LDA for binary-class problems [134], and it can be formulated as an LS problem for binary-class problems.

CCA leads to a generalized EVD problem. Thus, we can employ a kernelized version of CCA to compute a flexible contrast function for ICA. Generalized CCA consists of a generalization of CCA to more than two sets of variables [135].

Given two centered random multivariables $\mathbf{x} \in \mathcal{R}^{n_x}$ and $\mathbf{y} \in \mathcal{R}^{n_y}$, the goal of CCA is to find a pair of directions $\vec{\omega}_x$ and $\vec{\omega}_y$ such that the correlation $\rho(\mathbf{x}, \mathbf{y})$ between the two projections $\vec{\omega}_x^T\mathbf{x}$ and $\vec{\omega}_y^T\mathbf{y}$ is maximized.

Suppose that we are given a sample of instances $S = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ of $(\mathbf{x}, \mathbf{y})$. Let S_x denote $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and similarly S_y denote $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$. We can consider defining a new coordinate for $\mathbf{x}$ by choosing direction $\mathbf{w}_x$ and projecting $\mathbf{x}$ onto that direction, $\mathbf{x} \to \mathbf{w}_x^T\mathbf{x}$. If we do the same for $\mathbf{y}$ by choosing direction $\mathbf{w}_y$, we obtain a sample of the new mapping for $\mathbf{y}$. Let

$$S_{x,\mathbf{w}_x} = \left(\mathbf{w}_x^T\mathbf{x}_1, \ldots, \mathbf{w}_x^T\mathbf{x}_n\right), \tag{28}$$

with the corresponding values of the mapping for $\mathbf{y}$ being

$$S_{y,\mathbf{w}_y} = \left(\mathbf{w}_y^T\mathbf{y}_1, \ldots, \mathbf{w}_y^T\mathbf{y}_n\right). \tag{29}$$

The first stage of canonical correlation is to choose $\mathbf{w}_x$ and $\mathbf{w}_y$ to maximize the correlation between the two vectors

$$\rho = \max_{\mathbf{w}_x, \mathbf{w}_y} \text{corr}\left(S_{x\mathbf{w}_x}, S_{y\mathbf{w}_y}\right) = \max_{\mathbf{w}_x, \mathbf{w}_y} \frac{\left\langle S_{x\mathbf{w}_x}, S_{y\mathbf{w}_y}\right\rangle}{||S_{x\mathbf{w}_x}||\left|\left|S_{y\mathbf{w}_y}\right|\right|}. \tag{30}$$

After manipulation, we have

$$\begin{aligned}\rho &= \max_{\mathbf{w}_x, \mathbf{w}_y} \frac{\mathbf{w}_x^T E[\mathbf{x}\mathbf{y}^T]\mathbf{w}_y}{\sqrt{\mathbf{w}_x^T E[\mathbf{x}\mathbf{x}^T]\mathbf{w}_x \mathbf{w}_y^T E[\mathbf{y}\mathbf{y}^T]\mathbf{w}_y}} \\ &= \frac{\mathbf{w}_x^T\mathbf{C}_{\mathbf{xy}}\mathbf{w}_y}{\sqrt{\mathbf{w}_x^T\mathbf{C}_{\mathbf{xx}}\mathbf{w}_x\mathbf{w}_y^T\mathbf{C}_{\mathbf{yy}}\mathbf{w}_y}},\end{aligned} \tag{31}$$

where the covariance matrix of $(\mathbf{x}, \mathbf{y})$ is defined by

$$\mathbf{C}(\mathbf{x},\mathbf{y}) = E\left[\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}\begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^T\right] = \begin{bmatrix}\mathbf{C}_{\mathbf{xx}} & \mathbf{C}_{\mathbf{xy}}\\ \mathbf{C}_{\mathbf{yx}} & \mathbf{C}_{\mathbf{yy}}\end{bmatrix} = \mathbf{C}. \quad (32)$$

Under a mild condition which tends to hold for high-dimensional data, CCA in the multilabel case can be formulated as an LS problem [136]. Based on this, efficient algorithms for solving LS problems can be applied to scale CCA to very large data sets. In addition, several CCA extensions, including the sparse CCA formulation based on L_1-norm regularization, are proposed [136]. The LS formulation of CCA and its extensions can be solved efficiently. The LS formulation is extended to orthonormalized partial least squares by establishing the equivalence relationship between CCA and orthonormalized partial least squares [136]. The CCA projection for one set of variables is independent of the regularization on the other set of multidimensional variables.

In [137], a strategy for reducing LDA to CCA is proposed. Within-class coupling CCA (WCCCA) is to apply CCA to pairs of data samples that are most likely to belong to the same class. Each one of the samples of a class, serving as the first view, is paired with every other samples of that class serving as the second view. The equivalence between LDA and such an application of CCA is proved.

Two-dimensional CCA seeks linear correlation based on images directly. Motivated by locality-preserving CCA [138] and spectral clustering, a manifold learning method called local two-dimensional CCA [139] identifies the local correlation by weighting images differently according to their closeness. That is, the correlation is measured locally, which makes local two-dimensional CCA more accurate in finding correlative information. Local two-dimensional CCA is formulated as solving generalized eigenvalue equations tuned by Laplacian matrices.

CCArc [140] is a two-dimensional CCA that is based on representing the image as the sets of its rows and columns and implementation of CCA using these sets. CCArc does not require preliminary downsampling procedure, it is not iterative and it is applied along the rows and columns of input image. Size of covariance matrices in CCArc is equal to $\max\{M, N\}$ Small-sample-size problem in CCArc does not occur, because we actually use N images of size $M \times 1$ and M images of size $N \times 1$; this always meets the condition $\max\{M, N\} < (M + N)$.

15. A Simulation Example

The concept of subspace is involved in many information processing problems. This requires EVD of the autocorrelation matrix of a data set or SVD of the cross-correlation matrix of two data sets. For example, in the area of array signal processing, the APCA algorithm is used in beamforming [126] and the MCA or PCA method is used in DoA estimation [141]. We have also illustrated weighted SLA, GHA, and APEX in [10]. We now provide an example to illustrate the application of PCA.

Image compression is usually implemented by partitioning an image into many nonoverlapping 8×8 pixel blocks and then compressing them one by one. Based on the statistics of all the regions, one can use PCA to compress the image [8, 10]. Each region is concatenated into a vector, and all the vectors constitute a training set. PCA is then applied to extract those prominent PCs, as such the image is compressed. Similar results for image compression have been reported in [11, 142] by using a three-layer autoassociative network with BP learning. PCA as well as LDA achieves the same results for an original data set and its orthonormally transformed version [143]; thus, both methods can be directly implemented in the DCT domain, and the results are exactly the same as that obtained from the spatial domain.

In this example, we use the APEX algorithm to train the PCA network and then use the trained network to encode other pictures. We use the benchmark Lina picture of 560×560 pixels as the training set, and a kid picture of 640×560, which has the similar statistics, is then used for generalization. We first use 8×8 blocks. For each square, only the first PC is significant, and all the other PCs can be ignored in terms of the quality of the restored picture. In this sense, we achieve a compression ratio of 1 : 64. However, the quality of the restored image is very poor. This is because PCA assumes the Guassian statistics of the data set and a real picture usually does not satisfy this assumption.

To improve the quality of the restored image, we employ nonoverlapping 4×4 blocks and again use the first PCs of the blocks. This achieves a compression ratio of 1 : 16. The Lena picture is thus transformed into a training set of 14400 vectors, and the kid picture is transformed into a validation set of 19200 vectors. The training set is so large that the APEX algorithm converges after only two epochs. The obtained codebook is $\mathbf{w}_1 = (0.1949, 0.2222, 0.1920, 0.1747, 0.2255, 0.2065, 0.2195, 0.2246, 0.1972, 0.1701, 0.2268, 0.1908, 0.2224, 0.1996, 0.2140, 0.1783)^T$. The Lena picture and its restored version are shown in Figure 4. The trained network is then used to compress the kid picture. Both the kid picture and its restored version are shown in Figure 5.

16. Summary

In this paper, we have discussed various neural network implementations and algorithms for PCA and its various extensions, including PCA, MCA, generalized EVD, constrained PCA, two-dimensional methods, localized methods, complex-domain methods, and SVD. These neural network methods provide an advantage over their conventional counterparts in that they are adaptive algorithms and have low computational as well as low storage complexity. These neural network methods find wide applications in pattern recognition, blind source separation, adaptive signal processing, and information compression. Two methods that are strongly associated with PCA, namely, ICA and LDA, are described here in passing.

16.1. Independent Component Analysis. ICA [144] is a statistical model that extends PCA. ICA has been widely used for BSS, feature extraction, and signal detection. For BSS

Figure 4: The benchmark Lena picture and its restored version.

Figure 5: The kid picture and its restored version.

applications, the ICA model is required to have model identifiability and separability [144]. The first neural network model related to the ICA was developed for online BSS of linearly mixed signals in [145].

ICA can extract the statistically independent components from the input data set. It is to estimate the mutual information between the signals by adjusting the estimated matrix to give outputs that are maximally independent. By applying ICA to estimate the independent input data from raw data, a statistical test can be derived to reduce the input dimension. The dimensions to remove are those that are independent of the output. In contrast, in PCA the dimensionality reduction is achieved by removing those dimensions that have a low variance.

Let a J_1-vector $\mathbf{x}$ denote a linear mixture and a J_2-vector $\mathbf{s}$, whose components have zero mean and are statistically mutually independent, denote the original source signals. The ICA data model can be written as

$$\mathbf{x} = \mathbf{A}\mathbf{s} + \mathbf{n}, \tag{33}$$

where $\mathbf{A}$ is an unknown constant full-rank $J_1 \times J_2$ mixing matrix and $\mathbf{n}$ denotes the additive noise term, which is often omitted since it is usually impossible to separate noise from the sources. ICA takes one of three forms, namely, square ICA for $J_1 = J_2$, overcomplete ICA for $J_1 < J_2$, and undercomplete ICA for $J_1 > J_2$. While undercomplete ICA is useful for feature extraction, overcomplete ICA may be applied to signal and image processing methods based on multiscale and redundant basis sets.

The goal of ICA is to estimate $\mathbf{s}$ by

$$\mathbf{y} = \mathbf{W}^T\mathbf{x} \tag{34}$$

such that the components of $\mathbf{y}$, which is the estimate of $\mathbf{s}$, are statistically as independent as possible, $\mathbf{W}$ being a $J_1 \times J_2$ demixing matrix. The statistical independence property implies that the joint probability density of the components of $\mathbf{s}$ equals the product of the marginal densities of the individual components. Each component of $\mathbf{s}$ is a stationary stochastic process, and only one of the components

is allowed to be Gaussian distributed. The higher-order statistics of the original inputs are required for estimating $\mathbf{s}$, rather than the second-order moment or covariance of the samples as used in PCA.

The demixing matrix $\mathbf{W}$ of ICA is not orthogonal, while in PCA the components of the weights are represented on an orthonormal basis. ICA provides in many cases a more meaningful representation of the data than PCA. ICA can be realized by adding nonlinearity to linear PCA networks such that they are able to improve the independence of their outputs. In [146], an efficient ICA algorithm is derived by minimizing a nonlinear PCA criterion using the RLS approach. The three-layer (J_1-J_2-J_1) linear autoassociative network can also be used as an ICA network, as long as the outputs of the hidden layer are independent.

A well-known two-phase approach to ICA is to preprocess the data by PCA and then to estimate the necessary rotation matrix. A generic approach to ICA consists of preprocessing the data, defining measures of non-Gaussianity, and optimizing an objective function, known as a contrast function. Some measures of non-Gaussianity are kurtosis, differential entropy, negentropy, and mutual information, which can be derived from one another. Popular ICA algorithms include the Infomax, the natural-gradient, the equivariant adaptive separation via independence (EASI), and the FastICA algorithms [10]. FastICA is a well-known fixed-point ICA algorithm [147]. It is derived from the optimization of the kurtosis or the negentropy measure by using Newton's method. FastICA achieves a reliable and at least a quadratic convergence. FastICA with symmetric orthogonalization and tanh nonlinearity is concluded as the best trade-off for ICA [10].

The ICA methods can be easily extended to the complex domain by using Hermitian transpose and complex nonlinear functions. In the context of BSS, the higher-order statistics are necessary only for temporally uncorrelated stationary sources. Second-order statistics-based source separation exploits temporally correlated stationary sources and the nonstationarity of the sources [148]. Many natural signals are inherently nonstationary with time-varying variances, since the source signals incorporate time delays into the basic BSS model.

Blind separation of the original signals in nonlinear mixtures has many difficulties such as the intrinsic indeterminacy, the unknown distribution of the sources as well as the mixing conditions, and the presence of noise. It is impossible to separate the original sources using only the source independence assumption of some unknown nonlinear transformations of the sources [149]. Nonlinear ICA can be modeled by a parameterized neural network whose parameters can be determined under the criterion of independence of its outputs. The inverse of the nonlinear mixing model can be modeled by using the three-layer MLP, the RBFN, the SOM, or the kernel-based nonlinear BSS method [10].

Nonnegativity is a natural condition for many real-world applications, for example, in the analysis of images, text, or air quality. Neural networks can be suggested by imposing a nonnegativity constraint on the outputs [29] or weights. Nonnegative PCA and nonnegative ICA algorithms are given in [150], where the sources $\mathbf{s}_i$ must be nonnegative. Constrained ICA is a framework that incorporates additional requirements and prior information in form of constraints into the ICA contrast function [151].

16.2. Linear Discriminant Analysis. LDA creates a linear combination of the given independent features that yield the largest mean differences between the desired classes [2]. Given a set of N vectors of J_1 dimensions, $\{\mathbf{x}_i\}$, for all the samples of all the C classes, the within-class scatter matrix $\mathbf{S}_w$, the between-class scatter matrix $\mathbf{S}_b$, and the mixture scatter matrix $\mathbf{S}_m$ are defined. All the scatter matrices are of size $J_1 \times J_1$. The minimization of the MSE criterion is equivalent to the minimization of the trace of $\mathbf{S}_w$ or maximizing the trace of $\mathbf{S}_b$.

The objective for LDA is to maximize the between-class measure while minimizing the within-class measure after applying a $J_1 \times J_2$ transform matrix $\mathbf{W}$, $J_1 > J_2$, which transforms the $J_1 \times J_1$ scatter matrices into $J_2 \times J_2$ matrices $\widetilde{\mathbf{S}}_w$, $\widetilde{\mathbf{S}}_b$, $\widetilde{\mathbf{S}}_m$:

$$\begin{aligned} \widetilde{\mathbf{S}}_w &= \mathbf{W}^T \mathbf{S}_w \mathbf{W}, \\ \widetilde{\mathbf{S}}_b &= \mathbf{W}^T \mathbf{S}_b \mathbf{W}, \\ \widetilde{\mathbf{S}}_m &= \mathbf{W}^T \mathbf{S}_m \mathbf{W}. \end{aligned} \tag{35}$$

The $\mathrm{tr}(\mathbf{S}_w)$ measures the closeness of the samples within the clusters, and $\mathrm{tr}(\mathbf{S}_b)$ measures the separation between the clusters, where $\mathrm{tr}(\cdot)$ denotes the trace operator. An optimal $\mathbf{W}$ should preserve the given cluster structure, and simultaneously maximize $\mathrm{tr}(\widetilde{\mathbf{S}}_b)$ and minimize $\mathrm{tr}(\widetilde{\mathbf{S}}_w)$. Assuming that $\mathbf{S}_w$ is a nonsingular matrix, conventionally, the following Fisher's determinant ratio criterion is maximized for finding the projection directions [152, 153]:

$$E_{\mathrm{LDA}}(\mathbf{W}) = \frac{\det\left(\widetilde{\mathbf{S}}_b\right)}{\det\left(\widetilde{\mathbf{S}}_w\right)} = \frac{\det(\mathbf{W}^T \mathbf{S}_b \mathbf{W})}{\det(\mathbf{W}^T \mathbf{S}_w \mathbf{W})}, \tag{36}$$

where the column vectors $\mathbf{w}_i$, $i = 1, \ldots, J_2$, of $\mathbf{W}$ are the first J_2 principal eigenvectors of $\mathbf{S}_w^{-1}\mathbf{S}_b$. Under the assumption that the class distributions are identically distributed Gaussians, LDA is Bayes optimal [3].

There are at most $C - 1$ nonzero generalized eigenvalues and thus an upper bound on J_2 is $C - 1$; at least $J_1 + C$ samples are needed to guarantee $\mathbf{S}_w$ to be nonsingular. This requirement on the number of samples may be severe for some problems like image processing. In this case, $\mathbf{S}_w$ will be singular and regularization may be necessary. By introducing kernel into $\mathbf{W}$, nonlinear discriminant analysis is obtained [3, 154]. A multiple of the identity or the kernel matrix can be added to $\mathbf{S}_w$ or its reformulated matrix $\widetilde{\mathbf{S}}_w$ after introducing the kernels to penalize $\|\mathbf{w}\|^2$ or $\|\widetilde{\mathbf{w}}\|^2$, respectively, [154]. The high-dimensional features can also be first compressed by PCA into intermediate-dimensional space, which is further projected by LDA onto the low-dimensional space. The overall performance of the two-stage approach is sensitive to the reduced dimension in the first stage. A generalization of

LDA by using generalized SVD [153] can be used to solve the problem of singularity of $\mathbf{S}_w$. For image feature extraction, two-dimensional LDA algorithms [155, 156] provide an efficient approach and can overcome the singularity problem.

In [152], a nonlinear discriminant analysis network with the MLP as the architecture and Fisher's determinant ratio as the criterion function is obtained by combining the universal approximation properties of the MLP with the target-free nature of LDA. A layered lateral network-based LDA network and an MLP-based nonlinear discriminant analysis network are also proposed in [111]. Based on a single-layer linear feedforward network, LDA algorithms are also given in [112, 113].

References

[1] G. H. Golub and C. F. van Loan, *Matrix Computation*, John Hopkins University Press, Baltimore, Md, USA, 2nd edition, 1989.

[2] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, New York, NY, USA, 1973.

[3] K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001.

[4] M. Loeve, *Probability Theory*, Van Nostrand, New York, NY, USA, 3rd edition, 1963.

[5] J. Yang, D. Zhang, A. F. Frangi, and J. Y. Yang, "Two-dimensional PCA: a new approach to appearance-based face representation and recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 131–137, 2004.

[6] L. Ljung, "Analysis of recursive stochastic algorithm," *IEEE Transactions on Automatic Control*, vol. 22, no. 4, pp. 551–575, 1977.

[7] E. Oja and J. Karhunen, "On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix," *Journal of Mathematical Analysis and Applications*, vol. 106, no. 1, pp. 69–84, 1985.

[8] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Networks*, vol. 2, no. 6, pp. 459–473, 1989.

[9] D. O. Hebb, *The Organization of Behavior*, John Wiley & Sons, New York, NY, USA, 1949.

[10] K. L. Du and M. N. S. Swamy, *Networks in a Softcomputing Framework*, Springer, London, UK, 2006.

[11] M. H. Hassoun, *Fundamentals of Artificial Neural Networks*, MIT Press, Cambridge, Mass, USA, 1995.

[12] C. Von Der Malsburg, "Self organization of orientation sensitive cells in the striate cortex," *Kybernetik*, vol. 14, no. 2, pp. 85–100, 1973.

[13] J. Rubner and P. Tavan, "A self-organizing network for principal-component analysis," *Europhysics Letters*, vol. 10, pp. 693–698, 1989.

[14] E. Oja, "A simplified neuron model as a principal component analyzer," *Journal of Mathematical Biology*, vol. 15, no. 3, pp. 267–273, 1982.

[15] A. L. Yuille, D. M. Kammen, and D. S. Cohen, "Quadrature and the development of orientation selective cortical cells by Hebb rules," *Biological Cybernetics*, vol. 61, no. 3, pp. 183–194, 1989.

[16] R. Linsker, "From basic network principles to neural architecture: emergence of orientation-selective cells," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 83, no. 21, pp. 8390–8394, 1986.

[17] P. J. Zufiria, "On the discrete-time dynamics of the basic Hebbian neural-network node," *IEEE Transactions on Neural Networks*, vol. 13, no. 6, pp. 1342–1352, 2002.

[18] Z. Yi, M. Ye, J. C. Lv, and K. K. Tan, "Convergence analysis of a deterministic discrete time system of Oja's PCA learning algorithm," *IEEE Transactions on Neural Networks*, vol. 16, no. 6, pp. 1318–1328, 2005.

[19] A. Cichocki and R. Unbehauen, *Networks for Optimization and Signal Processing*, John Wiley & Sons, New York, NY, USA, 1992.

[20] L. H. Chen and S. Chang, "Adaptive learning algorithm for principal component analysis," *IEEE Transactions on Neural Networks*, vol. 6, no. 5, pp. 1255–1263, 1995.

[21] A. Krogh and J. A. Hertz, "Hebbian learning of principal components," in *Parallel Processing in Neural Systems and Computers*, R. Eckmiller, G. Hartmann, and G. Hauske, Eds., pp. 183–186, North-Holland, Amsterdam, The Netherlands, 1990.

[22] E. Oja, "Principal components, minor components, and linear neural networks," *Neural Networks*, vol. 5, no. 6, pp. 927–935, 1992.

[23] E. Oja, H. Ogawa, and J. Wangviwattana, "Principal component analysis by homogeneous neural networks," *IEICE Transactions on Information and Systems*, vol. E75-D, no. 3, pp. 366–382, 1992.

[24] M. Jankovic and H. Ogawa, "Time-oriented hierarchical method for computation of principal components using subspace learning algorithm," *International Journal of Neural Systems*, vol. 14, no. 5, pp. 313–323, 2004.

[25] A. Weingessel and K. Hornik, "A robust subspace algorithm for principal component analysis," *International Journal of Neural Systems*, vol. 13, no. 5, pp. 307–313, 2003.

[26] H. Chen and R. W. Liu, "On-line unsupervised learning machine for adaptive feature extraction," *IEEE Transactions on Circuits and Systems II*, vol. 41, no. 2, pp. 87–98, 1994.

[27] F. Peper and H. Noda, "A symmetric linear neural network that learns principal components and their variances," *IEEE Transactions on Neural Networks*, vol. 7, no. 4, pp. 1042–1047, 1996.

[28] F. L. Luo, R. Unbehauen, and Y. D. Li, "A principal component analysis algorithm with invariant norm," *Neurocomputing*, vol. 8, no. 2, pp. 213–221, 1995.

[29] L. Xu, "Least mean square error reconstruction principle for self-organizing neural-nets," *Neural Networks*, vol. 6, no. 5, pp. 627–648, 1993.

[30] B. Yang, "Projection approximation subspace tracking," *IEEE Transactions on Signal Processing*, vol. 43, no. 1, pp. 95–107, 1995.

[31] S. Bannour and M. R. Azimi-Sadjadi, "Principal component extraction using recursive least squares learning," *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 457–469, 1995.

[32] Y. Miao and Y. Hua, "Fast subspace tracking and neural network learning by a novel information criterion," *IEEE Transactions on Signal Processing*, vol. 46, no. 7, pp. 1967–1979, 1998.

[33] S. Ouyang, Z. Bao, and G. S. Liao, "Robust recursive least squares learning algorithm for principal component analysis," *IEEE Transactions on Neural Networks*, vol. 11, no. 1, pp. 215–221, 2000.

[34] S. Ouyang and Z. Bao, "Fast principal component extraction by a weighted information criterion," *IEEE Transactions on Signal Processing*, vol. 50, no. 8, pp. 1994–2002, 2002.

[35] S. Y. Kung, K. I. Diamantaras, and J. S. Taur, "Adaptive principal component extraction (APEX) and applications," *IEEE Transactions on Signal Processing*, vol. 42, no. 5, pp. 1202–1271, 1994.

[36] A. C. S. Leung, K. W. Wong, and A. C. Tsoi, "Recursive algorithms for principal component extraction," *Network*, vol. 8, no. 3, pp. 323–334, 1997.

[37] C. Chatterjee, V. P. Roychowdhury, and E. K. P. Chong, "On relative convergence properties of principal component analysis algorithms," *IEEE Transactions on Neural Networks*, vol. 9, no. 2, pp. 319–329, 1998.

[38] B. Yang, "An extension of the PASTd algorithm to both rank and subspace tracking," *IEEE Signal Processing Letters*, vol. 2, no. 9, pp. 179–182, 1995.

[39] Y. Chauvin, "Principal component analysis by gradient descent on a constrained linear Hebbian cell," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '89)*, pp. 373–380, Washington, DC, USA, June 1989.

[40] B. A. Pearlmutter, G. E. Hinton, and G. -maximization:, "An unsupervised learning procedure for discovering regularities," in *Proceedings of the Neural Networks for Computing*, J. S. Denker, Ed., vol. 151, pp. 333–338, American Institute of Physics, Snowbird, Utah, USA, 1986.

[41] Z. Fu and E. M. Dowling, "Conjugate gradient eigenstructure tracking for adaptive spectral estimation," *IEEE Transactions on Signal Processing*, vol. 43, no. 5, pp. 1151–1160, 1995.

[42] Z. Kang, C. Chatterjee, and V. P. Roychowdhury, "An adaptive quasi-newton algorithm for eigensubspace estimation," *IEEE Transactions on Signal Processing*, vol. 48, no. 12, pp. 3328–3333, 2000.

[43] S. Ouyang, P. C. Ching, and T. Lee, "Robust adaptive quasi-Newton algorithms for eigensubspace estimation," *IEE Proceedings*, vol. 150, no. 5, pp. 321–330, 2003.

[44] C. Chatterjee, Z. Kang, and V. P. Roychowdhury, "Algorithms for accelerated convergence of adaptive PCA," *IEEE Transactions on Neural Networks*, vol. 11, no. 2, pp. 338–355, 2000.

[45] M. D. Plumbley, "Lyapunov functions for convergence of principal component algorithms," *Neural Networks*, vol. 8, no. 1, pp. 11–23, 1995.

[46] R. Möller and A. Könies, "Coupled principal component analysis," *IEEE Transactions on Neural Networks*, vol. 15, no. 1, pp. 214–222, 2004.

[47] R. Möller, "First-order approximation of Gram-Schmidt orthonormalization beats deflation in coupled PCA learning rules," *Neurocomputing*, vol. 69, no. 13–15, pp. 1582–1590, 2006.

[48] P. Foldiak, "Adaptive network for optimal linear feature extraction," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '89)*, pp. 401–405, Washington, DC, USA, June 1989.

[49] J. Rubner and K. Schulten, "Development of feature detectors by self-organization," *Biological Cybernetics*, vol. 62, no. 3, pp. 193–199, 1990.

[50] S. Y. Kung and K. I. Diamantaras, "A neural network learning algorithm for adaptive principal component extraction (APEX)," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICCASP '90)*, pp. 861–864, Albuquerque, NM, USA, April 1990.

[51] S. Fiori and F. Piazza, "A general class of ψ-APEX PCA neural algorithms," *IEEE Transactions on Circuits and Systems I*, vol. 47, no. 9, pp. 1394–1397, 2000.

[52] S. Fiori, "Experimental comparison of three PCA neural networks," *Neural Processing Letters*, vol. 11, no. 3, pp. 209–218, 2000.

[53] M. Collins, S. Dasgupta, and R. E. Schapire, "A generalization of principal component analysis to the exponential family," in *Advances in Neural Information Processing Systems*, T. D. Dietterich, S. Becker, and Z. Ghahramani, Eds., vol. 14, pp. 617–624, MIT Press, Cambridge, Mass, USA, 2002.

[54] B. Schölkopf, A. Smola, and K. R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[55] L. Xu and A. L. Yuille, "Robust principal component analysis by self-organizing rules based on statistical physics approach," *IEEE Transactions on Neural Networks*, vol. 6, no. 1, pp. 131–143, 1995.

[56] P. J. Huber, *Robust Statistics*, John Wiley & Sons, New York, NY, USA, 1981.

[57] J. Karhunen and J. Joutsensalo, "Generalizations of principal component analysis, optimization problems, and neural networks," *Neural Networks*, vol. 8, no. 4, pp. 549–562, 1995.

[58] E. Oja, H. Ogawa, and J. Wangviwattana, "Learning in non-linear constrained Hebbian networks," in *Proceedings of the International Conference on Artificial Neural Networks (ICANN '91)*, T. Kohonen, K. Makisara, O. Simula, and J. Kangas, Eds., pp. 385–390, North-Holland, Amsterdam, The Netherlands, 1991.

[59] T. D. Sanger, "An optimality principle for unsupervised learning," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, Ed., vol. 1, pp. 11–19, Morgan Kaufmann, San Mateo, Calif, USA, 1989.

[60] H. Bourlard and Y. Kamp, "Auto-association by multilayer perceptrons and singular value decomposition," *Biological Cybernetics*, vol. 59, no. 4-5, pp. 291–294, 1988.

[61] P. Baldi and K. Hornik, "Neural networks and principal component analysis: learning from examples without local minima," *Neural Networks*, vol. 2, no. 1, pp. 53–58, 1989.

[62] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE Journal*, vol. 37, no. 2, pp. 233–243, 1991.

[63] C. M. Bishop, *Neural Networks For Pattern Recogonition*, Oxford Press, New York, NY, USA, 1995.

[64] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, D. E. Rumelhart and J. L. McClelland, Eds., vol. 1, pp. 318–362, MIT Press, Cambridge, Mass, USA, 1986.

[65] N. Kambhatla and T. K. Leen, "Fast non-linear dimension reduction," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 3, pp. 1213–1218, San Francisco, Calif, USA, 1993.

[66] R. Saegusa, H. Sakano, and S. Hashimoto, "Nonlinear principal component analysis to preserve the order of principal components," *Neurocomputing*, vol. 61, no. 1–4, pp. 57–70, 2004.

[67] J. A. Catalan, J. S. Jin, and T. Gedeon, "Reducing the dimensions of texture features for image retrieval using multilayer neural networks," *Pattern Analysis and Applications*, vol. 2, no. 2, pp. 196–203, 1999.

[68] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[69] H. Ritter, "Self-organizing feature maps: kohonen maps," in *The Handbook of Brain Theory and Neural Networks*, M. A.

Arbib, Ed., pp. 846–851, MIT Press, Cambridge, Mass, USA, 1995.

[70] T. Kohonen, "Emergence of invariant-feature detectors in the adaptive-subspace self-organizing map," *Biological Cybernetics*, vol. 75, no. 4, pp. 281–291, 1996.

[71] T. Kohonen, E. Oja, O. Simula, A. Visa, and J. Kangas, "Engineering applications of the self-organizing map," *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1358–1384, 1996.

[72] L. Xu, A. Krzyzak, and E. Oja, "Rival penalized competitive learning for clustering analysis, RBF net, and curve detection," *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 636–649, 1993.

[73] K. Gao, M. O. Ahmad, and M. N. S. Swamy, "Constrained anti-Hebbian learning algorithm for total least-squares estimation with applications to adaptive FIR and IIR filtering," *IEEE Transactions on Circuits and Systems II*, vol. 41, no. 11, pp. 718–729, 1994.

[74] D. Z. Feng, Z. Bao, and L. C. Jiao, "Total least mean squares algorithm," *IEEE Transactions on Signal Processing*, vol. 46, no. 8, pp. 2122–2130, 1998.

[75] L. Xu, E. Oja, and C. Y. Suen, "Modified Hebbian learning for curve and surface fitting," *Neural Networks*, vol. 5, no. 3, pp. 441–457, 1992.

[76] A. Taleb and G. Cirrincione, "Against the convergence of the minor component analysis neurons," *IEEE Transactions on Neural Networks*, vol. 10, no. 1, pp. 207–210, 1999.

[77] K. Gao, M. O. Ahmad, and M. N. S. Swamy, "A modified Hebbian rule for total least-squares estimation with complex valued arguments," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '92)*, pp. 1231–1234, San Diego, Calif, USA, 1992.

[78] B. Widrow and M. E. Hoff, "Adaptive switching circuits," in *Proceedings of the IRE Eastern Electronic Show & Convention Convention Record (WESCON '60)*, vol. 4, pp. 96–104, 1960.

[79] S. Ouyang, Z. Bao, and G. Liao, "Adaptive step-size minor component extraction algorithm," *Electronics Letters*, vol. 35, no. 6, pp. 443–444, 1999.

[80] S. C. Douglas, S. Y. Kung, and S. I. Amari, "A self-stabilized minor subspace rule," *IEEE Signal Processing Letters*, vol. 5, no. 12, pp. 328–330, 1998.

[81] S. Attallah and K. Abed-Meraim, "Fast algorithms for subspace tracking," *IEEE Signal Processing Letters*, vol. 8, no. 7, pp. 203–206, 2001.

[82] T. Chen, "Modified Oja's algorithms for principal subspace and minor subspace extraction," *Neural Processing Letters*, vol. 5, no. 2, pp. 105–110, 1997.

[83] T. Chen, S. I. Amari, and Q. Lin, "A unified algorithm for principal and minor components extraction," *Neural Networks*, vol. 11, no. 3, pp. 385–390, 1998.

[84] K. Abed-Meraim, S. Attallah, A. Chkeif, and Y. Hua, "Orthogonal Oja algorithm," *IEEE Signal Processing Letters*, vol. 7, no. 5, pp. 116–119, 2000.

[85] F. L. Luo, R. Unbehauen, and A. Cichocki, "A minor component analysis algorithm," *Neural Networks*, vol. 10, no. 2, pp. 291–297, 1997.

[86] T. Chen, S. I. Amari, and N. Murata, "Sequential extraction of minor components," *Neural Processing Letters*, vol. 13, no. 3, pp. 195–201, 2001.

[87] Q. Zhang and Y. W. Leung, "A class of learning algorithms for principal component analysis and minor component analysis," *IEEE Transactions on Neural Networks*, vol. 11, no. 1, pp. 200–204, 2000.

[88] G. Mathew, V. U. Reddy, and S. Dasgupta, "Adaptive estimation of eigensubspace," *IEEE Transactions on Signal Processing*, vol. 43, no. 2, pp. 401–411, 1995.

[89] A. Weingessel, H. Bischof, K. Hornik, and F. Leisch, "Adaptive combination of PCA and VQ networks," *IEEE Transactions on Neural Networks*, vol. 8, no. 5, pp. 1208–1211, 1997.

[90] R. Möller and H. Hoffmann, "An extension of neural gas to local PCA," *Neurocomputing*, vol. 62, no. 1–4, pp. 305–326, 2004.

[91] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten, "'Neural-gas' network for vector quantization and its application to time-series prediction," *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 558–569, 1993.

[92] J. Karhunen and S. Malaroiu, "Locally linear Independent Component Analysis," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '99)*, pp. 882–887, Washington, DC, USA, July 1999.

[93] T. Kohonen, *Self-Organizing Maps*, Springer, Berlin, Germany, 1997.

[94] E. Oja and K. Valkealahti, "Local independent component analysis by the self-organizing map," in *Proceedings of the International Conference on Artificial Neural Networks*, pp. 553–558, Lausanne, Switzerland, 1997.

[95] J. D. Horel, "Complex principal component analysis: theory and examples," *Journal of Climate and Applied Meteorology*, vol. 23, no. 12, pp. 1660–1673, 1984.

[96] S. S. P. Rattan and W. W. Hsieh, "Complex-valued neural networks for nonlinear complex principal component analysis," *Neural Networks*, vol. 18, no. 1, pp. 61–69, 2005.

[97] T. Kim and T. Adali, "Fully complex multi-layer perceptron network for nonlinear signal processing," *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 32, no. 1-2, pp. 29–43, 2002.

[98] M. C. F. De Castro, F. C. C. De Castro, J. N. Amaral, and P. R. G. Franco, "Complex valued Hebbian learning algorithm," in *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN '98)*, pp. 1235–1238, Anchorage, Alaska, USA, May 1998.

[99] Y. Chen and C. Hou, "High resolution adaptive bearing estimation using a complex-weighted neural network," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '92)*, vol. 2, pp. 317–320, San Francisco, Calif, USA, 1992.

[100] A. Cichocki, R. W. Swiniarski, and R. E. Bogner, "Hierarchical neural network for robust PCA computation of complex valued signals," in *Proceedings of the World Congress on Neural Networks*, pp. 818–821, San Diego, Calif, USA, 1996.

[101] E. Bingham and A. Hyvarinen, "ICA of complex valued signals: a fast and robust deflationary algorithm," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '00)*, pp. 357–362, Como, Italy, July 2000.

[102] S. Fiori, "Blind separation of circularly distributed sources by neural extended APEX algorithm," *Neurocomputing*, vol. 34, pp. 239–252, 2000.

[103] S. Fiori, "Extended Hebbian learning for blind separation of complex-valued sources," *IEEE Transactions on Circuits and Systems II*, vol. 50, no. 4, pp. 195–202, 2003.

[104] Z. Yi, Y. Fu, and H. J. Tang, "Neural networks based approach for computing eigenvectors and eigenvalues of symmetric matrix," *Computers and Mathematics with Applications*, vol. 47, no. 8-9, pp. 1155–1164, 2004.

[105] Y. Liu, Z. You, and L. Cao, "A simple functional neural network for computing the largest and smallest eigenvalues

and corresponding eigenvectors of a real symmetric matrix," *Neurocomputing*, vol. 67, no. 1–4, pp. 369–383, 2005.

[106] S. Chen and T. Sun, "Class-information-incorporated principal component analysis," *Neurocomputing*, vol. 69, no. 1–3, pp. 216–223, 2005.

[107] M. S. Park and J. Y. Choi, "Theoretical analysis on feature extraction capability of class-augmented PCA," *Pattern Recognition*, vol. 42, no. 11, pp. 2353–2362, 2009.

[108] A. D'Aspremont, F. Bach, and L. El Ghaoui, "Optimal solutions for sparse principal component analysis," *Journal of Machine Learning Research*, vol. 9, pp. 1269–1294, 2008.

[109] S. Y. Kung, "Constrained principal component analysis via an orthogonal learning network," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '90)*, pp. 719–722, New Orleans, La, USA, May 1990.

[110] A. Valizadeh and M. Karimi, "Fast subspace tracking algorithm based on the constrained projection approximation," *Eurasip Journal on Advances in Signal Processing*, vol. 2009, Article ID 576972, 16 pages, 2009.

[111] J. Mao and A. K. Jain, "Artificial neural networks for feature extraction and multivariate data projection," *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 296–317, 1995.

[112] G. K. Demir and K. Ozmehmet, "Online local learning algorithms for linear discriminant analysis," *Pattern Recognition Letters*, vol. 26, no. 4, pp. 421–431, 2005.

[113] C. Chatterjee, V. P. Roychowdhury, J. Ramos, and M. D. Zoltowski, "Self-organizing algorithms for generalized eigendecomposition," *IEEE Transactions on Neural Networks*, vol. 8, no. 6, pp. 1518–1530, 1997.

[114] D. Xu, J. C. Principe, and H. C. Wu, "Generalized eigendecomposition with an on-line local algorithm," *IEEE Signal Processing Letters*, vol. 5, no. 11, pp. 298–301, 1998.

[115] G. Mathew and V. U. Reddy, "A quasi-newton adaptive algorithm for generalized symmetric eigenvalue problem," *IEEE Transactions on Signal Processing*, vol. 44, no. 10, pp. 2413–2422, 1996.

[116] Y. N. Rao, J. C. Principe, and T. F. Wong, "Fast RLS-like algorithm for generalized eigendecomposition and its applications," *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 37, no. 2-3, pp. 333–344, 2004.

[117] Y. Tang and J. Li, "Notes on "Recurrent neural network model for computing largest and smallest generalized eigenvalue"," *Neurocomputing*, vol. 73, no. 4–6, pp. 1006–1012, 2010.

[118] J. Yang, Y. Zhao, and H. Xi, "Weighted rule based adaptive algorithm for simultaneously extracting generalized eigenvectors," *IEEE Transactions on Neural Networks*, vol. 22, no. 5, pp. 800–806, 2011.

[119] D. Zhang, Z. H. Zhou, and Songcan Chen, "Diagonal principal component analysis for face recognition," *Pattern Recognition*, vol. 39, no. 1, pp. 140–142, 2006.

[120] X. Li, Y. Pang, and Y. Yuan, "L1-norm-based 2DPCA," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 40, no. 4, pp. 1170–1175, 2010.

[121] W. Zuo, D. Zhang, and K. Wang, "Bidirectional PCA with assembled matrix distance metric for image recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 36, no. 4, pp. 863–872, 2006.

[122] C. X. Ren and D. Q. Dai, "Incremental learning of bidirectional principal components for face recognition," *Pattern Recognition*, vol. 43, no. 1, pp. 318–330, 2010.

[123] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Uncorrelated multilinear principal component analysis for unsupervised multilinear subspace learning," *IEEE Transactions on Neural Networks*, vol. 20, no. 11, pp. 1820–1836, 2009.

[124] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "MPCA: multilinear principal component analysis of tensor objects," *IEEE Transactions on Neural Networks*, vol. 19, no. 1, pp. 18–39, 2008.

[125] K. I. Diamantaras and S. Y. Kung, "Cross-correlation neural network models," *IEEE Transactions on Signal Processing*, vol. 42, no. 11, pp. 3218–3223, 1994.

[126] K. L. Du and M. N. S. Swamy, "Simple and practical cyclostationary beamforming algorithms," *IEE Proceedings*, vol. 151, no. 3, pp. 175–179, 2004.

[127] D. Z. Feng, Z. Bao, and W. X. Shi, "Cross-correlation neural network models for the smallest singular component of general matrix," *Signal Processing*, vol. 64, no. 3, pp. 333–346, 1998.

[128] R. Badeau, G. Richard, and B. David, "Sliding window adaptive SVD algorithms," *IEEE Transactions on Signal Processing*, vol. 52, no. 1, pp. 1–10, 2004.

[129] A. Kaiser, W. Schenck, and R. Möller, "Coupled singular value decomposition of a cross-covariance matrix," *International Journal of Neural Systems*, vol. 20, no. 4, pp. 293–318, 2010.

[130] L. Tucker, *Implication of Factor Analysis of Three-Way Matrices for Measurement of Change*, University Wisconsin Press, Madison, Wis, USA, 1963.

[131] R. Costantini, L. Sbaiz, and S. Süsstrunk, "Higher order SVD analysis for dynamic texture synthesis," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 42–52, 2008.

[132] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, pp. 321–377, 1936.

[133] M. S. Bartlett, "Further aspects of the theory of multiple regression," in *Proceedings of the Cambridge Philosophical Society*, vol. 34, pp. 33–40, 1938.

[134] T. Hastie, A. Buja, and R. Tibshirani, "Penalized discriminant analysis," *Annals of Statistics*, vol. 23, no. 1, pp. 73–102, 1995.

[135] J. R. Kettenring, "Canonical analysis of several sets of variables," *Biometrika*, vol. 58, no. 3, pp. 433–451, 1971.

[136] L. Sun, S. Ji, and J. Ye, "Canonical correlation analysis for multilabel classification: a least-squares formulation, extensions, and analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 194–200, 2011.

[137] O. Kursun, E. Alpaydin, and O. V. Favorov, "Canonical correlation analysis using within-class coupling," *Pattern Recognition Letters*, vol. 32, no. 2, pp. 134–144, 2011.

[138] T. Sun and S. Chen, "Locality preserving CCA with applications to data visualization and pose estimation," *Image and Vision Computing*, vol. 25, no. 5, pp. 531–543, 2007.

[139] H. Wang, "Local two-dimensional canonical correlation analysis," *IEEE Signal Processing Letters*, vol. 17, no. 11, pp. 921–924, 2010.

[140] G. Kukharev and E. Kamenskaya, "Application of two-dimensional canonical correlation analysis for face image processing and recognition," *Pattern Recognition and Image Analysis*, vol. 20, no. 2, pp. 210–219, 2010.

[141] K. L. Du, A. K. Y. Lai, K. K. M. Cheng, and M. N. S. Swamy, "Neural methods for antenna array signal processing: a review," *Signal Processing*, vol. 82, no. 4, pp. 547–561, 2002.

[142] G. W. Cottrell, P. Munro, and D. Zipser, "Learning internal representations from gray-scale images: an example of extensional programming," in *Proceedings of the 9th Conference*

of tile Cognitive Science Society, pp. 462–473, Seattle, Wash, USA, 1987.

[143] W. Chen, M. J. Er, and S. Wu, "PCA and LDA in DCT domain," *Pattern Recognition Letters*, vol. 26, no. 15, pp. 2474–2482, 2005.

[144] P. Comon, "Independent component analysis, a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[145] C. Jutten and J. Herault, "Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, 1991.

[146] J. Karhunen, P. Pajunen, and E. Oja, "The nonlinear PCA criterion in blind source separation: relations with other approaches," *Neurocomputing*, vol. 22, no. 1–3, pp. 5–20, 1998.

[147] A. Hyvärinen and E. Oja, "A fast fixed-point algorithm for independent component analysis," *Neural Computation*, vol. 9, no. 7, pp. 1483–1492, 1997.

[148] S. Choi, A. Cichocki, and S. Amari, "Equivariant nonstationary source separation," *Neural Networks*, vol. 15, no. 1, pp. 121–130, 2002.

[149] A. Hyvärinen and P. Pajunen, "Nonlinear independent component analysis: existence and uniqueness results," *Neural Networks*, vol. 12, no. 3, pp. 429–439, 1999.

[150] M. D. Plumbley, "Algorithms for nonnegative independent component analysis," *IEEE Transactions on Neural Networks*, vol. 14, no. 3, pp. 534–543, 2003.

[151] W. Lu and J. C. Rajapakse, "Approach and applications of constrained ICA," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 203–212, 2005.

[152] C. S. Cruz and J. R. Dorronsoro, "A nonlinear discriminant algorithm for feature extraction and data classification," *IEEE Transactions on Neural Networks*, vol. 9, no. 6, pp. 1370–1376, 1998.

[153] P. Howland and H. Park, "Generalizing discriminant analysis using the generalized singular value decomposition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 995–1006, 2004.

[154] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Muller, "Fisher discriminant analysis with kernels," in *Proceedings of the 9th IEEE Workshop on Neural Networks for Signal Processing (NNSP '99)*, pp. 41–48, Piscataway, NJ, USA, August 1999.

[155] M. Li and B. Yuan, "2D-LDA: a statistical linear discriminant analysis for image matrix," *Pattern Recognition Letters*, vol. 26, no. 5, pp. 527–532, 2005.

[156] H. Kong, L. Wang, E. K. Teoh, J. G. Wang, and R. Venkateswarlu, "A framework of 2D fisher discriminant analysis: application to face recognition with small number of training samples," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, pp. 1083–1088, San Diego, Calif, USA, June 2005.