Fast Subspace Tracking and Neural Network Learning by a Novel Information Criterion

Yongfeng Miao and Yingbo Hua, *Senior Member, IEEE*

***Abstract*— We introduce a novel information criterion (NIC) for searching for the optimum weights of a two-layer linear neural network (NN). The NIC exhibits a single global maximum attained if and only if the weights span the (desired) principal subspace of a covariance matrix. The other stationary points of the NIC are (unstable) saddle points. We develop an adaptive algorithm based on the NIC for estimating and tracking the principal subspace of a vector sequence. The NIC algorithm provides a fast on-line learning of the optimum weights for the two-layer linear NN. We establish the connections between the NIC algorithm and the conventional mean-square-error (MSE) based algorithms such as Oja's algorithm, LMSER, PAST, APEX, and GHA. The NIC algorithm has several key advantages such as faster convergence, which is illustrated through analysis and simulation.**

I. Introduction

IT IS KNOWN that there is a close relationship between a class of linear neural networks (NN's) and the concept of principal subspace [1], [2]. The concept of principal subspace is often referred to as principal subspace analysis (PSA) in the context of statistical analysis. When we are interested in the orthonormal eigenvectors spanning a principal subspace, the so-called principal component analysis (PCA) is then referred to. Both PSA and PCA also represent the desired function of a class of linear NN's when the weights of the NN's span a principal subspace. The process of finding the proper weights is called "learning."

A class of learning algorithms such as Oja's subspace algorithm [24], the symmetric error correction algorithm [5], and the symmetric version of the back propagation algorithm [6] were developed in the past based on some heuristic reasoning. However, the work in [2] showed that all these algorithms turn out to be identical. This class of algorithms will be referred to as Oja's algorithm. The convergence property of Oja's algorithm remained as a mystery for some time until the analyses done in [8]–[10], [32]. An improved version of Oja's algorithm, called the least mean square error reconstruction (LMSER) algorithm, was developed in [7], where the well-known concept of gradient searching was applied to minimize a mean squared error (MSE). Unlike Oja's algorithm, the LMSER algorithm could be claimed to be globally convergent since the global minimum of the MSE is only achieved by the principal subspace, and all the other stationary points of the MSE are saddle points. The MSE criterion has also led to many other algorithms, which include the projection approximation subspace tracking (PAST) algorithm [16], the conjugate gradient method [17], and the Gauss–Newton method [18]. It is clear that a properly chosen criterion is a very important part in developing any learning algorithm.

In this paper, we introduce a novel information criterion (NIC) for searching for the optimum weights of a two-layer linear NN [see Fig. 2(a)]. The NIC exhibits a single global maximum attained if and only if the weights span the (desired) principal subspace of a covariance matrix. The other stationary points of the NIC are (unstable) saddle points. Unlike the MSE, the NIC is nonquadratic and has a steep landscape along the trajectory from a small weight matrix to the optimum one. Applying gradient ascent searching to the NIC yields the NIC algorithm, which is globally convergent and fast.

The rest of this paper is organized as follows. Section II lists some notational symbols and introduces the conventional quadratic PSA formulation. In Section III, we propose the NIC formulation for PSA and depicts its landscape picture. Comparisons are made to the conventional formulation and the mutual information criterion (MIC). The NIC algorithm is derived in Section IV with comparisons to a number of existing PCA/PSA algorithms and some batch-mode eigenvalue decomposition (EVD) algorithms. Section V deals with the asymptotic global convergence analysis of the NIC algorithm using the Lyapunov function approach. In Section VI, we investigate various potential applications of the NIC algorithm, which include two-layer linear NN learning, signal subspace tracking, and adaptive direction of arrival (DOA) estimation and tracking. Performance of the NIC algorithm and other algorithms is demonstrated and compared through simulation examples. Conclusions are drawn in Section VII.

II. Preliminaries

A. Notations and Acronyms

Some notational symbols and acronyms are listed below.

$\Re^{p\times q}(\Re^{p})$	Set of all $p \times q$ real matrices (p-dimensional real vectors).
$\boldsymbol{A}^T$	Transpose of a matrix $\boldsymbol{A} \in \Re^{p\times q}$.
$\mathrm{tr}(\boldsymbol{A})$	Trace of $\boldsymbol{A} \in \Re^{p\times p}$.
$\mathrm{rank}(\boldsymbol{A})$	Rank of $\boldsymbol{A} \in \Re^{p\times q}$.

Manuscript received July 21, 1997; revised October 13, 1997. This work was supported by the Australian Coorperative Research Centre for Sensor Signal and Information Processing (CSSIP) and the Australian Research Council. The associate editor coordinating the review of this paper and approving it for publication was Prof. Yu-Hen Hu.

The authors are with the Department of Electrical and Electronic Engineering, The University of Melbourne, Parkville, Victoria, Australia.

Publisher Item Identifier S 1053-587X(98)04412-2.

$\mathrm{vec}\,(\boldsymbol{A})$	Vector formed by stacking columns of $\boldsymbol{A}$ one beneath the other.
$\log(\boldsymbol{A})$	Natural logarithm of a symmetric positive definite matrix $\boldsymbol{A}$.
$\det(\boldsymbol{A})$	Determinant of $\boldsymbol{A} \in \Re^{p\times p}$.
$\|\|\boldsymbol{A}\|\|_F$	Frobenius norm of $\boldsymbol{A} \in \Re^{p\times q}$.
$\|\|\boldsymbol{x}\|\|$	Euclidean norm of $\boldsymbol{x} \in \Re^p$.
$\boldsymbol{A}\otimes\boldsymbol{B}$	Kronecker product of two matrices.
$\boldsymbol{A} > (\geq)\boldsymbol{B}$	Difference matrix $\boldsymbol{A}-\boldsymbol{B}$ is positive (nonnegative) definite.
$\boldsymbol{I}_p$	$p \times p$ identity matrix.
$\boldsymbol{0}$	Null matrix or vector.
$\mathrm{diag}\,(d_1,\cdots,d_p)$	Diagonal matrix with diagonal elements $d_1,\cdots,d_p$.
$\boldsymbol{K}_{pq}$	$pq \times pq$ commutation matrix such that $\boldsymbol{K}_{pq}\,\mathrm{vec}\,(\boldsymbol{A}) = \mathrm{vec}\,(\boldsymbol{A}^T)$.
$E, d, \partial, \nabla, \boldsymbol{H}$	Expectation, differential, partial differential, gradient, and Hessian operators, respectively.

We use the follwing acronyms in this paper.

APEX	adaptive principal component extraction;
EVD	eigenvalue decomposition;
MIC	mutual information criterion;
NIC	novel information criterion;
PAST	projection approximation subspace tracking.

B. Conventional PSA Formulation

Suppose that the vector sequence $\boldsymbol{x}_k, k = 1, 2, \cdots$ is a stationary stochastic process with zero-mean and the covariance matrix $\boldsymbol{R} = E\{\boldsymbol{x}_k\boldsymbol{x}_k^T\} \in \Re^{n\times n}$. Let λ_i and $\boldsymbol{u}_i, i = 1, 2, \cdots, n$ denote the eigenvalues and the corresponding orthonormal eigenvectors of $\boldsymbol{R}$. We shall arrange the orthonormal eigenvectors $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_n$ in such a way that the corresponding eigenvalues are in a nonascending order: $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > \lambda_{r+1} \geq \cdots \geq \lambda_n \geq 0$. Note that we do not constrain $\boldsymbol{R}$ to be nonsingular.

In some applications, $\boldsymbol{x}_k$ may be composed of r independent signal components embedded in an n-dimensional noise with $r < n$. For example, in array processing, $\boldsymbol{x}_k$ is often composed of r narrowband signal waves impinging on an n-element antenna array with white receiver noise [3]. In this case, the last $n-r$ eigenvalues of $\boldsymbol{R}$ are caused by noise that must be separated from the signal. The data $\{\boldsymbol{x}_k\}$ may also represent a sequence of image or speech phonemes, and we may be interested in storing them using a minimum amount of memory [15]. The PCA or PSA produces an optimal solution to these applications in the sense that it minimizes the MSE between $\boldsymbol{x}_k$ and its reconstruction or maximizes the variance of $\boldsymbol{y}_k$ defined by the linear transformation

$$\boldsymbol{y}_k = \boldsymbol{W}^T\boldsymbol{x}_k \tag{1}$$

where $\boldsymbol{W} \in \Re^{n\times r}$ denotes the optimal weight matrix whose columns span the same space as $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_r$, which is commonly referred to as the principal subspace, and $\boldsymbol{y}_k$ is the low-dimensional representation of $\boldsymbol{x}_k$. If $\boldsymbol{W} = [\pm\boldsymbol{u}_1, \pm\boldsymbol{u}_2, \cdots, \pm\boldsymbol{u}_r]$, then (1) performs the true PCA. In other cases when the columns of $\boldsymbol{W}$ do not necessarily yield the exact principal eigenvectors, the PSA is sufficient to yield the optimal solution by an arbitrary orthonormal base vectors spanning the principal subspace.

Conventionally, the PSA is formulated into either of the following two optimization frameworks [7], [11]:

- Maximize the variance of $\boldsymbol{y}_k$ with an orthonormality constraint on $\boldsymbol{W}$, i.e.,

$$\max_{\boldsymbol{W}} \{J_{\mathrm{VAR}}(\boldsymbol{W}) = \tfrac{1}{2}E\{\boldsymbol{y}_k^T\boldsymbol{y}_k\} = \tfrac{1}{2}\,\mathrm{tr}\,(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})\} \quad \text{subject to } \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_r. \tag{2}$$

- Minimize the MSE reconstruction of $\boldsymbol{x}_k$ from $\boldsymbol{y}_k$, i.e.,

$$\min_{\boldsymbol{W}} \{J_{\mathrm{MSE}}(\boldsymbol{W}) = \tfrac{1}{2}E\{\|\boldsymbol{x}_k - \boldsymbol{W}\boldsymbol{y}_k\|^2\} = \tfrac{1}{2}\,[\mathrm{tr}\,(\boldsymbol{R}) - \mathrm{tr}\,(2\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} - \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}\boldsymbol{W}^T\boldsymbol{W})]\}. \tag{3}$$

It is shown in [7] that the stationary points of $J_{\mathrm{MSE}}(\boldsymbol{W})$ satisfy $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_r$. Therefore, by imposing the constraint $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_r$ on (3), it can be seen that the two frameworks are in fact equivalent. The landscape of $J_{\mathrm{MSE}}(\boldsymbol{W})$ has been shown to have a global minimum at the principal subspace with all the other stationary points being saddles [7], [16]. Based on this fact, the LMSER algorithm was derived by following the exact gradient descent rule to minimize $J_{\mathrm{MSE}}(\boldsymbol{W})$. Oja's subspace algorithm can also be derived as an approximate gradient algorithm from the LMSER [11], [12]. Although both algorithms are able to converge to the principal subspace solution, their convergence speed is slow due to the slow convergence of gradient searching of quadratic criteria. The nonquadratic NIC formulation shown next has a better defined landscape for PSA, which improves the convergence of gradient searching.

III. Novel Information Criterion Formulation for PSA

Various nonquadratic generalizations of the above quadratic formulation have been proposed [11], which generally yield robust PCA/PSA solutions that are totally different from the standard ones. In contrast, the following NIC formulation not only retains the standard PSA solution but also results in a fast PSA algorithm with some attractive properties.

A. NIC Formulation for PSA

Given $\boldsymbol{W}$ in the domain $\{\boldsymbol{W}|\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} > \boldsymbol{0}\}$, we propose the following framework for PSA:

$$\max_{\boldsymbol{W}} \{J_{\mathrm{NIC}}(\boldsymbol{W}) = \tfrac{1}{2}\,\{\mathrm{tr}\,[\log(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})] - \mathrm{tr}\,(\boldsymbol{W}^T\boldsymbol{W})\}\} \tag{4}$$

where the matrix logarithm is defined in [4]. This criterion is referred to as the NIC because it is different from all existing PSA criteria and is also closely related to the mutual information criterion (MIC) shown in [10] and [19]. Nevertheless, it is worth noting that despite their similar appearance, the NIC and the MIC are in fact very different, as will be shown later in this section. The landscape of NIC is depicted by the following two theorems. Since the matrix differential method will be used extensively in deriving derivatives, we present some useful facts of the method in Appendix A. Interested readers may refer to [20] for details.

Given that the eigenvalues of $\boldsymbol{R}$ are in the nonascending order, we denote its EVD as

$$\boldsymbol{R} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T = \boldsymbol{U}_1\boldsymbol{\Lambda}_1\boldsymbol{U}_1^T + \boldsymbol{U}_2\boldsymbol{\Lambda}_2\boldsymbol{U}_2^T \tag{5}$$

where $\boldsymbol{U}_1 = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_r], \boldsymbol{U}_2 = [\boldsymbol{u}_{r+1}, \cdots, \boldsymbol{u}_n]$, and $\boldsymbol{\Lambda} = \text{diag}\,(\lambda_1, \cdots, \lambda_r, \lambda_{r+1}, \cdots, \lambda_n)$. If the eigenvalues of $\boldsymbol{R}$ are allowed to be in an arbitrary order instead of the nonascending order, the EVD can also be represented by

$$\boldsymbol{R} = \boldsymbol{U}'\boldsymbol{\Lambda}'(\boldsymbol{U}')^T = \boldsymbol{U}_r\boldsymbol{\Lambda}\boldsymbol{U}_r^T + \boldsymbol{U}_{n-r}\boldsymbol{\Lambda}_{n-r}\boldsymbol{U}_{n-r}^T \tag{6}$$

where $\boldsymbol{U}' = [\boldsymbol{U}_r, \boldsymbol{U}_{n-r}] = \boldsymbol{U}\boldsymbol{P}_n, \boldsymbol{\Lambda}' = \text{diag}\,(\lambda_1', \cdots, \lambda_r', \lambda_{r+1}', \cdots, \lambda_n') = \boldsymbol{P}_n^{-1}\boldsymbol{\Lambda}\boldsymbol{P}_n$, and $\boldsymbol{P}_n$ is an arbitrary $n \times n$ permutation matrix. In other words, $\boldsymbol{U}_r$ consists of any r distinct orthonormal eigenvectors of $\boldsymbol{R}$ with $\boldsymbol{\Lambda}_r = \text{diag}\,(\lambda_1', \cdots, \lambda_r')$ containing the corresponding eigenvalues. $\boldsymbol{U}_{n-r}$ contains the rest of the $n-r$ orthonormal eigenvectors, and $\boldsymbol{\Lambda}_{n-r} = \text{diag}\,(\lambda_{r+1}', \cdots, \lambda_n')$ contains the corresponding eigenvalues.

Theorem 3.1: $\boldsymbol{W}$ is a stationary point of $J_{\text{NIC}}(\boldsymbol{W})$ in the domain $\{\boldsymbol{W} | \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} > 0\}$ if and only if $\boldsymbol{W} = \boldsymbol{U}_r\boldsymbol{Q}$, where $\boldsymbol{U}_r \in \Re^{n\times r}$ contains any r distinct orthonormal eigenvectors of $\boldsymbol{R}$, and $\boldsymbol{Q}$ is an arbitrary orthogonal matrix.

Proof: Since $\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}$ is positive definite, it is invertible. Thus, the gradient of $J_{\text{NIC}}(\boldsymbol{W})$ with respect to $\boldsymbol{W}$ exists and is given by (see Appendix A)

$$\nabla J_{\text{NIC}}(\boldsymbol{W}) = \boldsymbol{R}\boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1} - \boldsymbol{W}. \tag{7}$$

If $\boldsymbol{W} = \boldsymbol{U}_r\boldsymbol{Q}$, where $\boldsymbol{Q}$ is an arbitrary orthogonal matrix, it is easy to show that $\nabla J_{\text{NIC}}(\boldsymbol{W}) = \boldsymbol{0}$. Conversely, by definition, the stationary point of $J_{\text{NIC}}(\boldsymbol{W})$ satisfies $\nabla J_{\text{NIC}}(\boldsymbol{W}) = \boldsymbol{0}$, which yields

$$\boldsymbol{R}\boldsymbol{W} = \boldsymbol{W}\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}. \tag{8}$$

Premultiplying both sides of (8) by $\boldsymbol{W}^T$, we obtain $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_r$, which implies that the columns of $\boldsymbol{W}$ are orthonormal at any stationary point of $J_{\text{NIC}}(\boldsymbol{W})$. Let $\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} = \boldsymbol{Q}^T\boldsymbol{\Lambda}_r'\boldsymbol{Q}$ be the EVD, where $\boldsymbol{Q}$ is an orthogonal matrix. Substituting this into (8), we have $\boldsymbol{R}\boldsymbol{U}_r' = \boldsymbol{U}_r'\boldsymbol{\Lambda}_r'$, where $\boldsymbol{U}_r' = \boldsymbol{W}\boldsymbol{Q}^T$ with $(\boldsymbol{U}_r')^T\boldsymbol{U}_r' = \boldsymbol{I}_r$. Since $\boldsymbol{\Lambda}_r'$ is a diagonal matrix and $\boldsymbol{U}_r'$ has full rank, $\boldsymbol{U}_r'$ and $\boldsymbol{\Lambda}_r'$ must be the same as $\boldsymbol{U}_r$ and $\boldsymbol{\Lambda}_r$ in (6). □

Theorem 3.1 establishes the property of all the stationary points of $J_{\text{NIC}}(\boldsymbol{W})$. The next theorem further distinguishes the global maximum attained by $\boldsymbol{W}$ spanning the principal subspace from any other stationary points, which are saddle points.

Theorem 3.2: In the domain $\{\boldsymbol{W} | \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} > 0\}$, $J_{\text{NIC}}(\boldsymbol{W})$ has a global maximum that is attained when and only when $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{P}_r\boldsymbol{Q}$ with $\boldsymbol{P}_r$ an $r \times r$ permutation matrix and $\boldsymbol{Q}$ an arbitrary orthogonal matrix. All the other stationary points are saddle points of $J_{\text{NIC}}(\boldsymbol{W})$. At this global maximum, $J_{\text{NIC}} = (\Sigma_{i=1}^r \log\lambda_i - r)/2$.

Proof: Let $\boldsymbol{H}J_{\text{NIC}}(\boldsymbol{W})$ be the Hessian matrix of $J_{\text{NIC}}(\boldsymbol{W})$ with respect to the nr-dimensional vector $\text{vec}\,(\boldsymbol{W}) = [\boldsymbol{w}_1^T, \boldsymbol{w}_2^T, \cdots, \boldsymbol{w}_r^T]^T$. From Appendix A, we have

$$\begin{aligned}\boldsymbol{H}J_{\text{NIC}}(\boldsymbol{W}) &= -\boldsymbol{I}_{rn} - (\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1} \otimes [\boldsymbol{R}\boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{R}] \\ &\quad - \boldsymbol{K}_{rn}[\boldsymbol{R}\boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}] \otimes [(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{R}] \\ &\quad + (\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1} \otimes \boldsymbol{R}\end{aligned} \tag{9}$$

where $\boldsymbol{K}_{rn}$ is the $rn \times rn$ commutation matrix, which has the property $\boldsymbol{K}_{rn}^T = \boldsymbol{K}_{rn}^{-1} = \boldsymbol{K}_{nr}$. Since any matrices $\boldsymbol{A} \in \Re^{m\times n}$ and $\boldsymbol{B} \in \Re^{p\times q}$ satisfy

$$\boldsymbol{K}_{pm}(\boldsymbol{A} \otimes \boldsymbol{B}) = (\boldsymbol{B} \otimes \boldsymbol{A})\boldsymbol{K}_{qn} \tag{10}$$

it is easy to verify that $\boldsymbol{H}J_{\text{NIC}}(\boldsymbol{W})$ yields a symmetric matrix [20]. Simplifying (9), using (8), and then evaluating $\boldsymbol{H}J_{\text{NIC}}(\boldsymbol{W})$ at the stationary points $\boldsymbol{W} = \boldsymbol{U}_r\boldsymbol{Q}$, we obtain

$$\begin{aligned}\boldsymbol{H}^* &\triangleq \boldsymbol{H}J_{\text{NIC}}(\boldsymbol{W})|_{\boldsymbol{W}=\boldsymbol{U}_r\boldsymbol{Q}} \\ &= -\boldsymbol{I}_{rn} - (\boldsymbol{Q}^T\boldsymbol{\Lambda}_r^{-1}\boldsymbol{Q}) \otimes (\boldsymbol{U}_r\boldsymbol{\Lambda}_r\boldsymbol{U}_r^T) \\ &\quad - \boldsymbol{K}_{rn}(\boldsymbol{U}_r\boldsymbol{Q}) \otimes (\boldsymbol{U}_r\boldsymbol{Q})^T + (\boldsymbol{Q}^T\boldsymbol{\Lambda}_r^{-1}\boldsymbol{Q}) \otimes \boldsymbol{R}.\end{aligned} \tag{11}$$

Now, substituting the EVD (6) into (11), we have

$$\begin{aligned}\boldsymbol{H}^* &= -\boldsymbol{I}_{rn} - \boldsymbol{K}_{rn}(\boldsymbol{U}_r\boldsymbol{Q}) \otimes (\boldsymbol{U}_r\boldsymbol{Q})^T + (\boldsymbol{Q}^T\boldsymbol{\Lambda}_r^{-1}\boldsymbol{Q}) \\ &\quad \otimes (\boldsymbol{U}_{n-r}\boldsymbol{\Lambda}_{n-r}\boldsymbol{U}_{n-r}^T) \\ &= -\boldsymbol{I}_{rn} - \boldsymbol{K}_{rn}(\boldsymbol{U}_r \otimes \boldsymbol{Q}^T)(\boldsymbol{Q} \otimes \boldsymbol{U}_r^T) + (\boldsymbol{Q}^T \otimes \boldsymbol{U}_{n-r}) \\ &\quad \cdot (\boldsymbol{\Lambda}_r^{-1} \otimes \boldsymbol{\Lambda}_{n-r})(\boldsymbol{Q} \otimes \boldsymbol{U}_{n-r}^T).\end{aligned} \tag{12}$$

Applying (10) to the second term on the right-hand side of (12) yields

$$\boldsymbol{K}_{rn}(\boldsymbol{U}_r \otimes \boldsymbol{Q}^T)(\boldsymbol{Q} \otimes \boldsymbol{U}_r^T) = (\boldsymbol{Q}^T \otimes \boldsymbol{U}_r)\boldsymbol{K}_{rr}(\boldsymbol{Q} \otimes \boldsymbol{U}_r^T) \tag{13}$$

where $\boldsymbol{K}_{rr}$ is symmetric and orthogonal, i.e., $\boldsymbol{K}_{rr}^T = \boldsymbol{K}_{rr}^{-1} = \boldsymbol{K}_{rr}$. Hence, it has the following EVD

$$\boldsymbol{K}_{rr} = \boldsymbol{V}\boldsymbol{S}\boldsymbol{V}^T \tag{14}$$

where $\boldsymbol{V}$ is an $r^2 \times r^2$ orthogonal matrix, and $\boldsymbol{S} = \text{diag}\,(s_1, s_2, \cdots, s_{r^2})$. Since $\text{tr}\,(\boldsymbol{K}_{rr}) = \Sigma_{i=1}^{r^2} s_i = r$, $\boldsymbol{S}$ must have $(r^2 + r)/2$ multiple "1" and $(r^2 - r)/2$ multiple "-1" on the diagonal. Furthermore, it is noted that $\boldsymbol{I}_{rn}$ can be decomposed as

$$\boldsymbol{I}_{rn} = \begin{bmatrix}\boldsymbol{V}^T(\boldsymbol{Q} \otimes \boldsymbol{U}_r^T) \\ \boldsymbol{Q} \otimes \boldsymbol{U}_{n-r}^T\end{bmatrix}^T \begin{bmatrix}\boldsymbol{V}^T(\boldsymbol{Q} \otimes \boldsymbol{U}_r^T) \\ \boldsymbol{Q} \otimes \boldsymbol{U}_{n-r}^T\end{bmatrix}. \tag{15}$$

Inserting (13)–(15) into (12) and after some rearrangements, we obtain the EVD of $\boldsymbol{H}^*$ as

$$\boldsymbol{H}^* = \begin{bmatrix}\boldsymbol{V}^T(\boldsymbol{Q} \otimes \boldsymbol{U}_r^T) \\ \boldsymbol{Q} \otimes \boldsymbol{U}_{n-r}^T\end{bmatrix}^T \begin{bmatrix}\boldsymbol{D}_1 & \\ & \boldsymbol{D}_2\end{bmatrix} \begin{bmatrix}\boldsymbol{V}^T(\boldsymbol{Q} \otimes \boldsymbol{U}_r^T) \\ \boldsymbol{Q} \otimes \boldsymbol{U}_{n-r}^T\end{bmatrix} \tag{16}$$

where $\boldsymbol{D}_1 = -\boldsymbol{I}_{r^2} - \boldsymbol{S}$, and $\boldsymbol{D}_2 = \boldsymbol{\Lambda}_r^{-1} \otimes \boldsymbol{\Lambda}_{n-r} - \boldsymbol{I}_{r(n-r)}$, whose diagonal elements represent the nr eigenvalues of $\boldsymbol{H}^*$. It follows that the eigenvalues of $\boldsymbol{H}^*$ are "0" with the multiplicity $(r^2 - r)/2$, "-2" with the multiplicity $(r^2 + r)/2$, and the rest of the $r(n-r)$ eigenvalues are given by

$$\begin{aligned}&\frac{\lambda_{r+1}'}{\lambda_1'} - 1, \cdots, \frac{\lambda_n'}{\lambda_1'} - 1, \frac{\lambda_{r+1}'}{\lambda_2'} - 1, \cdots, \frac{\lambda_n'}{\lambda_2'} - 1, \cdots \\ &\quad \frac{\lambda_{r+1}'}{\lambda_r'} - 1, \cdots, \frac{\lambda_n'}{\lambda_r'} - 1\end{aligned} \tag{17}$$

where $\lambda_1', \lambda_2', \cdots, \lambda_n'$ is a permutation of $\lambda_1, \lambda_2, \cdots, \lambda_n$. It is obvious from (16) that $\boldsymbol{H}^* \leq \mathbf{o}$ if and only if $\min\{\lambda_1', \cdots, \lambda_r'\} > \max\{\lambda_{r+1}', \cdots, \lambda_n'\}$. This condition is satisfied if and only if $\boldsymbol{P}_n = \text{diag}(\boldsymbol{P}_r, \boldsymbol{P}_{n-r})$, where $\boldsymbol{P}_r$ and $\boldsymbol{P}_{n-r}$ are $r \times r$ and $(n-r) \times (n-r)$ permutation matrices, repsectively. It then follows from (5) and (6) that $\boldsymbol{\Lambda}_r = \boldsymbol{P}_r^{-1}\boldsymbol{\Lambda}_1\boldsymbol{P}_r$ and $\boldsymbol{U}_r = \boldsymbol{U}_1\boldsymbol{P}_r$. Therefore, $\boldsymbol{W} = \boldsymbol{U}_r\boldsymbol{Q} = \boldsymbol{U}_1\boldsymbol{P}_r\boldsymbol{Q}$ is a local maximum, where $\boldsymbol{Q}$ is the arbitrary matrix as defined in Theorem 3.1. Furthermore, since $J_{\text{NIC}}(\boldsymbol{W})$ is unbounded from below as $||\boldsymbol{W}||_F \to \infty$ or $||\boldsymbol{W}||_F \to 0$, this local maximum is also the global one. Except for $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{P}_r\boldsymbol{Q}$, it is easy to verify that all other stationary points result in $\boldsymbol{H}^*$ being indefinite (having both positive and negative eigenvalues) and, thus, are saddle points. Finally, substituting $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{P}_r\boldsymbol{Q}$ into (4), we obtain the maximum value of $J_{\text{NIC}}(\boldsymbol{W})$ to be

$$J_{\text{NIC}} = \tfrac{1}{2}\left(\sum_{i=1}^{r} \log \lambda_i - r\right) \tag{18}$$

which concludes the proof of Theorem 3.2. □

B. Remarks and Comparisons

In the following, we make some remarks on the NIC and compare it with the conventional formulation, the MIC, and a determinant maximization framework.

1) From the above theorems, it is seen that the maximization of $J_{\text{NIC}}(\boldsymbol{W})$ automatically orthonormalizes the columns of $\boldsymbol{W}$. Therefore, we need not impose any explicit constraints on $\boldsymbol{W}$. This is similar to the MSE framework (3) but in contrast to the constrained variance maximization (2) in which the explicit orthonormalization of $\boldsymbol{W}$ is needed [24]. It should be further noted that although the NIC and the MSE both yield $\boldsymbol{W}$ with orthonormal columns, their corresponding gradient algorithms have different convergence properties. One of the attractive properties of the NIC is that $\boldsymbol{W}$ orthonormalizes itself at a constant rate independent of the eigenstructure of $\boldsymbol{R}$ (refer to Lemma 5.1). This is not the case for the MSE.
2) At the maximum of $J_{\text{NIC}}(\boldsymbol{W})$, $\boldsymbol{W}$ only yields an arbitrary orthonormal basis of the principal subspace but not the true eigenvectors, hence, the reference as a PSA criterion. This fact is indicated by the orthogonal matrix $\boldsymbol{Q}$ as well as the zero eigenvalues of $\boldsymbol{H}^*$. However, $\boldsymbol{W}\boldsymbol{W}^T = \boldsymbol{U}_1\boldsymbol{U}_1^T$ is always uniquely determined, which is the orthogonal projection onto the principal subspace of $\boldsymbol{R}$. For the special case when $\lambda_1 = \lambda_2 = \cdots = \lambda_r = \lambda$, $\boldsymbol{W}$ yields the true eigenvectors because $\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} = \lambda\boldsymbol{I}_r$ produces the required partial EVD of $\boldsymbol{R}$. In other cases, the true principal eigenvectors can be obtained from the principal subspace as follows. First, compute the EVD $\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} = \boldsymbol{Q}^T\boldsymbol{\Lambda}_r\boldsymbol{Q}$ to obtain $\boldsymbol{Q}$; then, from $\boldsymbol{W}\boldsymbol{Q}^T$, obtain $\boldsymbol{U}_1$ up to a permutation matrix $\boldsymbol{P}_r$. Note that $\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}$ can have a much smaller dimension than $\boldsymbol{R}$. In addition, note that for the special case as $r = 1$, (4) also represents a PCA criterion, and $\boldsymbol{W}$ yields the first principal eigenvector of $\boldsymbol{R}$.
3) Similar to the MSE landscape depicted in [7], [16], the NIC has a global maximum and no local ones. Thus, iterative methods like the gradient ascent search method are guaranteed to globally converge to the desired principal subspace for proper initializations of $\boldsymbol{W}$ (see the domain of attraction identified in Section V). The presence of saddle points does not cause any problem of convergence because they are avoided through random perturbations of $\boldsymbol{W}$ in practice. Furthermore, it should be noted that different landscapes of the NIC and the MSE determine different convergence speed of their corresponding gradient searching algorithms. To illustrate this, we plot in Fig. 1(a) and (b) the three-dimensional (3-D) surface for the special case when $\boldsymbol{R} = \text{diag}(2, 1)$, and $r = 1$. Note that the global extrema of J_{MSE} and J_{NIC} are located at the points $\boldsymbol{W} = [\pm 1, 0]^T$, which form the principal eigenvectors of $\boldsymbol{R}$. As visually shown in Fig. 1(a) and (b), the surface of NIC is steeper than that of the MSE along the trajectory from a small $\boldsymbol{W}$ to the optimum $\boldsymbol{W}$ (which spans the principal subspace). Note that the NIC has a steep slope in the region where $\boldsymbol{W}$ is small because of the logarithm in the first term of (4). Hence, the gradient search of the NIC is expected to converge faster than that of the MSE. This is actually implemented by an adaptive step size of the NIC algorithm, which will be seen in the next section.
4) Under the condition that $\boldsymbol{x}_k$ is a multivariate Gaussian signal corrupted by an uncorrelated Guassian noise with equal variance σ^2, the principal subspace maximizes the following MIC [19]:

$$J_{MIC}(\boldsymbol{W}) = \tfrac{1}{2}\{\text{tr}[\log(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})] - \text{tr}[\log(\sigma^2\boldsymbol{W}^T\boldsymbol{W})]\}. \tag{19}$$

If $\sigma^2 = 1$, (19) appears almost the same as $J_{\text{NIC}}(\boldsymbol{W})$, except for the logarithm in the second term. However, they are quite different behind their appearance. We note that although the principal subspace maximizes the MIC, the maximum value of the MIC may not be achieved only by the orthonormal principal subspace. This can be seen from the landscape of $J_{\text{MIC}}(\boldsymbol{W})$ shown in Fig. 1(c) for the special case when $\boldsymbol{R} = \text{diag}(2, 1)$, and $r = 1$. Note that any point at $[w_{11}, 0]^T$ with $w_{11} \neq 0$ maximizes $J_{\text{MIC}}(\boldsymbol{W})$, but it does not immediately yield the normalized principal eigenvectors. In other words, the orthonormality constraint $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_r$ has to be explicitly imposed on maximization of $J_{\text{MIC}}(\boldsymbol{W})$ to make it an effective PSA criterion.

Furthermore, for any positive definite $\boldsymbol{B} \in \Re^{p \times p}$, we have $\text{tr}[\log(\boldsymbol{B})] = \log[\det(\boldsymbol{B})]$ [10]. Therefore, the NIC in (4) can be rewritten as

$$\max_{\boldsymbol{W}} \{J_{\text{NIC}}(\boldsymbol{W}) = \tfrac{1}{2}\{\log[\det(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})] - \text{tr}(\boldsymbol{W}^T\boldsymbol{W})\}\} \tag{20}$$

which is in contrast with the determinant maximization $\max_{\boldsymbol{W}}\{\det(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})\}$. However, it is important to note that the NIC maximization is unconstrained, except

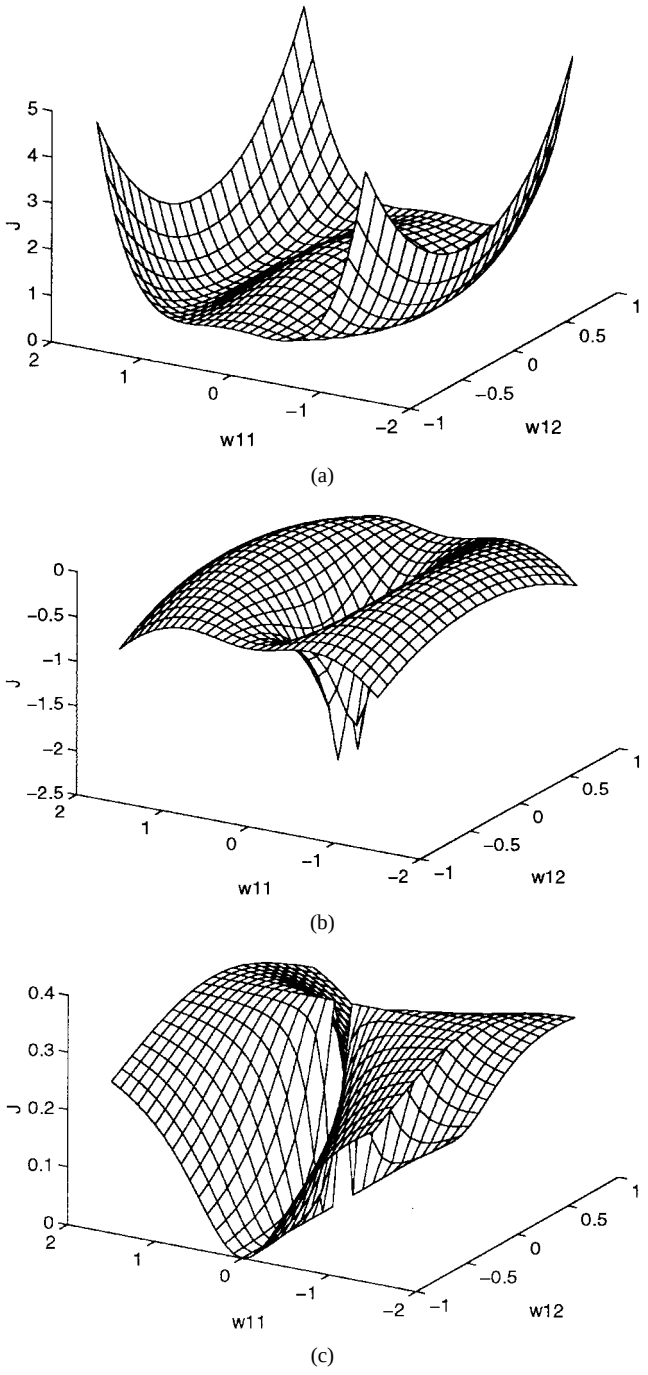


Fig. 1. Surface plots of three functions for $\boldsymbol{R} = \text{diag}\,(2, 1)$. (a) MSE. (b) NIC. (c) MIC.

for the "soft constraint" term $\text{tr}\,(\boldsymbol{W}^T\boldsymbol{W})$, but the determinant maximization requires the hard orthonormality constaint $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}_r$.

IV. NIC Learning Algorithm

At time instant k, we have $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_k$ available and are interested in estimating $\boldsymbol{W}_k$ recursively. Our objective here is to develop a fast adaptive algorithm that calculates an estimate of the principal subspace at time instant k from the estimate at $k-1$ and the newly observed sample $\boldsymbol{x}_k$. We will apply the gradient ascent searching to the unconstrained maximization of $J_{\text{NIC}}(\boldsymbol{W})$.

A. Batch Implementation

Given the gradient of $J_{\text{NIC}}(\boldsymbol{W})$ with respect to $\boldsymbol{W}$ in (7), we have the following gradient ascent rule for updating $\boldsymbol{W}_k$:

$$\boldsymbol{W}_k = (1-\eta)\boldsymbol{W}_{k-1} + \eta\hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1}(\boldsymbol{W}_{k-1}^T\hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1})^{-1} \quad (21)$$

where $0<\eta<1$ denotes the learning step size. In what follows, we will use η to denote all the constant step sizes with the understanding that they may assume different values for different algorithms. Since the exact covariance matrix $\boldsymbol{R}$ is often unknown *a priori*, we use its estimate at time k denoted by $\hat{\boldsymbol{R}}_k$. This estimate can be obtained as

$$\hat{\boldsymbol{R}}_k = 1/k\sum_{i=1}^{k}\alpha^{k-i}\boldsymbol{x}_i\boldsymbol{x}_i^T = \alpha(k-1)\hat{\boldsymbol{R}}_{k-1}/k + \boldsymbol{x}_k\boldsymbol{x}_k^T/k \quad (22)$$

where $0<\alpha\leq 1$ denotes the forgetting factor. If all training samples come from a stationary process, we choose $\alpha = 1$, and thus, (22) calculates the sample average of $\boldsymbol{x}_k\boldsymbol{x}_k^T$. For a large number of training samples, $\boldsymbol{W}_k$ will converge to the true principal subspace of $\boldsymbol{R}$ as $\hat{\boldsymbol{R}}_k$ converges to $\boldsymbol{R}$. On the other hand, if $\boldsymbol{x}_k$ are from a nonstationary process, α should be chosen in the range (0, 1) to implement an effective window of size $1/(1-\alpha)$. This effective window ensures that the past data samples are less significantly weighted than the recent ones. The exact value for α depends on specific applications. Generally speaking, for slow time-varying $\boldsymbol{x}_k$, α is chosen near one to implement a large effective window, whereas for fast time-varying inputs, the effective window should be relatively small, and α should be chosen near zero [13].

Equation (21) represents a batch implementation of the NIC algorithm where each update of $\boldsymbol{W}_k$ is in alignment with the rank-1 update of the covariance matrix. To compare it with the batch implementation of Oja's algorithm and that of the LMSER algorithm, we first apply the gradient descent rule to (3) and obtain the batch-mode LMSER algorithm [7]

$$\boldsymbol{W}_k = \boldsymbol{W}_{k-1} + \eta(2\hat{\boldsymbol{R}}_k - \hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1}\boldsymbol{W}_{k-1}^T - \boldsymbol{W}_{k-1}\cdot\boldsymbol{W}_{k-1}^T\hat{\boldsymbol{R}}_k)\boldsymbol{W}_{k-1}. \quad (23)$$

Since, for a small η, $\boldsymbol{W}_k$ tends to a matrix with orthonormal columns as $k\rightarrow\infty$ [7], the batch Oja's algorithm can be obtained from (23) by making the approximation $\boldsymbol{W}_{k-1}^T\boldsymbol{W}_{k-1}\approx\boldsymbol{I}_r$

$$\boldsymbol{W}_k = \boldsymbol{W}_{k-1} + \eta(\hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1} - \boldsymbol{W}_{k-1}\boldsymbol{W}_{k-1}^T\hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1}). \quad (24)$$

Note that (21) can be rewritten as

$$\boldsymbol{W}_k = \boldsymbol{W}_{k-1} + \eta(\hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1} - \boldsymbol{W}_{k-1}\boldsymbol{W}_{k-1}^T\hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1})\cdot(\boldsymbol{W}_{k-1}^T\hat{\boldsymbol{R}}_k\boldsymbol{W}_{k-1})^{-1}. \quad (25)$$

Comparing (24) and (25), we see that the NIC algorithm actually extends Oja's algorithm by introducing a mechanism to adaptively adjust the step size at each time step. This becomes even more obvious in the case of PCA, when $\boldsymbol{w}\in\Re^n$ represents the principal eigenvector of $\boldsymbol{R}$, and $\boldsymbol{w}_{k-1}^T\hat{\boldsymbol{R}}_k\boldsymbol{w}_{k-1}$ is a scalar. In this case, the NIC as a PCA algorithm turns out to be an extended Oja's single-neuron rule with a time-varying step size of $\eta_k = \eta/(\boldsymbol{w}_{k-1}^T\hat{\boldsymbol{R}}_k\boldsymbol{w}_{k-1})$ instead of the constant η,

which brings about a greatly improved convergence speed, as will be demonstrated by our simulations.

The batch NIC algorithm (21) has a computational complexity of $2n^2r + O(nr^2)$ flops per update. This complexity is cheaper than $12n^2r + O(nr^2)$ of the conjugate gradient eigenstructure tracking algorithm 1 (CGET1) in [17] and $8n^2r + O(nr^2)$ of the Gauss–Newton algorithm in [18]. The operations involved in (21) are simple matrix addition, multiplication, and inversion, which are easy for the systolic array implementation [13]. This batch implementation, however, is mainly suitable for adaptive subspace estimation and tracking, where the covariance matrix $\hat{\boldsymbol{R}}_k$ is explicitly invovled in computations. For the on-line learning of NN's, it is expected that the network should learn the principal subspace directly from the input data sequence $\{\boldsymbol{x}_k\}$. Such a learning algorithm is derived below.

B. Recursive Least-Squares (RLS) Implementation

Let us first assume that $\boldsymbol{W}_{k-1}^T\boldsymbol{x}_i$, which is the projection of $\boldsymbol{x}_i$ onto the column space of $\boldsymbol{W}_{k-1}$, can be approximated by $\boldsymbol{y}_i \triangleq \boldsymbol{W}_{i-1}^T\boldsymbol{x}_i$ for all $1 \le i \le k$. Note that in doing so, $\boldsymbol{y}_i$ becomes immediately available at any time instant i given the weight estimate $\boldsymbol{W}_{i-1}$ for $i = 1, 2, \cdots, k$. (Without this approximation, $\boldsymbol{W}_{k-1}^T\boldsymbol{x}_i$ would only be available at time $k-1$, when $\boldsymbol{W}_{k-1}$ is obtained.) Substituting (22) into (21), we have

$$\boldsymbol{W}_k = (1-\eta)\boldsymbol{W}_{k-1} + \eta\left(\sum_{i=1}^{k}\alpha^{k-i}\boldsymbol{x}_i\boldsymbol{y}_i^T\right) \cdot \left(\sum_{i=1}^{k}\alpha^{k-i}\boldsymbol{y}_i\boldsymbol{y}_i^T\right)^{-1}. \tag{26}$$

Note that the difference between $\boldsymbol{W}_{k-1}^T\boldsymbol{x}_i$ and $\boldsymbol{W}_{i-1}^T\boldsymbol{x}_i$ is small within an effective window if η is small. The above projection approximation has been utilized in [16] to modify the MSE criterion so that the standard RLS technique can be applicable in order to derive the so-called PAST algorithm.

Now, we define $\boldsymbol{P}_k = (\Sigma_{i=1}^k \alpha^{k-i}\boldsymbol{y}_i\boldsymbol{y}_i^T)^{-1}$ and $\tilde{\boldsymbol{W}}_k = (\Sigma_{i=1}^k \alpha^{k-i}\boldsymbol{x}_i\boldsymbol{y}_i^T)\boldsymbol{P}_k$. Applying the matrix inversion lemma [4] to $\boldsymbol{P}_k$, we obtain

$$\begin{aligned}\boldsymbol{P}_k &= \left(\alpha\sum_{i=1}^{k-1}\alpha^{k-i-1}\boldsymbol{y}_i\boldsymbol{y}_i^T + \boldsymbol{y}_k\boldsymbol{y}_k^T\right)^{-1} \\ &= (\alpha\boldsymbol{P}_{k-1}^{-1} + \boldsymbol{y}_k\boldsymbol{y}_k^T)^{-1} = \alpha^{-1}(\boldsymbol{P}_{k-1} - \boldsymbol{g}_k\boldsymbol{y}_k^T\boldsymbol{P}_{k-1})\end{aligned} \tag{27}$$

where $\boldsymbol{g}_k = \boldsymbol{P}_{k-1}\boldsymbol{y}_k/(\alpha + \boldsymbol{y}_k^T\boldsymbol{P}_{k-1}\boldsymbol{y}_k)$ is the gain vector. It is easy to verify by using (27) that $\boldsymbol{g}_k = \boldsymbol{P}_k\boldsymbol{y}_k$. Then, we can write

$$\begin{aligned}\tilde{\boldsymbol{W}}_k &= \left(\alpha\sum_{i=1}^{k-1}\alpha^{k-i-1}\boldsymbol{x}_i\boldsymbol{y}_i^T + \boldsymbol{x}_k\boldsymbol{y}_k^T\right)\boldsymbol{P}_k \\ &= \left(\sum_{i=1}^{k-1}\alpha^{k-i-1}\boldsymbol{x}_i\boldsymbol{y}_i^T\right)\boldsymbol{P}_{k-1} \\ &\quad - \left(\sum_{i=1}^{k-1}\alpha^{k-i-1}\boldsymbol{x}_i\boldsymbol{y}_i^T\right)\boldsymbol{P}_{k-1}\boldsymbol{y}_k\boldsymbol{g}_k^T + \boldsymbol{x}_k\boldsymbol{y}_k^T\boldsymbol{P}_k \\ &= \tilde{\boldsymbol{W}}_{k-1} + (\boldsymbol{x}_k - \tilde{\boldsymbol{W}}_{k-1}\boldsymbol{y}_k)\boldsymbol{g}_k^T\end{aligned} \tag{28}$$

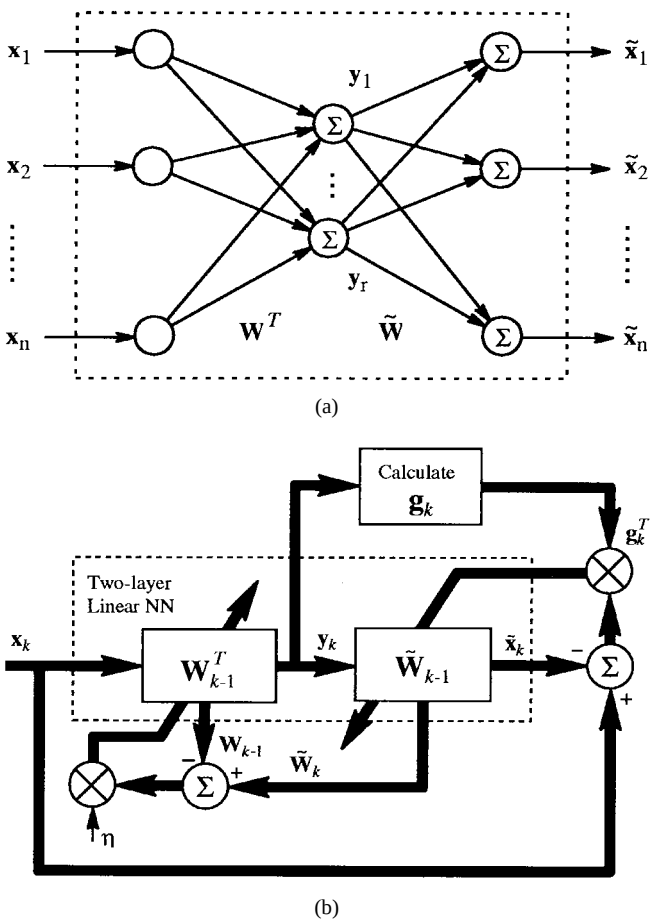


Fig. 2. (a) Two-layer linear neural network. (b) Block diagram of the two-layer linear NN learning using the NIC algorithm.

where the property that $\boldsymbol{g}_k\boldsymbol{y}_k^T\boldsymbol{P}_{k-1} = (\boldsymbol{g}_k\boldsymbol{y}_k^T\boldsymbol{P}_{k-1})^T = \boldsymbol{P}_{k-1}\boldsymbol{y}_k\boldsymbol{g}_k^T$ has been used. Now, we have shown that $\tilde{\boldsymbol{W}}_k$ can be estimated recursively from $\tilde{\boldsymbol{W}}_{k-1}$ and the new data sample $\boldsymbol{x}_k$ given $\boldsymbol{y}_k = \boldsymbol{W}_{k-1}^T\boldsymbol{x}_k$. Summarizing the above derivations, we obtain the RLS implementation of the NIC algorithm shown below.

Initializations:

$$\begin{aligned}&\boldsymbol{P}_0 = \delta\boldsymbol{I}_r \ (\delta \text{ is a small positive number}) \\ &\tilde{\boldsymbol{W}}_0 = \boldsymbol{0} \\ &\boldsymbol{W}_0 = \text{a small random } n \times r \text{ matrix}\end{aligned}$$

Update equations:

$$\boldsymbol{y}_k = \boldsymbol{W}_{k-1}^T\boldsymbol{x}_k \tag{29}$$

$$\boldsymbol{g}_k = \frac{\alpha^{-1}\boldsymbol{P}_{k-1}\boldsymbol{y}_k}{1 + \alpha^{-1}\boldsymbol{y}_k^T\boldsymbol{P}_{k-1}\boldsymbol{y}_k} \tag{30}$$

$$\boldsymbol{P}_k = \alpha^{-1}\boldsymbol{P}_{k-1} - \alpha^{-1}\boldsymbol{g}_k\boldsymbol{y}_k^T\boldsymbol{P}_{k-1} \tag{31}$$

$$\tilde{\boldsymbol{W}}_k = \tilde{\boldsymbol{W}}_{k-1} + (\boldsymbol{x}_k - \tilde{\boldsymbol{W}}_{k-1}\boldsymbol{y}_k)\boldsymbol{g}_k^T \tag{32}$$

$$\boldsymbol{W}_k = (1-\eta)\boldsymbol{W}_{k-1} + \eta\tilde{\boldsymbol{W}}_k. \tag{33}$$

The update equations (29)–(33) yield an on-line learning algorithm for the two-layer linear NN with n inputs, r hidden neurons, and n outputs [see Fig. 2(a)]. $\boldsymbol{W}_k^T$ denotes the weight matrix of the input-to-hidden layer and $\tilde{\boldsymbol{W}}_k$ that of the hidden-to-output layer at time instant k. Learning of the two-layer

linear NN is carried out by first adjusting $\tilde{\boldsymbol{W}}_k$ according to (32) and then adjusting $\boldsymbol{W}_k$ by (33). Equations (30) and (31) calculate $\boldsymbol{g}_k$, which adaptively adjusts the step size at time instant k, as will be shown next. Equation (29) and the calculation of the reconstruction $\tilde{\boldsymbol{x}}_k = \tilde{\boldsymbol{W}}_{k-1}\boldsymbol{y}_k$ simply form the forward feeding path of the network. Fig. 2(b) shows the block diagram of this learning process, where we use the symbols as defined in [13, p. 809]. It should be noted from the analysis in Section V that as $\boldsymbol{W}_k$ tends to $\boldsymbol{W}$, so does $\tilde{\boldsymbol{W}}_k$.

The initialization of this data-driven NIC algorithm is similar to that of the standard RLS algorithm [13]. The initial values for $\boldsymbol{P}_0, \tilde{\boldsymbol{W}}_0$ and $\boldsymbol{W}_0$ shall be set properly to ensure convergence. Since the covariance matrix of $\boldsymbol{y}_k$ is positive definite, $\boldsymbol{P}_0$ must be initialized as a symmetric positive definite matrix. A simple choice is $\boldsymbol{P}_0 = \delta \boldsymbol{I}_r$. The choice of $\boldsymbol{W}_0$ shall be such that $\boldsymbol{W}_k$ has full column rank and will not evolve into any of the saddle points of $J_{\mathrm{NIC}}(\boldsymbol{W})$. In other words, no column vector of $\boldsymbol{W}_0$ should be a linear combination of only $\boldsymbol{u}_{r+1}, \boldsymbol{u}_{r+2}, \cdots, \boldsymbol{u}_n$. This can be satisfied with probability 1 when $\boldsymbol{W}_0$ is randomly chosen. Clearly, $\tilde{\boldsymbol{W}}_0 = \boldsymbol{0}$ is a natural choice. In the next section, we will formally establish the required initial conditions by identifying the domain of attraction around the principal subspace for the NIC algorithm.

C. Comparisons with Other Algorithms

The stochastic LMSER algorithm was obtained by replacing $\hat{\boldsymbol{R}}_k$ in (23) with its instantaneous estimate $\boldsymbol{x}_k\boldsymbol{x}_k^T$ [7]

$$\boldsymbol{W}_k = \boldsymbol{W}_{k-1} + \eta(2\boldsymbol{x}_k\boldsymbol{y}_k^T - \boldsymbol{x}_k\boldsymbol{y}_k^T\boldsymbol{W}_{k-1}^T\boldsymbol{W}_{k-1} - \boldsymbol{W}_{k-1} \cdot \boldsymbol{y}_k\boldsymbol{y}_k^T) \tag{34}$$

where $\boldsymbol{y}_k = \boldsymbol{W}_{k-1}^T\boldsymbol{x}_k$. For stationary signals, it is expected that $\boldsymbol{W}_k$ will converge to a matrix with orthonormal columns when $\eta = \eta_k \to 0$ as $k \to \infty$. Otherwise, for a constant but small enough η, $\boldsymbol{W}_k$ will tend to a matrix with nearly orthonormal columns. When $\boldsymbol{W}_{k-1}^T\boldsymbol{W}_{k-1}$ is approximated by $\boldsymbol{I}_r$, (34) is simplified to the stochastic Oja's algorithm [24]

$$\boldsymbol{W}_k = \boldsymbol{W}_{k-1} + \eta(\boldsymbol{x}_k\boldsymbol{y}_k^T - \boldsymbol{W}_{k-1}\boldsymbol{y}_k\boldsymbol{y}_k^T). \tag{35}$$

In this sense, Oja's algorithm is derived as an approximate stochastic gradient rule to minimize the MSE.

The convergence of these two algorithms are slow because they both hinge on the standard gradient searching with the quadratic cost function and a constant step size. The PAST algorithm [16] overcomes this problem by modifying the MSE criterion so that it permits direct application of the standard RLS technique. Compared with all those algorithms, the NIC algorithm has the following distinctive characteristics:

1) Although both Oja's algorithm and the LMSER algorithm need to use a small constant step size, the NIC algorithm employs a mechanism to adaptively adjust the "actual" step size. This was discussed in the case of batch-mode implementation. The same can be seen for the RLS implementation if we write $\boldsymbol{g}_k^T = [g_{1,k}, g_{2,k}, \cdots, g_{r,k}]$ and substitute it into (32). By doing so, we obtain a set of r updating equations corresponding to the r columns of $\tilde{\boldsymbol{W}}_k$. At time instant k, the ith column of $\tilde{\boldsymbol{W}}_k$ is updated with the time-varying step size $g_{i,k}$. Since $\lambda_1, \lambda_2, \cdots, \lambda_r$ are very likely different from one another, their corresponding eigenvectors need to be treated differently to ensure a fast overall global convergence. The NIC algorithm implements this strategy through $\boldsymbol{g}_k$, resulting in a convergence speed superior to both Oja's and the LMSER algorithm.
2) It is similar to the contrast between the standard least mean-squares (LMS) and RLS algorithms for adaptive filtering [13] that Oja's algorithm and the LMSER algorithm only use the instantaneous estimate $\hat{\boldsymbol{R}}_k = \boldsymbol{x}_k\boldsymbol{x}_k^T$, whereas the NIC algorithm uses all data samples available up to time instant k to estimate the covariance matrix. For the special case when $r = 1$, we can obtain the stochastic NIC algorithm by replacing $\hat{\boldsymbol{R}}_k$ with $\boldsymbol{x}_k\boldsymbol{x}_k^T$ in (25) as
$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{\eta}{y_k^2}(y_k\boldsymbol{x}_k - y_k^2\boldsymbol{w}_{k-1}) \tag{36}$$
where $y_k = \boldsymbol{w}_{k-1}^T\boldsymbol{x}_k$. We note that (36) extends the stochastic Oja's single-neuron rule [23] by using an adaptive step size of η/y_k^2. Since this instantaneous estimate of $\hat{\boldsymbol{R}}_k$ does not directly apply to $r > 1$ ($\boldsymbol{W}_{k-1}^T\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{W}_{k-1}$ is not invertible for $r > 1$), and we estimate $\hat{\boldsymbol{R}}_k$ using (22) and obtain the above RLS type NIC algorithm. The advantage of sample averaging is that the steady-state error of the principal subspace estimate will be much smaller than that of the instantaneous estimate. Actually, the NIC will converge to an orthonormal basis of the true principal subspace for any constant $\eta \in (0, 1)$ as $\hat{\boldsymbol{R}}_k \to \boldsymbol{R}$ for large samples. However, for a constant η, the stochastic Oja's and the LMSER algorithms will always have a bias for the subspace estimate. This bias can only vanish if one imposes that $\eta = \eta_k \to 0$ as $k \to \infty$, which for practical tracking purposes is not realistic [15].
3) Note that the PAST algorithm is a special case of the NIC algorithm when η goes to one. For the NIC, η is in principle allowed to be any value within the interval (0, 1). This is also in contrast with Oja's algorithm for which the proper value of η depends on both $\boldsymbol{W}_0$ and $\boldsymbol{x}_k$ [27]. The NIC algorithm with a small η has the advantage that it will approximate the associated ODE, as will be given in Section V. From the ODE, we can formally show the asymptotic global convergence of the NIC algorithm. On the other hand, a large η will render a fast tracking capability to the NIC algorithm. It should be noted that in practice, the choice of η represents a tradeoff between accuracy and tracking capability. This is common for any adaptive algorithm [13].

It is also interesting to observe that (21) and (33) resemble the leaky LMS algorithm [13, p. 668] with a leakage factor $(1 - \eta)$. The leaky factor increases robustness and stabilizes the digital implementation of the LMS algorithm [13]. For the NIC, the update of $\boldsymbol{W}_k$ by (21) or (33) serves a similar purpose to the original PAST algorithm. In this sense, the NIC essentially represents a robust improvement of the PAST. This can be understood from the fact that the batch implemen-

tation of the PAST algorithm is, in fact, unstable. As an example, we consider the special covariance matrix $\boldsymbol{R} = \text{diag}\,(2,1)$ and $r = 1$. We choose $\boldsymbol{w}_0 = [0.5, 0]^T$ and calculate $\boldsymbol{w}_k$ by using (21) with $\eta = 1$ for all $k > 0$. It turns out that $\boldsymbol{w}_k$ always alternates from $[0.5, 0]^T$ to $[2, 0]^T$ but never converges to the correct solution $\boldsymbol{w} = [\pm 1, 0]^T$. However, by simply using an $\eta \in (0, 1)$, (21) always converges to the right solution for the NIC algorithm. The same needs to be investigated between the data-driven PAST and NIC algorithms, which is beyond the scope of the current paper.

4) Although derived primarily for extracting the principal subspace of dimension r, the NIC algorithm can be easily adapted to extract the individual principal eigenvectors when used in conjunction with the deflation technique [15]. As we have noticed earlier, for the first principal eigenvector, the NIC algorithm improves Oja's single-neuron rule by utilizing a well-defined adaptive step size. This adaptive step size is $\eta/(\boldsymbol{w}_{k-1}^T \hat{\boldsymbol{R}}_k \boldsymbol{w}_{k-1})$ in the batch implementation (25), η/y_k^2 in the stochastic case (36), and $(1/\Sigma_{i=1}^k \alpha^{k-i} y_i^2)$ for the RLS implementation (29)–(33), where $0 < \eta < 1$. Once again, this RLS implementation of the NIC algorithm generalizes the existing RLS-type PCA algorithm [14], [15] as $\eta \to 1$. For the rest of the principal eigenvectors, all the above implementations of the NIC algorithm can be readily applied to the "deflated" covariance matrix or the data samples. This has profound implications for a large number of other PCA algorithms like, for example, the adaptive principal component extraction (APEX) algorithm [15] and the generalized Hebbian algorithm (GHA) [26], which are extensions of Oja's single-neuron rule with some deflation mechanisms. The NIC will therefore provide some potential improvements over the APEX and the GHA because of a well-defined adaptive step size. Furthermore, since the NIC corresponds to an exact gradient rule while Oja's rule does not, the global convergence of the PCA algorithms based on the NIC will be easy to establish.

V. Global Convergence Analysis

We now study a convergence property of the NIC algorithm by considering the gradient rule (21). Under the condition that $\boldsymbol{x}_k$ is from a stationary process and the step size η is small enough, the discrete-time difference equation (21) approximates the continuous-time ordinary differential equation (ODE)

$$\frac{d\boldsymbol{W}(t)}{dt} = \boldsymbol{R}\boldsymbol{W}(t)[\boldsymbol{W}^T(t)\boldsymbol{R}\boldsymbol{W}(t)]^{-1} - \boldsymbol{W}(t) \tag{37}$$

where $t = \eta k$. By analyzing the global convergence properties of (37), we will establish the condition for the global convergence of (21) as well as that of the RLS based NIC algorithm (29)–(33). In particular, we will answer the following questions based on the Lyapunov function approach [21].

- Is the dynamical system described by (37) able to globally converge to the principal subspace solution?
- What is the domain of attraction around the equilibrium attained at the principal subspace, or equivalently, what is the initial condition to ensure the global convergence?

Denote $L'(\boldsymbol{W}) = -J_{\text{NIC}}(\boldsymbol{W})$. It defines a Lyapunov function for the ODE (37). To show this, let us define a region $D = \{\boldsymbol{W} | L'(\boldsymbol{W}) < \infty\} = \{\boldsymbol{W} | \boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W} > \boldsymbol{0}\}$. Within this region, $L'(\boldsymbol{W})$ is continuous and has a continuous first-order derivative. It has a global minimum at $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{P}_r\boldsymbol{Q}$ (see Theorem 3.2). Since all the other stationary points of $L'(\boldsymbol{W})$ are saddles and, thus, are unstable, we need only to consider the stability at this global minimum. By the chain rule, we have

$$\frac{dL'(\boldsymbol{W}(t))}{dt} = -\text{tr}\left[\nabla J_{\text{NIC}}(\boldsymbol{W}(t)) \cdot \frac{d\boldsymbol{W}^T(t)}{dt}\right]. \tag{38}$$

By substituting (7) and (37) into (38), it is easy to show that $(dL'(\boldsymbol{W})/dt) \le 0$ for any $\boldsymbol{W} \in D$. Therefore, the equilibrium at $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{P}_r\boldsymbol{Q}$ is stable. Since $L'(\boldsymbol{W}) = \infty$ if and only if $\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}$ is singular, the above also means that $\boldsymbol{W}(t)$ is in D if $\boldsymbol{W}_0$ is in D. To further establish the asymptotic stability and, equivalently, the global convergence property at $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{P}_r\boldsymbol{Q}$, we construct

$$L(\boldsymbol{W}) = \tfrac{1}{2}\{\text{tr}\,(\boldsymbol{W}^T\boldsymbol{W}) - \text{tr}\,[\log(\boldsymbol{W}^T\boldsymbol{R}_r\boldsymbol{W})]\} \tag{39}$$

where $\boldsymbol{R}_r = \boldsymbol{U}_1\boldsymbol{\Lambda}_1\boldsymbol{U}_1^T$ is the best (least-squares) rank-r approximation of $\boldsymbol{R}$. The following lemma paves the way for our proof of $L(\boldsymbol{W})$ as a Lyapunov function.

Lemma 5.1: Let $\boldsymbol{W}(t)$ be the solution of the ODE (37) and $\boldsymbol{W}(0) \in D$. Then, for all $t \in [0, \infty)$, we have

$$||\boldsymbol{W}^T(t)\boldsymbol{W}(t) - \boldsymbol{I}_r||_F = e^{-2t}||\boldsymbol{W}^T(0)\boldsymbol{W}(0) - \boldsymbol{I}_r||_F. \tag{40}$$

Proof: From (37), $\boldsymbol{W}^T(t)\boldsymbol{W}(t)$ satisfies the following ODE:

$$\frac{d\boldsymbol{W}^T(t)\boldsymbol{W}(t)}{dt} = 2\boldsymbol{I}_r - 2\boldsymbol{W}^T(t)\boldsymbol{W}(t). \tag{41}$$

Since it is linear in $\boldsymbol{W}^T(t)\boldsymbol{W}(t)$, this ODE admits the explicit solution

$$\boldsymbol{W}^T(t)\boldsymbol{W}(t) = \boldsymbol{I}_r - e^{-2t}[\boldsymbol{I}_r - \boldsymbol{W}^T(0)\boldsymbol{W}(0)] \tag{42}$$

which gives (40) directly. □

This lemma establishes the constant convergence rate of $\boldsymbol{W}^T(t)\boldsymbol{W}(t)$ to the identity matrix from any initial $\boldsymbol{W}(0)$ satisfying $\boldsymbol{W}^T(0)\boldsymbol{R}\boldsymbol{W}(0) > \boldsymbol{0}$. The conclusion is in sharp contrast to the convergence rate at which the weight matrix orthonormalizes itself in Oja's subspace algorithm. As discovered in [9], that rate depends on both the initial value $\boldsymbol{W}(0)$ and the smallest nonzero eigenvalue of $\boldsymbol{R}$. In addition, from (42), the following relationships hold for all $t \in [0, \infty)$:

$$\begin{aligned} \min\{\sqrt{r}, ||\boldsymbol{W}(0)||_F\} \le ||\boldsymbol{W}(t)||_F \le \max\{\sqrt{r}, ||\boldsymbol{W}(0)||_F\} \\ \text{rank}[\boldsymbol{I}_r - \boldsymbol{W}^T(t)\boldsymbol{W}(t)] = \text{rank}[\boldsymbol{I}_r - \boldsymbol{W}^T(0)\boldsymbol{W}(0)] \\ \text{rank}[\boldsymbol{W}(t)] = \text{rank}[\boldsymbol{W}(0)]. \end{aligned} \tag{43}$$

Theorem 5.1: $L(\boldsymbol{W})$ is a Lyapunov function for the ODE (37), which identifies the domain of attraction for convergence of $\boldsymbol{W}$ to $\boldsymbol{U}_1\boldsymbol{Q}'$ as

$$\Omega = \{\boldsymbol{W} | \boldsymbol{W}^T \boldsymbol{R}_r \boldsymbol{W} > \boldsymbol{0}\} \tag{44}$$

where $\boldsymbol{Q}'$ is an arbitrary orthogonal matrix. For any $\boldsymbol{W}(0) \in \Omega$, $\boldsymbol{W}(t)$ globally converges along the trajectory of (37) to an arbitrary orthonormal basis of the principal subspace.

Proof: See Appendix B.

Corollary 5.1: Let $\boldsymbol{W}(t)$ be a solution of the ODE (37). If $\boldsymbol{W}(0) \in \Omega$, then for all $t \in [0, \infty)$, $\boldsymbol{W}(t) \in \Omega$.

Proof: From Theorem 5.1, $L(\boldsymbol{W})$ is a nonincreasing function of $\boldsymbol{W}$ within the domain Ω, which implies that $\boldsymbol{W}^T(t)\boldsymbol{R}_r\boldsymbol{W}(t)$ cannot be singular if $\boldsymbol{W}(0) \in \Omega$. [Otherwise, $L(\boldsymbol{W})$ would be infinitely large.] Thus, we must have $\boldsymbol{W}(t) \in \Omega$ for all $t \in [0, \infty)$. □

We know that for $\boldsymbol{W}(0) \in \Omega$, $\boldsymbol{W}^T(0)\boldsymbol{U}_1$ is nonsingular, and hence, no column of $\boldsymbol{W}(0)$ should be orthogonal to the principal subspace spanned by the columns of $\boldsymbol{U}_1$. This is a natural requirement for any gradient-based subspace algorithm to converge. For a full rank $\boldsymbol{W}(0) \notin \Omega$, Corollary 5.1 implies that the solution of the ODE (37) would get stuck at some saddle points of $J_{\mathrm{NIC}}(\boldsymbol{W})$. Therefore, Theorem 5.1 identifies the largest domain of attraction for the ODE to converge to the principal subspace solution. A randomly selected $\boldsymbol{W}(0)$ satisfies (44) almost surely. Another point worth noting is about the singularity of $\boldsymbol{R}$, as mentioned earlier in Section II. In the worst case, when all the last $n-r$ eigenvalues of $\boldsymbol{R}$ are zero, $\boldsymbol{R}$ becomes the same as $\boldsymbol{R}_r$. However, it follows from Corollary 5.1 that as long as $\boldsymbol{W}(0) \in \Omega$, $\boldsymbol{W}(t)$ will remain within Ω along the trajectory of (37) and will converge to the principal subspace solution.

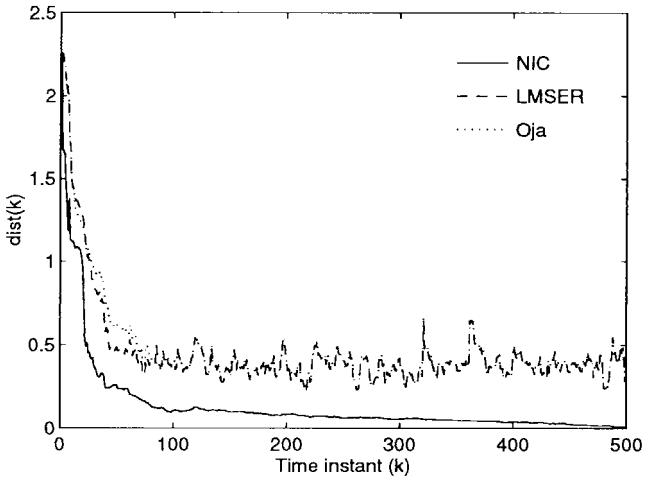
VI. Application Examples

The potential applications of the NIC algorithm are broad. For applications such as frequency estimation in array processing [3], optimal feature extraction in pattern recognition [25], data compression in image processing [28], and shape and motion estimation in computer vision [29], the NIC algorithm provides a fast adaptive method to estimate the principal subspace of data. Some examples are presented in the following simulations.

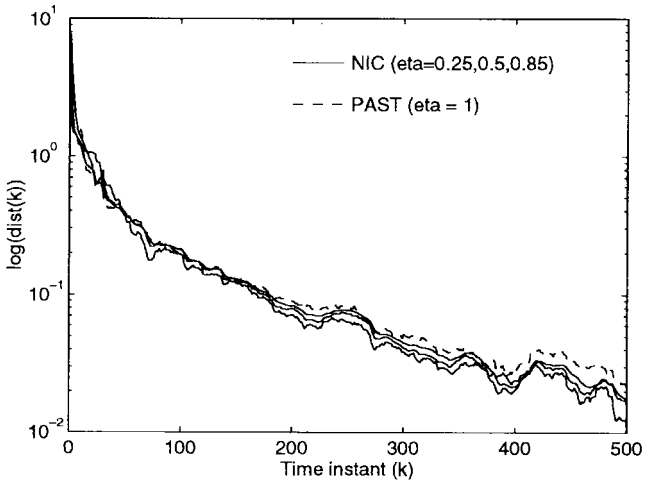
A. Learning of Two-Layer Linear NN

We generate a random vector sequence from an ideal covariance matrix with these eigenvalues: $\{26.57, 19.91, 11.25, 1.29, 1.22, 1.03, 0.99, 0.93, 0.44, 0.12\}$ and $n = 10$. The three learning algorithms—the NIC (29)–(33), the LMSER (34), and Oja's subspace algorithm (35)—are run for the same random $\boldsymbol{W}_0$ with $\eta = 0.85$, 0.006, and 0.006, respectively. The reason why a small η is used for the LMSER and Oja's algorithms will be explained later. For the NIC and the PAST algorithms, $\boldsymbol{P}_0 = 0.05\boldsymbol{I}_3$. The error of the estimated principal subspace at time k is measured by the "learning curve"

$$\mathrm{dist}\,(k) = \frac{1}{L} \sum_{\mathrm{run}} ||\boldsymbol{W}_k \boldsymbol{W}_k^T - \boldsymbol{U}_1 \boldsymbol{U}_1^T||_F$$



(a)



(b)

Fig. 3. (a) Learning curves for subspace distance of the NIC algorithm, the LMSER algorithm, and Oja's algorithm. (b) Learning curves for the subspace distance of the NIC algorithm and the PAST algorithm.

where $L = 50$ is the number of Monte Carlo runs, and $\boldsymbol{U}_1$ contains the first three principal eigenvectors of the covariance matrix based on 500 samples. The learning curves $\mathrm{dist}\,(k)$ of the three algorithms are shown in Fig. 3(a). Fig. 3(b) depicts the learning curves of the NIC for different choices of $\eta \in (0, 1)$ and that of the PAST algorithm with $\eta = 1$ [In this special case, $\tilde{\boldsymbol{W}}_0$ must be chosen as a random matrix instead of zero by (33)].

It is observed that the NIC algorithm outperforms the others in both the convergence speed and the estimation accuracy. It converges to the true principal subspace as $k \to \infty$ for any $\eta \in (0, 1)$. However, due to stochastic approximation of the covariance matrix, any particular run of Oja's subspace or the LMSER algorithm cannot converge to but must fluctuate around the true principal subspace. Furthermore, the proper step size for these two algorithms is difficult to determine because it is dependent on both $\boldsymbol{R}$ and $\boldsymbol{W}_0$. The step size $\eta = 0.006$ was chosen by trial and error to be almost optimal. It is also observed from Fig. 3(b) that for some $\eta \in (0, 1)$, the NIC algorithm demonstrates faster convergence and better

accuracy than the PAST algorithm. This verifies Remark 3 we made in Section IV-C.

B. Signal Subspace Tracking

To study the tracking capability of the NIC algorithm, we use the example given by Comon and Golub in their survey paper for comparing tracking capabilities of various subspace algorithms [3]. Two random signals are generated using moving-average models of order 2 driven by independent Gaussian white noise. Let $\boldsymbol{e}_i$ denote the vector formed by all zeros with one at the ith position. At $k = k_0$, a sudden change of the signal subspace is introduced by rotating the principal eigenvectors 90°. Thus, the observed data at time instant k are given by

$$\begin{cases} \boldsymbol{x}_k = S_{1,k}\boldsymbol{e}_1 + S_{2,k}\boldsymbol{e}_2 + \boldsymbol{n}_k & \text{for } k \le k_0 = 10 \\ \boldsymbol{x}_k = S_{1,k}\boldsymbol{e}_3 + S_{2,k}\boldsymbol{e}_4 + \boldsymbol{n}_k & \text{for } k > k_0 = 10 \end{cases}$$

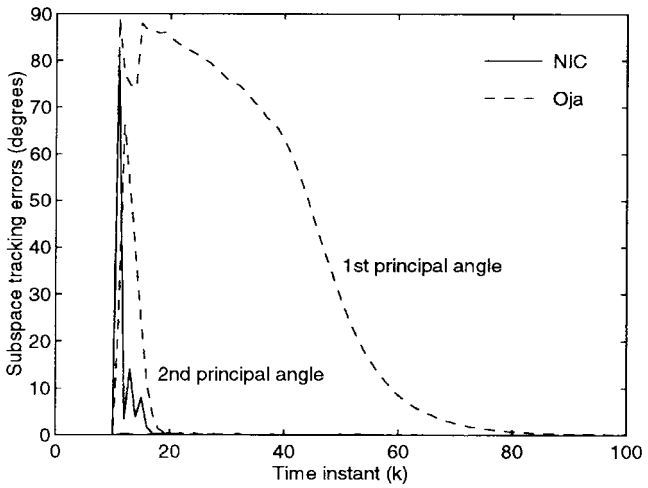
where $S_{1,k}$ and $S_{2,k}$ are two zero-mean signals with $E\{(S_{1,k})^2\} = 4$ and $E\{(S_{2,k})^2\} = 1$; $\boldsymbol{n}_k$ is a vector white noise with $E\{\boldsymbol{n}_k\boldsymbol{n}_k^T\} = \sigma^2\boldsymbol{I}_{10}$ and $\sigma = 0.1$; and the dimension of observation is $n = 10$.

The batch NIC algorithm (21) is simulated to compare against the batch Oja's algorithm as well as the simulation results in [3]. The subspace tracking error of each algorithm is measured using the set of principal angles [4, p. 584] between the subspace spanned by columns of $\boldsymbol{W}_k$ and that obtained from the EVD applied directly to (22) at each time instant. The principal angles are zero if the subspaces compared are identical. Thus, the subspace tracking capability of an algorithm is fully tested by enforcing a maximal abrupt change in the principal subspace. In Fig. 4(a), the subspace tracking error of the NIC and that of Oja's algorithm are depicted, and the estimated principal eigenvalues are shown in Fig. 4(b) against those obtained by the direct EVD.

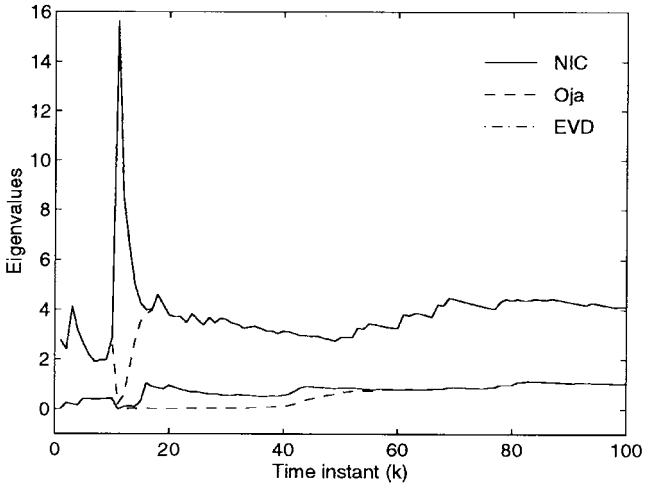
We note that both the principal angles and the estimated eigenvalues of the NIC follow the true values by EVD almost immediately, demonstrating very good tracking capability of the NIC algorithm. In contrast, very bad convergence properties of some standard gradient-based algorithms such as the LMSER and Oja's algorithm or some variants are demonstrated in [3] and by Fig. 4(a) and (b) for the same data. More importantly, the NIC algorithm not only has a tracking capability comparable with the direct EVD but also permits analog neural network realization of the ODE (37), which could achieve much faster speed than the digital implementation [30].

C. DOA Estimation and Tracking

This test shows the applicability of the NIC algorithm (21) for adaptive DOA estimation and tracking in array processing. Consider the scenario where two equipower incoherent plane waves impinge on a uniform linear array with eight sensors from the directions 9° and 12°. The receiver noise is spatially white (possibly after a prewhitening process) with the unit variance $\sigma^2 = 1$, and the signal-to-noise ratio is 20 dB. The NIC algorithm is initiated by choosing $\eta = 0.5, \boldsymbol{R}_0 = 0.001\boldsymbol{I}_8$, and $\boldsymbol{W}_0 = [\boldsymbol{e}_1, \boldsymbol{e}_2]$. After $\boldsymbol{W}_k$ is obtained at each time instant



(a)

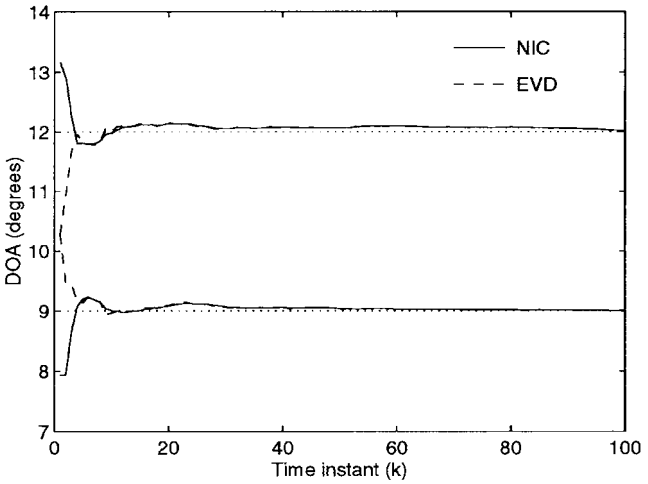


(b)

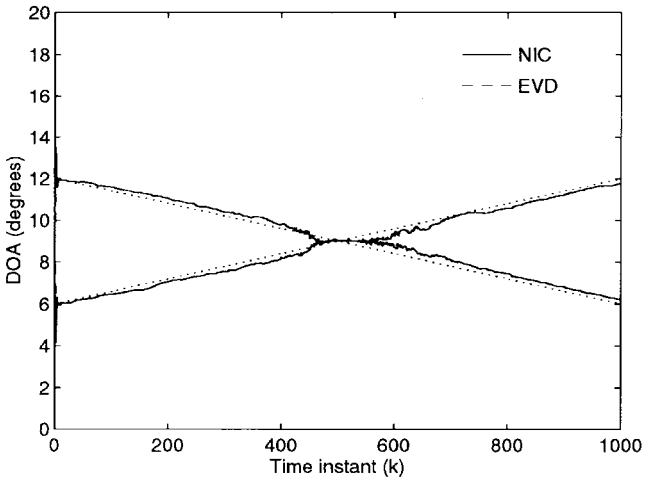
Fig. 4. (a) Subspace tracking errors of the NIC algorithm and Oja's algorithm after a $90°$ rotation of the principal subspace at $k = 10$. (b) Two principal eigenvalues obtained from the NIC and Oja's algorithm against the true values by EVD. Note that the eigenvalue curves by the NIC coincide with those by the EVD.

k, the ROOT-MUSIC estimator [13] is used to get the DOA estimates. (Better methods such as matrix pencil or ESPRIT [31] could also be used.) Twenty independent simulations of the problem are performed. Fig. 5(a) shows the DOA learning curves of the NIC algorithm in comparison with those of the direct EVD applied to (22), where the two true DOA's (9° and 12°) are shown in dotted lines. It is noted that the NIC is able to track the EVD almost immediately after the start-up transient and attains an accurate DOA estimate in about 30 time steps. By contrast, the gradient-based algorithms as shown in [22] took more than 200 samples to get a satisfactory estimate.

To further test the tracking ability of the NIC algorithm, we consider the above scenario where two signal waves have linearly time-varying frequency tracks. The two DOA's start at 6° and 12°, cross at 9°, and finish at 12° and 6° over a span of 1000 samples. The NIC algorithm is executed for ten independent runs based on the same initial condition and parameters shown previously, and the forgetting factor is $\alpha =$



(a)



(b)

Fig. 5. (a) DOA learning curves of the NIC algorithm in comparison with EVD using ROOT-MUSIC estimator. (b) DOA tracks of slowly time-varying signal waves of the NIC algorithm in comparison with EVD using ROOT-MUSIC estimator. Note that the DOA tracks by the NIC coincide with those of the EVD.

0.97. The estimated DOA tracks by the NIC and the EVD are shown in Fig. 5(b) with the dotted lines representing the true DOA tracks. It is seen that the NIC algorithm demonstrates a very good capability for tracking time-varying DOA's and attains good DOA estimates in a very short transient time. It also maintains an accurate direction estimate, even though the number of signals drops from 2 to 1 at the crossing.

VII. Conclusions

The NIC maximization is a novel nonquadratic formulation of the PSA and has some significant advantages over the conventional formulation. Among these advantages, the global maximum at the principal subspace and all the other stationary points being saddles enable us to directly apply the gradient-based searching technique to the NIC maximization problem. It is important to note that due to the nonquadratic property of the NIC, the resultant NIC algorithm brings about some attractive properties. In particular, it overcomes the slow convergence of Oja's subspace algorithm and the LMSER algorithm and is able to globally converge to the PSA solution for almost all weight initializations. The NIC also generalizes some well-known PSA/PCA algorithms by introducing a well-defined adaptive step size for learning, which provides potential improvements of a number of other PCA algorithms based on Oja's single-neuron rule. The global analysis using the Lyapunov function approach has identified the largest domain of attraction for the equilibrium attained at the principal subspace. The NIC algorithm is clearly useful in real-time signal processing applications where fast adaptive subspace estimation is required. Issues such as the explicit convergence rate of the NIC algorithm, the connections between the iterative equation (21) and the classical orthogonal iteration technique [4], and the effect of step size η of the NIC are currently under further investigation.

Appendix A
Some Facts of the Matrix Differential Method

We briefly explain the procedure of using the matrix differential method to compute the derivative of a function of matrix. Since the computation of differentials is relatively easy, the computation of derivatives can be performed simply based on the following lemma [20].

Lemma A.1: Let ϕ be a twice differentiable real-valued function of an $n \times r$ matrix $\boldsymbol{X}$. Then, the following relationships hold

$$d\phi(\boldsymbol{X}) = \mathrm{tr}\,(\boldsymbol{A}^T d\boldsymbol{X}) \Leftrightarrow \nabla\phi(\boldsymbol{X}) = \boldsymbol{A} \tag{A.1}$$

$$\begin{aligned} d^2\phi(\boldsymbol{X}) &= \mathrm{tr}\,[\boldsymbol{B}(d\boldsymbol{X})^T \boldsymbol{C} d\boldsymbol{X}] \Leftrightarrow \boldsymbol{H}\phi(\boldsymbol{X}) \\ &= \tfrac{1}{2}\,(\boldsymbol{B}^T \otimes \boldsymbol{C} + \boldsymbol{B} \otimes \boldsymbol{C}^T) \end{aligned} \tag{A.2}$$

$$\begin{aligned} d^2\phi(\boldsymbol{X}) &= \mathrm{tr}\,[\boldsymbol{B}(d\boldsymbol{X})\boldsymbol{C} d\boldsymbol{X}] \Leftrightarrow \boldsymbol{H}\phi(\boldsymbol{X}) \\ &= \tfrac{1}{2}\,\boldsymbol{K}_{rn}(\boldsymbol{B}^T \otimes \boldsymbol{C} + \boldsymbol{C}^T \otimes \boldsymbol{B}) \end{aligned} \tag{A.3}$$

where d denotes the differential, and $\boldsymbol{A}, \boldsymbol{B}$, and $\boldsymbol{C}$ are matrices, each of which may be a function of $\boldsymbol{X}$. The gradient of ϕ with respect to $\boldsymbol{X}$ and the Hessian matrix of ϕ at $\boldsymbol{X}$ are defined as

$$\nabla\phi(\boldsymbol{X}) = \frac{\partial\phi(\boldsymbol{X})}{\partial \boldsymbol{X}}$$

and

$$\boldsymbol{H}\phi(\boldsymbol{X}) = \frac{\partial}{\partial(\mathrm{vec}\,\boldsymbol{X})^T}\left(\frac{\partial\phi(\boldsymbol{X})}{\partial(\mathrm{vec}\,\boldsymbol{X})^T}\right)^T \tag{A.4}$$

where vec is the vector operator.

Based on this Lemma, we are ready to derive the gradient and Hessian matrix of $J_{\mathrm{NIC}}(\boldsymbol{W})$. From (4) and after some calculations, we have

$$dJ_{\mathrm{NIC}}(\boldsymbol{W}) = \mathrm{tr}\,[(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{R}d\boldsymbol{W}] - \mathrm{tr}\,(\boldsymbol{W}^T d\boldsymbol{W}). \tag{A.5}$$

Applying (A.1) to (A.5) gives the gradient of $J_{\mathrm{NIC}}(\boldsymbol{W})$ with respect to $\boldsymbol{W}$, as shown in (7). From (A.5), we calculate the

second-order differential

$$\begin{aligned} d^2J_{\mathrm{NIC}}(\boldsymbol{W}) &= -\mathrm{tr}\,(d\boldsymbol{W}^T d\boldsymbol{W}) - \mathrm{tr}\,[(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}d(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W}) \\ &\quad \cdot (\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{R}d\boldsymbol{W}] \\ &\quad + \mathrm{tr}\,[(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}(d\boldsymbol{W}^T)\boldsymbol{R}d\boldsymbol{W}] \\ &= -\mathrm{tr}\,(d\boldsymbol{W}^T d\boldsymbol{W}) - \mathrm{tr}\,[(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}(d\boldsymbol{W}^T)\boldsymbol{R} \\ &\quad \cdot \boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{R}d\boldsymbol{W}] \\ &\quad - \mathrm{tr}\,[(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{R}(d\boldsymbol{W})(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}\boldsymbol{W}^T \\ &\quad \cdot \boldsymbol{R}d\boldsymbol{W}] \\ &\quad + \mathrm{tr}\,[(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1}(d\boldsymbol{W}^T)\boldsymbol{R}d\boldsymbol{W}]. \end{aligned} \tag{A.6}$$

The Hessian matrix (9) follows directly from the superposition principle on (A.2), (A.3), and (A.6).

Appendix B
Proof of Theorem 5.1

We will follow the Lyapunov function approach to prove Theorem 5.1. Consider (39), and define $\Omega = \{\boldsymbol{W} | L(\boldsymbol{W}) < \infty\}$. For any $\boldsymbol{W}$ with a norm bounded by the inequality in (43), $L(\boldsymbol{W}) = \infty$ if and only if $\boldsymbol{W}^T\boldsymbol{R}_r\boldsymbol{W}$ is singular. Thus, we obtain

$$\Omega = \{\boldsymbol{W} | \boldsymbol{W}^T\boldsymbol{R}_r\boldsymbol{W} > \boldsymbol{0}\}. \tag{A.7}$$

It is obvious that within this region, $L(\boldsymbol{W})$ is continuous and has the following continuous derivative:

$$\nabla L(\boldsymbol{W}) = \boldsymbol{W} - \boldsymbol{R}_r\boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{R}_r\boldsymbol{W})^{-1}. \tag{A.8}$$

Differentiating (39) with respect to t and using (37) and (A.8), we have

$$\begin{aligned} \frac{dL(\boldsymbol{W})}{dt} &= -\mathrm{tr}\,\{[\boldsymbol{R}_r\boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{R}_r\boldsymbol{W})^{-1} - \boldsymbol{W}]^T \\ &\quad \cdot [\boldsymbol{R}\boldsymbol{W}(\boldsymbol{W}^T\boldsymbol{R}\boldsymbol{W})^{-1} - \boldsymbol{W}]\}. \end{aligned} \tag{A.9}$$

It is clear from [21] that we now only need to establish $(dL(\boldsymbol{W})/dt) < 0$ for any $\boldsymbol{W} \in \Omega^* = \Omega - \{\boldsymbol{W} | \boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{Q}'\}$ and $(dL(\boldsymbol{W})/dt) = 0$ for $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{Q}'$. For convenience, we have dropped the dependence of $\boldsymbol{W}(t)$ on t in equations.

Based on the EVD given by (5), we rewrite (A.9) as

$$\begin{aligned} \frac{dL(\boldsymbol{Z})}{dt} &= -\mathrm{tr}\left\{\begin{bmatrix} \boldsymbol{\Lambda}_1\boldsymbol{Z}_1(\boldsymbol{Z}_1^T\boldsymbol{\Lambda}_1\boldsymbol{Z}_1)^{-1} - \boldsymbol{Z}_1 \\ -\boldsymbol{Z}_2 \end{bmatrix}^T \right. \\ &\quad \left. \cdot \begin{bmatrix} \boldsymbol{\Lambda}_1\boldsymbol{Z}_1(\boldsymbol{Z}_1^T\boldsymbol{\Lambda}_1\boldsymbol{Z}_1 + \boldsymbol{Z}_2^T\boldsymbol{\Lambda}_2\boldsymbol{Z}_2)^{-1} - \boldsymbol{Z}_1 \\ \boldsymbol{\Lambda}_2\boldsymbol{Z}_2(\boldsymbol{Z}_1^T\boldsymbol{\Lambda}_1\boldsymbol{Z}_1 + \boldsymbol{Z}_2^T\boldsymbol{\Lambda}_2\boldsymbol{Z}_2)^{-1} - \boldsymbol{Z}_2 \end{bmatrix} \right\} \end{aligned} \tag{A.10}$$

where $\boldsymbol{Z}_1 = \boldsymbol{U}_1^T\boldsymbol{W}$, and $\boldsymbol{Z}_2 = \boldsymbol{U}_2^T\boldsymbol{W}$. From $\boldsymbol{W}^T\boldsymbol{R}_r\boldsymbol{W} > \boldsymbol{0}$, $\boldsymbol{Z}_1$ must be invertible. Therefore, there exists a matrix $\boldsymbol{A} \in \Re^{(n-r)\times r}$ such that $\boldsymbol{Z}_2 = \boldsymbol{A}\boldsymbol{Z}_1$. Equation (A.10) can then be simplified as

$$\begin{aligned} \frac{dL(\boldsymbol{Z})}{dt} &= -\mathrm{tr}\,\{-2\boldsymbol{I}_r + \boldsymbol{Z}_1^T(\boldsymbol{I}_r + \boldsymbol{A}^T\boldsymbol{A})\boldsymbol{Z}_1 \\ &\quad + \boldsymbol{Z}_1^{-1}(\boldsymbol{I}_r + \boldsymbol{A}^T\boldsymbol{\Lambda}_2\boldsymbol{A}\boldsymbol{\Lambda}_1^{-1})^{-1}\boldsymbol{Z}_1^{-T}\}. \end{aligned} \tag{A.11}$$

Now on one hand, if $\boldsymbol{Z}_2(t) \neq \boldsymbol{0}$, we have from Lemma 5.1 that

$$\begin{aligned} \mathrm{tr}\,\{\boldsymbol{Z}_1^T(\boldsymbol{I}_r + \boldsymbol{A}^T\boldsymbol{A})\boldsymbol{Z}_1\} &> \mathrm{tr}\,\{\boldsymbol{Z}_1^T(\boldsymbol{I}_r + \boldsymbol{A}^T\boldsymbol{\Lambda}_2\boldsymbol{A}\lambda_r^{-1})\boldsymbol{Z}_1\} \\ &> \mathrm{tr}\,\{\boldsymbol{Z}_1^T(\boldsymbol{I}_r + \boldsymbol{A}^T\boldsymbol{\Lambda}_2\boldsymbol{A}\boldsymbol{\Lambda}_1^{-1})\boldsymbol{Z}_1\}. \end{aligned} \tag{A.12}$$

Applying (A.12) to (A.11), we have

$$\begin{aligned} \frac{dL(\boldsymbol{Z})}{dt} &< 2r - \mathrm{tr}\,(\boldsymbol{B}) - \mathrm{tr}\,(\boldsymbol{B}^{-1}) \\ &= 2r - \sum_{i=1}^{r}(\mu_i + \mu_i^{-1}) \leq 0 \end{aligned} \tag{A.13}$$

where $\boldsymbol{B} = \boldsymbol{Z}_1^T(\boldsymbol{I}_r + \boldsymbol{A}^T\boldsymbol{\Lambda}_2\boldsymbol{A}\boldsymbol{\Lambda}_1^{-1})\boldsymbol{Z}_1$ and μ_i denote the positive eigenvalues of $\boldsymbol{B}$ for $i = 1, 2, \cdots, r$. On the other hand, if $\boldsymbol{Z}_2(t) = \boldsymbol{0}$, (A.11) becomes

$$\frac{dL(\boldsymbol{Z})}{dt} = 2r - \mathrm{tr}\,(\boldsymbol{Z}_1^T\boldsymbol{Z}_1) - \mathrm{tr}\,[(\boldsymbol{Z}_1^T\boldsymbol{Z}_1)^{-1}] \leq 0. \tag{A.14}$$

Obviously, $(dL(\boldsymbol{W})/dt) = 0$ if and only if $\boldsymbol{Z}_2(t) = \boldsymbol{0}$, and all eigenvalues of $\boldsymbol{Z}_1$ are constant -1 or 1, which indicates $\boldsymbol{U}_1^T\boldsymbol{W}$ must be an orthogonal matrix. Denote this orthogonal matrix by $\boldsymbol{Q}'$. It turns out that $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{Q}'$. For all other cases when $\boldsymbol{W} \in \Omega^*$, it is seen from (A.13) and (A.14) that $(L(\boldsymbol{W})/dt) < 0$. Therefore, the equilibrium $\boldsymbol{W} = \boldsymbol{U}_1\boldsymbol{Q}'$ is asymptotically stable with the domain of attraction Ω. This is equivalent to say that for any $\boldsymbol{W}(0) \in \Omega$, the solution of (37) globally converges to an arbitrary orthonormal basis of the principal subspace. □

Acknowledgment

The authors would like to thank Dr. W.-Y. Yan of Nanyang Technological University, Dr. W. Liu of the University of Western Australia, and Prof. T. Chen of Fudan University for many fruitful discussions regarding the material in Section V. They also thank reviewers for their helpful comments and suggestions.

References

[1] H. Bourlard and Y. Kamp, "Auto-association by the multilayer perceptrons and singular value decomposition," *Biol. Cybern.*, vol. 59, pp. 219–294, 1988.
[2] P. Baldi and K. Hornik, "Learning in linear neural networks: A survey," *IEEE Trans. Neural Networks*, vol. 6, pp. 837–858, July 1995.
[3] P. Comon and G. H. Golub, "Tracking a few extreme singular values and vectors in signal processing," *Proc. IEEE*, vol. 78, pp. 1327–1343, Aug. 1990.
[4] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed. Baltimore, MD: Johns Hopkins Univ. Press, 1989.
[5] R. J. Williams, "Feature discovery through error-correction learning," Inst. Cogn. Sci., Univ. California, San Diego, Tech. Rep. 8501, 1985.
[6] P. Baldi, "Linear learning: Landscapes and algorithms," in *Advances in Neural Information Processing Systems 1*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1989.
[7] L. Xu, "Least mean square error reconstruction principle for self-organizing neural nets," *Neural Networks*, vol. 6, pp. 627–648, 1993.
[8] K. Hornik and C.-M. Kuan, "Convergence analysis of local feature extraction algorithms," *Neural Networks*, vol. 5, pp. 229–240, 1992.
[9] W.-Y. Yan, U. Helmke, and J. B. Moore, "Global analysis of Oja's flow for neural networks," *IEEE Trans. Neural Networks*, vol. 5, pp. 674–683, 1994.
[10] M. Plumbley, "Lyapunov function for convergence of principal component algorithms," *Neural Networks*, vol. 8, pp. 11–23, 1995.

[11] J. Karhunen and J. Joutsensalo, "Generalizations of principal component analysis, optimization problems, and neural networks," *Neural Networks*, vol. 8, pp. 549–562, 1995.
[12] Y. Miao and Y. Hua, "A unified approach to adaptive subspace estimation," in *Proc. Int. Symp. Signal Process. Applicat.*, Brisbane, Australia, Nov. 1996, pp. 680–683.
[13] S. Haykin, *Adaptive Filter Theory*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1991.
[14] S. Bannour and M. R. Azimi-Sadjadi, "Principal component extraction using recursive least squares learning," *IEEE Trans. Neural Networks*, vol. 6, pp. 457–469, Mar. 1995.
[15] S. Y. Kung, K. I. Diamantaras, and J. S. Taur, "Adaptive principal component extraction (APEX) and applications," *IEEE Trans. Signal Processing*, vol. 42, pp. 1202–1217, May 1994.
[16] B. Yang, "Projection approximation subspace tracking," *IEEE Trans. Signal Processing*, vol. 43, pp. 95–107, Jan. 1995.
[17] Z. Fu and E. M. Dowling, "Conjugate gradient eigenstructure tracking for adaptive spectral estimation," *IEEE Trans. Signal Processing*, vol. 43, pp. 1151–1160, May 1995.
[18] G. Mathew, V. U. Reddy, and S. Dasgupta, "Adaptive estimation of eigensubspace," *IEEE Trans. Signal Processing*, vol. 43, pp. 401–411, Feb. 1995.
[19] R. Linsker, "Self-organization in a perceptual network," *IEEE Comput. Mag.*, vol. 21, pp. 105–127, Mar. 1988.
[20] J. R. Magnus and H. Neudecker, *Matrix Differential Calculus with Applications in Statistics and Econometrics*, 2nd ed. New York: Wiley, 1991.
[21] P. A. Cook, *Nonlinear Dynamical Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
[22] J.-F. Yang and M. Kaveh, "Adaptive eigensubspace algorithms for direction or frequency estimation and tracking," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 241–251, Feb. 1988.
[23] E. Oja, "A simplified neuron model as a principal component analyzer," *J. Math. Biol.*, vol. 15, pp. 267–273, 1982.
[24] ———, "Neural networks, principal components, and subspaces," *Int. J. Neural Syst.*, vol. 1, no. 1, pp. 61–68, 1989.
[25] ———, *Subspace Methods for Pattern Recognition*. Letchworth, U.K.: Res. Studies, 1983.
[26] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward network," *Neural Networks*, vol. 2, pp. 459–473, 1989.
[27] J. Karhunen, "Stability of Oja's PCA rule," *Neural Comput.*, vol. 6, pp. 739–747, 1994.
[28] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
[29] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method," *Int. J. Comput. Vis.*, vol. 9, no. 2, pp. 137–154, 1992.
[30] A. Cichocki and R. Unbehauen, "Simplified neural networks for solving a wide variety of matrix algebra problems," *IEEE Trans. Neural Networks*, vol. 5, pp. 910–923, 1994.
[31] Y. Hua and T. K. Sarkar, "On SVD for estimating generalized eigenvalues of singular matrix pencils in noise," *IEEE Trans. Signal Processing*, vol. 39, pp. 892–900, Apr. 1991.
[32] T. Chen, Y. Hua, and W. Yan., "Global convergence of Oja's subspace algorithm for principal component extraction," *IEEE Trans. Neural Networks*, vol. 9, pp. 58–67, Jan 1998.

Yongfeng Miao was born on April 18, 1968, in Shaanxi Province, China. He received the B.S. degree in computer science and the M.E. degree in control from Northwestern Polytechnical University, Xi'an, China, in 1989 and 1991, respectively. He is working toward the Ph.D. degree in the Department of Electrical and Electronic Engineering, University of Melbourne, Melbourne, Australia.

From 1991 to 1994, he was a Faculty Member of the Department of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu, China. In June 1996, he joined Motorola Australia, where he is currently working in telecommunications. His research interests include adaptive signal processing, neural networks, and software engineering in digital signal processing and telecommunications.

Yingbo Hua (S'86–M'88–SM'92) was born in China in 1960. He received the B.S. degree from the Nanjing Institute of Technology (currently Southeast University), Nanjing, China, in February 1982 and the M.S. and Ph.D. degrees from Syracuse University, Syracuse, NY, in 1983 and 1988, respectively.

In February 1990, he joined the University of Melbourne, Parkville, Victoria, Australia, where he was Lecturer from 1990 to 1992, Senior Lecturer from 1993 to 1995, and has been Associate Professor and Reader since January 1996. He has published over 55 journal papers and 90 conference papers in the areas of spectral estimation, array processing, radar and NMR imaging, system identification, neural networks, and wireless communications.

Dr. Hua served as an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1994 to 1997 and is currently a member of the IEEE Signal Processing Society's Technical Committee for Sensor Array and Multichannel Signal Processing.