A Class of Neural Networks for Independent Component Analysis

Juha Karhunen, *Member, IEEE,* Erkki Oja, *Senior Member, IEEE,* Liuyue Wang, Ricardo Vigário, and Jyrki Joutsensalo

***Abstract*—Independent component analysis (ICA) is a recently developed, useful extension of standard principal component analysis (PCA). The ICA model is utilized mainly in blind separation of unknown source signals from their linear mixtures. In this application only the source signals which correspond to the coefficients of the ICA expansion are of interest. In this paper, we propose neural structures related to multilayer feedforward networks for performing complete ICA. The basic ICA network consists of whitening, separation, and basis vector estimation layers. It can be used for both blind source separation and estimation of the basis vectors of ICA. We consider learning algorithms for each layer, and modify our previous nonlinear PCA type algorithms so that their separation capabilities are greatly improved. The proposed class of networks yields good results in test examples with both artificial and real-world data.**

***Index Terms*—Blind source separation, independent component analysis, neural networks, principal component analysis, signal processing, unsupervised learning.**

I. Introduction

MANY meaningful information processing operations can be accomplished using simple neural networks, whose input–output mappings become linear after learning [3]. Several unsupervised learning algorithms of such networks are neural realizations [10], [15], [18], [42] of the widely used statistical technique principal component analysis (PCA). PCA is defined by the eigenvectors of the covariance matrix of the input data. However, these networks are able to perform other tasks [3] such as singular value decomposition, decorrelation, and discriminant analysis, to mention a few. In this paper, we introduce a class of neural networks for independent component analysis (ICA).

ICA is a useful extension of PCA that has been developed in context with blind separation of independent sources from their linear mixtures [13], [19], [24]–[26]. Such blind techniques are needed for example in various applications of array processing, communications, medical signal processing, and speech processing [32]. In a sense, the starting point of ICA is the uncorrelatedness property of standard PCA. Roughly speaking, rather than requiring that the coefficients of a linear expansion of the data vectors be uncorrelated, in ICA they must be mutually independent (or as independent as possible). This implies that higher order statistics are needed in determining the ICA expansion. It also implies that some suitable nonlinearities must be used in the learning phase, even though the final input–output mapping is still linear. As will be seen later on, ICA provides in many cases a more meaningful representation of the data than PCA. This can be utilized for example in linear projection pursuit techniques.

More generally, a recent trend in neural-network research is to study various forms of unsupervised learning beyond PCA. Such techniques are often called "nonlinear PCA" methods [30]. The main reason for this interest is that even though standard PCA is optimal in approximating the input data in the mean-square error sense, the representation that it provides is often not the most meaningful in describing some fundamental characteristics of the data. In PCA, the data are represented in an orthonormal basis determined merely by the second-order statistics (covariances) of the input data. Such a representation is adequate for Gaussian data. However, non-Gaussian data contain a lot of additional information in its higher order statistics. This should be somehow utilized if possible.

Various nonlinear PCA methods (including ICA) have the advantage over standard PCA that they take into account higher order statistics at least implicitly. In nonlinear PCA approaches, a closed-form solution is usually not possible, which makes neural learning algorithms computationally attractive [27], [28], [30]. While PCA is a fairly standardized technique, nonlinear or robust PCA type methods can be developed from various starting points, usually leading to mutually different solutions. In our previous papers [27], [28], [30], [52], [53], we have derived several robust and nonlinear extensions of PCA starting either from maximization of the output variances or from minimization of the mean-square representation error. Some other authors have proposed neural extensions of PCA by choosing optimization of some information-theoretic criterion as their starting point; see [14], [18], [48] for further information. Though generally different, these approaches have relationships to ICA. In particular, we show later on in this paper that some of our previous nonlinear or robust PCA algorithms can be applied to estimating ICA on certain conditions.

In this paper, we introduce neural networks that can be used for both blind source separation and estimation of the basis vectors of ICA. The remainder of this paper is organized as follows. The next section presents in more detail the necessary background on ICA and source separation. In Section III, we introduce and justify the basic ICA network model and its variants. Section IV deals with learning algorithms for each of the three layers of the proposed ICA network. Section V provides mathematical analysis justifying the separation ability

Manuscript received November 18, 1995; revised July 12, 1996 and December 29, 1996.

The authors are with the Helsinki University of Technology, Laboratory of Computer and Information Science, FIN-02150 Espoo, Finland.

Publisher Item Identifier S 1045-9227(97)02750-1.

of a nonlinear PCA-type learning algorithm. In Section IV, we present experimental results for both artificial data and real image and speech data. The last section contains the conclusions of this study, and outlines some possibilities for extending the basic linear data model.

II. Independent Component Analysis and Blind Source Separation

A. The Data Model

In the following, we present the basic data model used in defining both ICA [13], [26] and the source separation problem [26], [35] for linear memoryless channels, and discuss the necessary assumptions. A precise mathematical discussion of ICA is given in Comon's recent fundamental paper [13]. It also contains many references and a numerical batch-type algorithm for estimating the ICA expansion.

Assume that there exist M zero mean source signals $s_k(1), \cdots, s_k(M)$, $k = 1, 2, \cdots$, that are scalar-valued and mutually statistically independent for each sample value k. The independence condition is formally defined so that the joint probability density of the source signals must be the product of the marginal densities of the individual sources

$$p[s_k(1), \cdots, s_k(M)] = \prod_{i=1}^{M} p_i[s_k(i)]. \tag{1}$$

More concretely, the source signals could be sampled speech waveforms; for different speakers the sources are then at least approximately independent. In this case, the index k represents discrete time. In another example, the source signals are two-dimensional discrete images. Here, the index i in the source $s_k(i)$ stands for each of the images, while the two-dimensional index k denotes the pixels. A concrete example of image sources is given in context with experimental results in Section VI.

We assume that the original sources are unobservable, and all that we have are a set of noisy linear mixtures $x_k(1), \cdots, x_k(L)$, with

$$x_k(j) = \sum_{i=1}^{M} s_k(i) a(ij) + n_k(j). \tag{2}$$

The coefficients $a(ij)$ are unknown; however, we assume that the mixtures are all different. Such mixtures arise in several practical situations like speech separation or antenna array processing.

Denote now by $\mathbf{x}_k = [x_k(1), \cdots, x_k(L)]^T$ the L-dimensional kth data vector made up of the mixtures (2) at discrete time (or point) k. By (2), we can write the ICA signal model in vector form as follows:

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k = \sum_{i=1}^{M} s_k(i)\mathbf{a}(i) + \mathbf{n}_k. \tag{3}$$

Here $\mathbf{s}_k = [s_k(1), \cdots, s_k(M)]^T$ is the source vector consisting of the M source signals (independent components) $s_k(i) (i = 1, \cdots, M)$ at the index value k. $\mathbf{A} = [\mathbf{a}(1), \cdots, \mathbf{a}(M)]$ is a constant $L \times M$ "mixing matrix" whose columns $\mathbf{a}(i)$ are the basis vectors of ICA, and $\mathbf{n}_k$ denotes possible corrupting additive noise. The noise term $\mathbf{n}_k$ is often omitted from (3), because it is usually impossible to distinguish noise from the source signals.

We shall consider in more detail the following two related problems. In the first problem, source separation, the task is merely to find the waveforms $\{s_k(i)\}$ of the sources, knowing only the data vectors $\mathbf{x}_k$ and the number M of the sources. The second problem is to estimate the complete ICA expansion, including also the basis vectors $\mathbf{a}(i) (i = 1, \cdots, M)$ of ICA.

The assumptions typically made in ICA and source separation on the model (3) can be listed more precisely as follows [9], [35].

1) $\mathbf{A}$ is a constant matrix with full column rank. Thus the number of sources M is at most equal to L, the dimension of the data vectors $\mathbf{x}_k$. Usually M is assumed to be known in advance. If $M < L$ and there is no noise, the data vectors $\mathbf{x}_k$ lie in the M-dimensional subspace spanned by the basis vectors of ICA.
2) The source signals (coefficients) $s_k(i) (i = 1, \cdots, M)$ must be mutually statistically independent at each sample value k, or in practice as independent as possible. The degree of independence can be measured using suitable contrast functions [13].
3) Each source signal $s_k(i)$ is a stationary zero-mean stochastic process. Only one of the source signals $s_k(i)$ is allowed to have a Gaussian marginal distribution. This restriction essentially follows from the fact that linear combinations of Gaussian source signals are Gaussian. Thus it is impossible to separate several Gaussian sources from each other.

Note that very little prior information is assumed on the matrix $\mathbf{A}$. Therefore, the strong independence assumptions are required in determining the ICA expansion (3). Even then, only the directions of the ICA basis vectors $\mathbf{a}(i), i = 1, \cdots, M$, are defined, because their magnitudes and the amplitudes of the source signals $s_k(i)$ can be interchanged in the model (3). To get a more unique expansion (3), one can normalize the basis vectors $\mathbf{a}(i)$ to unit length and arrange the terms according to the powers $\mathrm{E}\{s_k(i)^2\}$ of the rescaled source signals in a similar way as in standard PCA.

B. ICA, PCA, and Projection Pursuit

At this point, it is useful to compare ICA to standard PCA. In PCA, the data model has the same form (3), but instead of independence the coefficients $s_k(i)$ are required to have sequentially maximal variances (powers) $E\{s_k(i)^2\}$, and the basis vectors $\mathbf{a}(i)$ are constrained to be mutually orthonormal. Usually the basis vectors $\mathbf{a}(i)$ of ICA are not mutually orthogonal. A simple example illustrating the differences of ICA and PCA is given in Fig. 1. The data vectors are assumed to be uniformly distributed inside the parallelogram. The solid lines show the two basis vectors of ICA, and the dashed lines the respective orthogonal basis vectors of PCA. Clearly, the basis vectors of ICA characterize better the data. Other

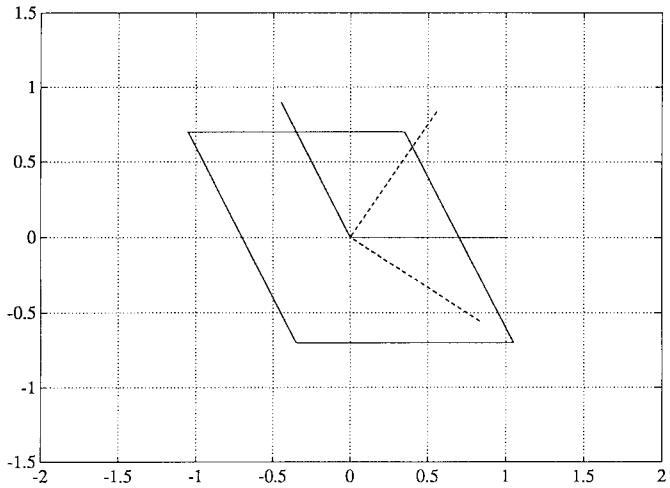

Fig. 1. The theoretical basis vectors of ICA (solid lines) and PCA (dashed lines). In this example, the data vectors are uniformly distributed inside the parallelogram.

examples on comparing PCA with ICA are given in [25] and [26].

On the other hand, the basis vectors of ICA are considerably more difficult to estimate, and their use in technical representation or approximation of the data is not as handy as in PCA because of the nonorthogonal basis. Assume for a while that the basis matrix $\mathbf{A}$ in (3) would be *known*. The standard least-squares solution minimizing the squared modeling error $||\mathbf{n}_k||^2$ is then $\hat{\mathbf{s}}_k = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{x}_k$, yielding the least-squares approximation

$$\hat{\mathbf{x}}_k = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{x}_k \tag{4}$$

of $\mathbf{x}_k$. The projection formula (4) holds for any subspace defined by the columns of $\mathbf{A}$. For PCA, $\mathbf{A}^T\mathbf{A} = \mathbf{I}_M$, where $\mathbf{I}_M$ denotes $M \times M$ unit matrix, leading to the simpler expression $\hat{\mathbf{x}}_k = \mathbf{A}\mathbf{A}^T\mathbf{x}_k$.

The basis vectors of ICA should be especially useful in linear projection pursuit and in extracting characteristic features from natural data [5], [6], [21], [32]. In projection pursuit [16], [20], one tries to describe the structure of high-dimensional data by projecting them onto a low-dimensional subspace and looking for the structure of the projection. In practice, when the data are projected onto a given direction (one-dimensional subspace), the distribution of the projected data is in most cases almost Gaussian. Therefore such directions which give as non-Gaussian projections as possible are often considered to be the most interesting ones, describing well the properties of high-dimensional data. Various indices have been designed for measuring the deviation of a distribution from the Gaussian one. The index used in ICA, maximal independence of the data projected onto a direction, describes well the fundamental characteristics of the data. Projections of the data onto the ICA basis vectors are typically non-Gaussian.

To our knowledge, the use of ICA in projection pursuit has not yet been explicitly considered. However, recently Fyfe and Baddeley [17] have applied a nonlinear (robust) PCA algorithm suggested and derived by us earlier in [27], [41] to finding projection pursuit directions from prewhitened data. In the next sections it will become clear that in [17], the basis vectors of ICA have actually been estimated without an explicit mention. The good results achieved in [17] compared to some other neural projection pursuit algorithms justify the usefulness of ICA in this application.

C. Blind Source Separation

During the last years, techniques called blind source (or signal) separation have been studied especially in signal processing. In blind source separation, one tries to extract the waveforms $\{s_k(i)\}, k = 1, \cdots$, of the independent source signals in (3) from the data vectors $\mathbf{x}_k$. Here the unknown ICA basis vectors $\mathbf{a}(i)$ are usually not of much interest. Such blind techniques are useful for example in array processing, speech enhancement, and communications. If the structure of the matrix $\mathbf{A}$ were known from the problem statement excluding some parameters, more efficient subspace or maximum likelihood type methods [49] are available for estimating the unknown parameters. Both batch type and data-adaptive source separation algorithms have been suggested. With a neural realization in mind, adaptive algorithms that are as simple as possible but yet provide sufficient performance are of primary interest to us.

In adaptive source separation [9], [26], [35], an $M \times L$ separating matrix $\mathbf{B}_k$ is updated so that the M-vector

$$\mathbf{y}_k = \mathbf{B}_k\mathbf{x}_k \tag{5}$$

becomes an estimate $\mathbf{y}_k = \hat{\mathbf{s}}_k$ of the original independent source signals. Under the assumptions 1–3) made before, the estimate $\hat{s}_k(i)$ of the ith source signal may appear in any component $y_k(j)$ of $\mathbf{y}_k$. It is also impossible to determine the amplitudes of the source signals $s_k(i)$ from the model (3) without additional assumptions. Instead of normalizing the basis vectors $\mathbf{a}(i)$, in source separation it is often assumed [9], [35] that each source signal $s_k(i)$ has unit variance.

In several blind separation algorithms, the data vectors $\mathbf{x}_k$ are first preprocessed by whitening (sphering) them, so that their covariance matrix becomes the unit matrix. Various whitening methods are discussed in more detail later on. After prewhitening, the separating matrix $\mathbf{B}_k$ in (5) can be taken orthogonal: $\mathbf{B}_k\mathbf{B}_k^T = \mathbf{I}_M$. This auxiliary constraint which preserves the whiteness property makes the subsequent separating algorithms simpler, and also normalizes the variances of the estimated sources $\hat{s}_k(i)$ automatically to unity.

A practical difficulty in designing source separation and ICA algorithms is reliable verification of the independence condition (1). It is impossible to do this directly or measure the degree of independence using mutual information because the involved probability densities are unknown. Therefore, approximating contrast functions which are maximized by separating matrices have been introduced in [13]. Even these contrast functions require fairly intensive batch type computations using the estimated higher order statistics of the data, or lead to complicated adaptive separation algorithms. Fortunately, it is often sufficient to use the simple higher order statistics called kurtosis, which is a fourth-order cumulant with zero time lags. For the ith source signal $s(i)$, the

(unnormalized) kurtosis is defined by

$$\mathrm{cum}[s(i)^4] = \mathrm{E}\{s(i)^4\} - 3[\mathrm{E}\{s(i)^2\}]^2. \tag{6}$$

If $s(i)$ is Gaussian, its kurtosis $\mathrm{cum}[s(i)^4] = 0$. Source signals that have a negative kurtosis are often called sub-Gaussian ones. Typically, their probability distribution is "flatter" than Gaussian, for example bimodal [17]. Sources with a positive kurtosis (super-Gaussian sources) have usually a distribution which has longer tails and a sharper peak than standard Gaussian distribution [4], [17].

The division of sources into sub-Gaussian and super-Gaussian ones is important, because the separation capability of many algorithms crucially depends on this property. In particular, for prewhitened input vectors it can be shown [37] that the relatively simple contrast function

$$J_1(\mathbf{y}) = \sum_{i=1}^{M} |\mathrm{cum}[y(i)^4]| = \sum_{i=1}^{M} |\mathrm{E}\{y(i)^4\} - 3[\mathrm{E}\{y(i)^2\}]^2| \tag{7}$$

is maximized by a separating matrix $\mathbf{B}$ in the model (5), if the sign of the kurtosis (6) is the same for all the source signals $s_k(i), i = 1, \cdots, M$. For prewhitened input vectors $\mathbf{x}$ and orthogonal separating matrices, the output powers $\mathrm{E}\{y(i)^2\} = 1$, implying that $\mathrm{cum}[y(i)^4] = \mathrm{E}\{y(i)^4\} - 3$. Thus the criterion (7) is maximized if the sum of the fourth moments

$$J_2(\mathbf{y}) = \sum_{i=1}^{M} \mathrm{E}\{y(i)^4\} \tag{8}$$

is *minimized for sources that have a negative kurtosis*, and *maximized for sources with positive kurtosis*. We use the criterion (8) in this paper, because it is simple enough, and can be applied in a straightforward way to our nonlinear PCA type neural learning algorithms.

Instead of optimizing some contrast function, one can use other type of neural algorithms for achieving separation. Most of them have been introduced quite recently. We refer to the tutorial paper [31], where various neural approaches are reviewed. Roughly speaking, neural blind source separation algorithms are often some modifications of the seminal Herault–Jutten (HJ) algorithm [26], [10]. This heuristic algorithm is attractive because it is simple and can be realized locally, but it may fail in separating more than two independent sources. A few new neural separating algorithms [1], [4], [14] have been derived from information theoretic concepts. Also some adaptive blind separation algorithms proposed in the field of signal processing, such as the equivalent adaptive separation via independence (EASI) (or PFS) algorithm [9], [35], can be interpreted as learning algorithms of a neural network.

The performance of a separation algorithm can be studied in test simulations where the mixing matrix $\mathbf{A}$ is known to the user (but not to the learning algorithms) by inspecting the separation results visually. A more quantitative way is to use some performance measure. In [1] the authors define a performance index which measures the difference of the matrix $\mathbf{BA}$ from a permutation matrix. If the basis vectors of ICA are estimated, too, one can for example compute the angles between the true and respective estimated basis vectors. However, in practical situations it is very difficult to assess the results quantitatively, because the true source signals and basis vectors of ICA are unknown.

Generally, it is impossible to separate the possible noise in the input data from the source signals [31]. In practice, noise smears the results in all the separation algorithms. If the amount of noise is considerable, the separation results are often fairly poor. There is not yet any good solution available to this problem. Some of the noise can usually be filtered out using standard PCA if the number of mixtures is larger than the number of sources.

III. The Network Model

A. The ICA Network Structures

Consider now neural estimation of the complete ICA expansion (3). Let us denote the estimated expansion by

$$\mathbf{x}_k = \mathbf{Q}\mathbf{y}_k + \mathbf{n}'_k. \tag{9}$$

Here, the $L \times M$ matrix $\mathbf{Q}$ denotes the estimate of the ICA basis matrix $\mathbf{A}, \mathbf{y}_k$ is the estimate of the source (or independent component) vector $\mathbf{s}_k$, and $\mathbf{n}'_k$ is the noise or error term. The first task is always separation of the sources, or estimation of the vector $\mathbf{y}_k$. As discussed before, this can be done by learning the separating matrix $\mathbf{B}$ in (5) using some suitable algorithm. After this, the components of the vector $\mathbf{y}_k$ should be as independent as possible. For learning the matrix $\mathbf{Q}$, we then simply minimize the mean-square error $\mathrm{E}\{||\mathbf{n}'_k||^2\} = \mathrm{E}\{||\mathbf{x}_k - \mathbf{Q}\mathbf{y}_k||^2\}$ with respect to $\mathbf{Q}$.

This estimation procedure can be realized using the two-layer feedforward network shown in Fig. 2. The L inputs of the network are the components of the vector $\mathbf{x}$ (not counted as a layer). In the hidden layer there are M neurons, and the output layer consists again of L neurons. Let $\mathbf{R}$ denote for clarity the $M \times L$ weight matrix between the inputs and the hidden layer, and $\mathbf{Q}$, respectively, the $L \times M$ weight matrix between the hidden and output layers. Based on above, the ICA expansion (3) can be estimated using the network of Fig. 2 in two subsequent stages as follows.

1. Learn an $M \times L$ weight matrix $\mathbf{R} = \mathbf{B}$ for which the components of $\mathbf{y} = \mathbf{R}\mathbf{x}$ are as independent as possible.
2. Learn an $L \times M$ weight matrix $\mathbf{Q}$ which minimizes the MSE error $\mathrm{E}\{||\mathbf{x} - \mathbf{Q}\mathbf{y}||^2\}$.

If prewhitening is used, the first stage is further divided into two subsequent parts. First, the data (input) vectors $\mathbf{x}_k$ are whitened by applying the transformation

$$\mathbf{v}_k = \mathbf{V}\mathbf{x}_k \tag{10}$$

where $\mathbf{v}_k$ denotes the kth whitened vector, and $\mathbf{V}$ is an $M \times L$ whitening matrix. If $L > M$, $\mathbf{V}$ simultaneously reduces the dimension of the data vectors from L to M. After this, the sources (independent components) are separated

$$\mathbf{y}_k = \mathbf{W}^T\mathbf{v}_k. \tag{11}$$

Here $\mathbf{W}^T$ denotes for clarity the orthonormal ($\mathbf{W}^T\mathbf{W} = \mathbf{I}_M$) $M \times M$ separating matrix that the network should learn. Fig. 3

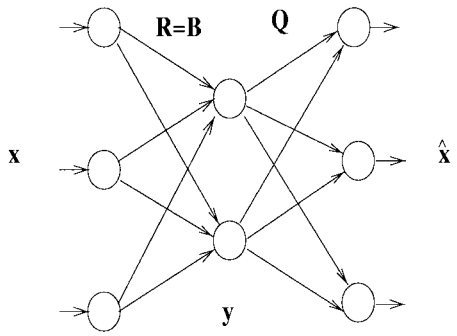

Fig. 2. The linear feedforward network structure. When used as an ICA network, the outputs of the hidden layer are required to be mutually independent.

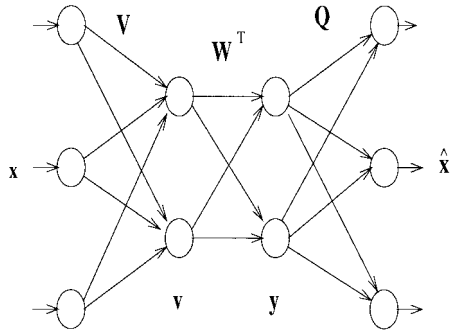

Fig. 3. The proposed ICA network. The network consists of whitening, separation, and basis vector estimation layers. The respective weight matrices are $\mathbf{V}$, $\mathbf{W}^T$, and $\mathbf{Q}$.

shows the ensuing three-layer ICA network structure, where now $\mathbf{R} = \mathbf{B} = \mathbf{W}^T\mathbf{V}$.

In the ICA networks of Figs. 2 and 3, the number of sources M is often equal to L, the dimension of the input vectors. In this case, no data compression takes place in the hidden layer, but the independence constraint anyway provides an ICA solution. As usual, feedback connections (not shown) are needed in the learning phase, but after learning these networks become purely feedforward if the data is stationary. Even though the input–output mappings of the proposed ICA networks are linear after learning due to the linear data model (3), nonlinearities must be used in learning the separating matrix $\mathbf{B}$ or $\mathbf{W}^T$. They introduce higher order statistics into computations, which is necessary in achieving independence. Second-order statistics, which are used in standard PCA, can provide decorrelation only.

The network of Fig. 3 is used in context with our robust or nonlinear PCA learning algorithms, which require whitening of the input data for yielding good separation results. Each of the three layers performs one of the processing tasks required for complete ICA: 1) whitening; 2) separation; and 3) estimation of the basis vectors of ICA. Any of these three tasks can be done either neurally or conventionally. Various possibilities are discussed in the next section. In some separating algorithms the separating matrix $\mathbf{B} = \mathbf{R}$ tries to perform the tasks of whitening, reducing the dimension, and separating the sources simultaneously. One can then use the simpler original network of Fig. 2, but on the other hand the learning algorithms tend to be more complicated. If the task is merely source separation, the last ICA basis vector estimation layer is not needed in these networks.

Before proceeding, we point out that Burel and Rondel have independently proposed a network similar to that in Fig. 2 for blind source separation in [8]. However, their network is intended for array processing applications, and the mixing matrix $\mathbf{A}$ as well as the separating matrix $\mathbf{B}$ have a constrained parametric form.

B. Connection to Linear Autoassociative Network

In this subsection, we try to further clarify the basic ideas behind the introduced ICA networks by studying their connections to the linear autoassociative network [2] which realizes standard PCA.

Consider encoding and decoding of the data vectors $\mathbf{x}$ by using the network of Fig. 2 in autoassociate mode. Here, we first only assume that $\mathbf{Q}$ and $\mathbf{R}$ are $L \times M$ and $M \times L$ constant matrices, respectively. If $M < L$, data compression takes place in the hidden layer, and the output $\hat{\mathbf{x}} = \mathbf{QRx}$ of the network is generally an approximation of the input vector $\mathbf{x}$. Assume that the autoassociative network is trained by minimizing the mean-square approximation (representation) error $J_1(\mathbf{Q}, \mathbf{R}) = \mathrm{E}\{\|\mathbf{x} - \mathbf{QRx}\|^2\}$ using for example the backpropagation algorithm. It is well known [2] that the optimal solution is given by any matrix of the form $\mathbf{R} = (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T$, where the columns of $\mathbf{Q}$ span the M-dimensional PCA subspace of the input vectors $\mathbf{x}$. This subspace is defined by the M principal eigenvectors of the data covariance matrix $\mathrm{E}\{\mathbf{xx}^T\}$ (assuming that $\mathbf{x}$ has zero mean). It should be noted that the projection operator $\mathbf{QR} = \mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T$ onto the PCA subspace is unique, even though the columns of $\mathbf{Q}$ constitute some arbitrary linearly independent basis of the PCA subspace.

Assume now that the noise term $\mathbf{n}_k$ in (3) is standard zero-mean white noise with covariance matrix $\mathrm{E}\{\mathbf{n}_k\mathbf{n}_k^T\} = \sigma^2\mathbf{I}_L$, where σ^2 is the common variance of the components of the vector $\mathbf{n}_k$, and that $\mathbf{n}_k$ is uncorrelated with the sources $s_k(i)$. Under these assumptions, it is easy to show (see, for example, [49]) that the covariance matrix of the data vectors (3) is

$$\mathrm{E}\{\mathbf{x}_k\mathbf{x}_k^T\} = \sum_{i=1}^{M} \mathrm{E}\{s_k(i)^2\}\mathbf{a}(i)\mathbf{a}(i)^T + \sigma^2\mathbf{I}_L. \qquad (12)$$

From the structure of the data covariance matrix (12) one can deduce that the basis vectors $\mathbf{a}(i), i = 1, \cdots, M$, of ICA theoretically lie in its M-dimensional PCA subspace [49]. Usually the basis vectors of ICA are some (unknown) linear combinations of the M first PCA eigenvectors, and vice versa. The PCA and ICA basis vectors coincide only if the basis vectors $\mathbf{a}(1), \cdots, \mathbf{a}(M)$ of ICA happen to be mutually orthonormal, and the variances of the sources $s_k(i)$ are different. In this very specific case, PCA alone can provide separation.

In the ICA networks, we utilize the extra freedom in choosing the matrix $\mathbf{Q}$ (and $\mathbf{R}$). This is done by forcing the network to converge to such a minimizing solution where the columns of $\mathbf{Q}$ not only lie in the PCA subspace, but also coincide with the directions of the desired basis vectors $\mathbf{a}(1), \cdots, \mathbf{a}(M)$ of ICA. Note that if $\mathbf{Q} = \mathbf{A}$, $\hat{\mathbf{x}} = \mathbf{QRx} =$

$\mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{x}$, which is exactly the ICA approximation formula (4). Naturally, the estimated basis vectors need not appear in $\mathbf{Q}$ in the same order as the true ones in $\mathbf{A}$, and their norms can be different.

Clearly, ICA solution can be obtained by imposing the following additional constraint: *the components of the output vector* $\mathbf{y} = \mathbf{R}\mathbf{x}$ *of the hidden layer in the network of Figs. 2 or 3 must be mutually independent (or as independent as possible).* This is the central idea of our ICA networks. Often $M = L$, in which case no data compression takes place. Furthermore, if the networks of Figs. 2 and 3 are used for blind separation only, minimization of the mean-square error is not needed in any phase. In any case, we need not explicitly use the structure $\mathbf{R} = (\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T$ in the proposed ICA networks.

IV. Learning

A. Whitening

Prior to inputting the data vectors $\mathbf{x}_k$ to the ICA networks, they are made zero mean by subtracting the mean, if necessary. This normalizes the data with respect to the first-order statistics. Furthermore, the effects of second-order statistics to the nonlinearities can be removed by whitening the data using the transformation (10). The components of the whitened vectors $\mathbf{v}_k$ must be mutually uncorrelated and normalized in such a way that they have unit variance. This is equivalent to requiring that the covariance matrix $\mathrm{E}\{\mathbf{v}_k\mathbf{v}_k^T\}$ is the unit matrix $\mathbf{I}_M$. Uncorrelatedness is a necessary prerequisite for the stronger independence condition; so after prewhitening the separation task becomes usually somewhat easier. There exist (infinitely) many ways to decorrelate and subsequently whiten the input data (provided that $L \geq M$). Let us briefly discuss some relevant possibilities.

Standard PCA is often used for whitening, because one can then simultaneously compress information optimally in the mean-square error sense and filter possible Gaussian noise. The PCA whitening matrix is given by

$$\mathbf{V} = \mathbf{D}^{-1/2}\mathbf{E}^T. \tag{13}$$

Here the $M \times M$ diagonal matrix $\mathbf{D} = \mathrm{diag}[\lambda(1), \cdots, \lambda(M)]$, and the $L \times M$ matrix $\mathbf{E} = [\mathbf{c}(1), \cdots, \mathbf{c}(M)]$, with $\lambda(i)$ denoting the ith largest eigenvalue of the data covariance matrix $\mathrm{E}\{\mathbf{x}_k\mathbf{x}_k^T\}$, and $\mathbf{c}(i)$ the respective ith principal eigenvector. PCA whitening is easy to do using standard software. This is a preferable way in practice if M is not small and high accuracy is required.

A further advantage of PCA whitening is that standard PCA provides a convenient means for estimating the number M of the sources or independent components. This can be done by estimating all the eigenvalues of the covariance matrix $\mathrm{E}\{\mathbf{x}_k\mathbf{x}_k^T\}$. From the theoretical expression (12) of the covariance matrix it is easy to see [49] that the M largest eigenvalues $\lambda(1), \cdots, \lambda(M)$ of $\mathrm{E}\{\mathbf{x}_k\mathbf{x}_k^T\}$ are some linear combinations of the source signal powers $\mathrm{E}\{s_k(i)^2\}$ added to the noise power σ^2. The remaining $L - M$ eigenvalues correspond to noise only, and are all theoretically equal to σ^2. If the signal-to-noise ratio is good enough, the M largest "signal" eigenvalues are clearly larger than the remaining "noise" eigenvalues. From this, one can deduce the number M of the sources.

Instead of using standard numerical software, one can estimate the principal eigenvectors adaptively. For this purpose, many well-established neural learning algorithms based on the single-unit PCA rule [38] are available [3], [10], [15], [18], [42]. Assuming that the ith weight vector $\mathbf{w}_k(i)$ of a PCA network at step k is a roughly normalized estimate of $\mathbf{c}(i)$, the respective eigenvalue $\lambda(i)$ can be adaptively estimated using the simple algorithm [40]

$$\hat{\lambda}_{k+1}(i) = (1 - \mu_k)\hat{\lambda}_k(i) + \mu_k\left[\mathbf{x}_k^T\mathbf{w}_k(i)\right]^2 \tag{14}$$

where $\mu_k > 0$ is usually a small constant. Recently, Plumbley [45], [46] has introduced neural algorithms that simultaneously whiten the input data and compress them into the PCA subspace. In [46], these approaches are related to the maximization of mutual information in a linear network.

A simple algorithm for learning the whitening matrix $\mathbf{V}_k$ neurally is

$$\mathbf{V}_{k+1} = \mathbf{V}_k - \mu_k\left[\mathbf{v}_k\mathbf{v}_k^T - \mathbf{I}\right]\mathbf{V}_k. \tag{15}$$

This stochastic approximation algorithm has been independently proposed in [35] and [47], and it is used as a part of the EASI (PFS) separation algorithm [9], [35]. In (15), $\mathbf{V}_k$ can be an $M \times L$ matrix with $M < L$. However, the algorithm (15) does not have any optimality properties in data compression, and it sometimes suffers from stability problems. It can be justified by observing that after convergence the vectors $\mathbf{v}_k$ should satisfy the whiteness condition $\mathrm{E}\{\mathbf{v}_k\mathbf{v}_k^T\} = \mathbf{I}_M$.

Generally speaking, separation algorithms using prewhitened data converge faster and have often better stability properties. On the other hand, whitening can make separation of sources more difficult or even impossible if the mixing matrix $\mathbf{A}$ is ill-conditioned or if some of the sources are weak compared to the others [9], [31].

B. Separating Algorithms

The core part and most difficult task in ICA is learning of the separating matrix $\mathbf{B}_k$ in (5). Recall that $\mathbf{B}_k$ can be sought either directly, or using prewhitening in the form $\mathbf{B}_k = \mathbf{W}_k^T\mathbf{V}_k$, where $\mathbf{W}_k^T$ is the orthogonal $M \times M$ separating matrix applied to the whitened vectors $\mathbf{v}_k$. During the last years, many neural blind separation algorithms have been proposed. For a brief review, see [31]. In the following, we discuss and propose separation algorithms which are suitable for learning the matrix $\mathbf{W}_k^T$ or $\mathbf{B}_k$ in PCA-type networks.

In [41], one of the authors proposed two nonlinear extensions of his PCA subspace learning rule which can be applied to learning the orthogonal separating matrix $\mathbf{W}_k$. Consider first so-called *robust PCA subspace rule*

$$\begin{aligned}\mathbf{W}_{k+1} &= \mathbf{W}_k + \mu_k[\mathbf{v}_k - \mathbf{W}_k\mathbf{y}_k]\mathbf{g}(\mathbf{y}_k^T) \\ &= \mathbf{W}_k + \mu_k\left[\mathbf{I} - \mathbf{W}_k\mathbf{W}_k^T\right]\mathbf{v}_k\mathbf{g}(\mathbf{v}_k^T\mathbf{W}_k).\end{aligned} \tag{16}$$

Here and later on $\mathbf{g}(\mathbf{y})$ denotes a vector whose ith component is $g(y(i))$, where $g(t)$ is a nonlinear function. In all these

algorithms, the function $g(t)$ is usually chosen to be odd for stability and separation reasons. The learning parameter μ_k is usually positive, and slowly tends to zero or is a small constant.

In [27], we have shown that (16) is a stochastic gradient algorithm which tries to maximize the criterion $\sum_{j=1}^{M} \mathrm{E}\{f[y(j)]\}$ under the constraint that $\mathbf{W}$ is orthonormal. The function $g(t)$ in (16) is related to the criterion function $f(t)$ so that it is the derivative $df(t)/dt$ of $f(t)$. Thus by choosing $f(t) = t^4/4$ with $g(t) = t^3$, (16) could be used for maximization of the sum of the fourth moments (8), leading to separation of super-Gaussian sources. In principle, the robust PCA subspace rule could be applied to the respective minimization problem by using the negative stochastic gradient with $\mu_k < 0$, but the algorithm is then not stable. However, the sum of fourth moments can be approximately minimized by using the nonlinearity $g(t) = \tanh(t)$ in (16) with $\mu_k > 0$. This follows from the Taylor series expansion $\tanh(t) = t - t^3/3 + 2t^5/15 - \cdots$, where the cubic nonlinearity is dominating if the data are prewhitened [17]. This can be seen by noting that in the Taylor series expansion of the respective criterion function $f(t) = \ln\cosh(t) = t^2/2 - t^4/12 + t^6/45 - \cdots$, the second-order term $t^2/2$ is on an average constant due to the whitening.

In our simulations, the robust PCA subspace rule (16) often worked satisfactorily for two sub-Gaussian sources using $g(t) = \tanh(t)$ and $\mu_k > 0$, but usually not in the case of three sources. The reason is that the algorithm is originally intended for seeking a subspace, or for $L \times M$ matrices $\mathbf{W}$ with $M < L$. If $\mathbf{W}$ is a square $M \times M$ matrix, it tends to unit matrix, and the orthonormalizing term in the square brackets in (16) becomes almost zero before the algorithm learns a separating matrix.

This problem can be circumvented at least in two ways. Instead of (16), we can use the nonlinear PCA subspace rule introduced by one of the authors in [41]; see also [27] and [30]

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k[\mathbf{v}_k - \mathbf{W}_k\mathbf{g}(\mathbf{y}_k)]\mathbf{g}(\mathbf{y}_k^T). \quad (17)$$

The update formula (17) differs slightly from (16), but the nonlinearity inside the square brackets enables in practice the application of this algorithm also to square weight matrices $\mathbf{W}_k$. The columns of $\mathbf{W}_k$ are now not exactly orthonormal. We can anyway justify that the nonlinear PCA subspace rule (17) converges to a separating matrix $\mathbf{W}_k$. This analysis is presented in the next section. Otherwise, the same remarks on minimization and maximization of the sum of fourth moments as for the robust PCA subspace rule hold for (17), too.

We have recently developed another so-called *bigradient algorithm* [52], [53], which is applied for learning the orthonormal separating matrix $\mathbf{W}$ as follows:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k\mathbf{v}_k\mathbf{g}(\mathbf{y}_k^T) + \gamma_k\mathbf{W}_k(\mathbf{I} - \mathbf{W}_k^T\mathbf{W}_k). \quad (18)$$

Here γ_k is another gain parameter, usually about 0.5 or one in practice. The bigradient algorithm is again a stochastic gradient algorithm for maximizing or minimizing the criterion $\sum_{j=1}^{M} \mathrm{E}\{f[y(j)]\}$ under the constraint that the weight matrix $\mathbf{W}$ must be orthonormal. However, the orthonormalization constraint is realized in (18) in an additive way instead of multiplying the gradient $\mathbf{v}_k\mathbf{g}(\mathbf{y}_k^T)$ as in (16). This has two distinct advantages. First, if $\mathbf{W}$ is an approximately orthonormal square $M \times M$ matrix, this does not nullify the effect of gradient learning. Second, (18) can be used directly for the respective minimization problem, because the algorithm is stable also when μ_k is negative. Thus the bigradient algorithm (18) is a better choice than the robust PCA subspace rule in separation. Its disadvantage is that two gain parameters are needed. The algorithm (18) is derived and discussed in more detail in [52] and [53].

In all the above algorithms, the gradient $\mathbf{v}_k\mathbf{g}(\mathbf{y}_k^T)$ is essentially responsible for learning the separating matrix. It can be interpreted as a nonlinear Hebbian learning term. The other terms appearing in the algorithms (16)–(18) have the role of stabilizing and normalizing the matrix $\mathbf{W}$.

Cardoso and Laheld recently introduced so-called *EASI (or PFS) algorithm* [9], [35], where the total separating matrix is computed from the formula

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \mu_k\left[\frac{\mathbf{y}_k\mathbf{y}_k^T - \mathbf{I}}{1 + \mu_k\mathbf{y}_k^T\mathbf{y}_k} + \frac{\mathbf{g}(\mathbf{y}_k)\mathbf{y}_k^T - \mathbf{y}_k\mathbf{g}(\mathbf{y}_k^T)}{1 + \mu_k\mathbf{y}_k^T\mathbf{g}(\mathbf{y}_k)}\right]\mathbf{B}_k. \quad (19)$$

The scalar terms in the denominator are needed to stabilize the learning rule (19) in practice, if the nonlinearity $g(t)$ grows faster than linearly. Otherwise, they can usually be omitted provided that μ_k is not too large. A similar stabilization can be applied to the other separation algorithms described in this subsection if necessary. In [9] and [35], the EASI algorithm (19) is actually derived by first whitening the data vectors; this yields the "linear" whitening part $\mathbf{y}_k\mathbf{y}_k^T - \mathbf{I}$ in (19). After this, separation is achieved by minimizing $\sum_{i=1}^{M} \mathrm{E}\{y(i)^4\}$. After heavy approximations, this yields the nonlinear separating part $\mathbf{g}(\mathbf{y}_k)\mathbf{y}_k^T - \mathbf{y}_k\mathbf{g}(\mathbf{y}_k^T)$, where $g(t) = t^3$. These parts are combined in (19) in an elegant way. In practice, other nonlinearities than just $g(t) = t^3$ are often able to learn a separating matrix in (19).

The learning rule (19) is introduced in [9] and [35] as an adaptive source separation algorithm without any reference to neural networks. However, especially in its unstabilized form (19) can directly be used as a learning algorithm of a PCA type network. For introducing even more higher order statistics, we can use another odd nonlinear function $h(t)$ in (19). This leads to a kind of generalized EASI (or PFS) algorithm

$$\mathbf{B}_{k+1} = \mathbf{B}_k - \mu_k\left[\mathbf{y}_k\mathbf{y}_k^T - \mathbf{I} + \mathbf{g}(\mathbf{y}_k)\mathbf{h}(\mathbf{y}_k^T) - \mathbf{h}(\mathbf{y}_k)\mathbf{g}(\mathbf{y}_k^T)\right]\mathbf{B}_k \quad (20)$$

which is stabilized in a similar way as (19) if necessary. A heuristic justification to the algorithm (20) is that it should after convergence satisfy on an average the conditions $\mathrm{E}\{\mathbf{y}_k\mathbf{y}_k^T\} = \mathbf{I}$ (whiteness) and $\mathrm{E}\{\mathbf{g}(\mathbf{y}_k)\mathbf{h}(\mathbf{y}_k^T)\} = \mathrm{E}\{\mathbf{h}(\mathbf{y}_k)\mathbf{g}(\mathbf{y}_k^T)\}$ (a kind of independence condition).

The mutual performance of the separation algorithms discussed above depends on a number of factors such as the mixing matrix, source signals, chosen parameters, and nonlinearities. In our experiments, the bigradient algorithm often converged faster than the nonlinear PCA subspace rule, but it seems to be more sensitive to the choice of gain parameters. An advantage of the nonlinear PCA subspace rule is that it

can be realized fairly easily using hardware, for example by modifying slightly the implementation presented in [34] for the standard PCA subspace rule. The EASI learning algorithm (and its generalization) usually perform rather similarly than the other learning rules, but it can separate the sources even though the mixing matrix is ill-conditioned provided that there is no noise [9], [31].

A practical, ever-present problem with the above type of simple stochastic algorithms is the choice of the learning parameter(s). Some hints on how to do this are given in [27]. Generally, the learning parameters should be the smaller the larger is the relative magnitude of the update term. For nonlinearities growing faster than linearly, special measures like the denominator terms in (19) are often needed to ensure the stability of the algorithm. For the robust PCA subspace rule (16), the first author has derived a bound ensuring the stability of the algorithm in [29]. This can be applied as a good approximation to the nonlinear PCA subspace rule (17), too. Quite recently, we have introduced in [33] adaptive least-squares type algorithms for minimizing the same criterion function from which the nonlinear PCA subspace rule (17) has been derived in [27]. These algorithms are somewhat more complicated than the simple stochastic gradient algorithms discussed above. Their great advantage is that the learning parameter is determined automatically from the input data so that it is roughly optimal, resulting in a fast convergence [33].

Another problem that is often encountered in practice is that the input data are not stationary. Either the source signals may be nonstationary or the mixing matrix $\mathbf{A}$ changes with time or both. We have made some simulations on this situation, too, showing that the presented algorithms are able to track at least slow changes in the data provided that the learning parameters are chosen suitably. More detailed results on this case will be presented elsewhere.

Finally, we emphasize that *any* suitable algorithm can be used for learning the separating matrix $\mathbf{W}_k^T$ in the network of Fig. 3 or the matrix $\mathbf{B}_k$ in the network of Fig. 2. The above algorithms have been proposed and discussed because they can be regarded neural, are applicable in PCA type networks, and are among the simplest available to our knowledge. All these algorithms require that the original source signals have a kurtosis with the same sign: sgn(cum$[s(i)^4]) = +1$ or -1 for $i = 1, \cdots, M$. This condition can be mildened in the EASI algorithm somewhat so that the sum of kurtosises for any pair of two sources must have the same sign [9]. This means that the kurtosis of one source may have an opposite sign if its absolute value is the smallest. The same condition seems to hold for the other discussed algorithms in practice.

The condition on the signs of the kurtosis has been recently removed in one-unit learning algorithms [22], [23] that are able to find one source signal at a time, and have computationally efficient fixed-point variants.

C. Estimation of the Basis Vectors of ICA

The task of the last layer in the networks of Fig. 3 or Fig. 2 is to estimate the basis vectors $\mathbf{a}(i), i = 1, \cdots, M$, of ICA. We first present the standard nonneural solution to this problem and then a neural learning algorithm for the weight matrix $\mathbf{Q}$.

Assuming that the matrix $\mathbf{B}_k$ has converged to a separating solution $\mathbf{B}$, the basis vectors of ICA can be estimated by using the theory of pseudoinverses. If $\mathbf{x}_k$ is solved directly from (5), in the general case $L > M$ there exist infinitely many possible solutions. They all can be represented in the form

$$\mathbf{x}_k = \mathbf{B}^T\left(\mathbf{B}\mathbf{B}^T\right)^{-1}\mathbf{y}_k + [\mathbf{I} - \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}\mathbf{B}]\mathbf{z} \tag{21}$$

where $\mathbf{z}$ is an arbitrary vector having the same dimension L as $\mathbf{x}_k$. Clearly, the $\mathbf{z}$ part of the general solution (21) is not interesting, because it does not depend on the estimated source vector $\mathbf{y}_k$, and represents the portion of possible solutions that lie in the subspace orthogonal to the rows of $\mathbf{B}$. Setting $\mathbf{z} = \mathbf{0}$ in (21) yields the remaining meaningful part, which is the unique minimum-norm (pseudoinverse) solution

$$\hat{\mathbf{x}}_k = \hat{\mathbf{A}}\mathbf{y}_k = \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}\mathbf{y}_k = \sum_{j=1}^{M} \hat{s}_k(j)\hat{\mathbf{a}}(j). \tag{22}$$

Here $\hat{\mathbf{a}}(j)$ denotes the jth column of the $L \times M$ matrix $\hat{\mathbf{A}} = \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}$. Comparing this with the ICA expansion (3), we see that the vectors $\hat{\mathbf{a}}(j)$ are the desired estimates of the basis vectors of ICA. They can be normalized and ordered suitably.

If PCA whitening (13) is used, the estimated ICA basis matrix $\hat{\mathbf{A}} = \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}$ can be simplified to the form

$$\hat{\mathbf{A}} = \mathbf{E}\mathbf{D}^{1/2}\mathbf{W}. \tag{23}$$

Thus the unnormalized ith basis vector of ICA is $\hat{\mathbf{a}}(i) = \mathbf{E}\mathbf{D}^{1/2}\mathbf{w}(i)$, where $\mathbf{w}(i)$ is the ith column of $\mathbf{W}$, and its squared norm becomes $||\hat{\mathbf{a}}(i)||^2 = \mathbf{w}(i)^T\mathbf{D}\mathbf{w}(i)$. Due to the diagonality of the eigenvalue matrix $\mathbf{D}$, these expressions are somewhat easier to compute than (22), but still require square rooting and knowledge of the principal eigenvalues and -vectors of the covariance matrix of the input data.

A completely neural algorithm for estimating the basis vectors of ICA which does not require any inversion of matrices or square rooting can be developed as follows. Recall from Section III that the columns of the weight matrix $\mathbf{Q}$ become estimates of the basis vectors of ICA, if the mean-square error $\mathrm{E}\{||\mathbf{x} - \mathbf{Q}\mathbf{y}||^2\}$ is minimized under the constraint that the components of the vector $\mathbf{y}$ are statistically mutually independent. Assume now that as a result of the whitening and separation stages, the matrix $\mathbf{B}_k$ has converged to a separating solution $\mathbf{B}$, and the components of $\mathbf{y}$ are as independent as possible. Then it suffices to search for the matrix $\mathbf{Q}$ which minimizes the mean-square error.

Omitting the expectation, the gradient of $||\mathbf{x} - \mathbf{Q}\mathbf{y}||^2$ with respect to $\mathbf{Q}$ is $-2(\mathbf{x} - \mathbf{Q}\mathbf{y})\mathbf{y}^T$, which in a standard way yields the stochastic gradient algorithm

$$\mathbf{Q}_{k+1} = \mathbf{Q}_k + \mu_k(\mathbf{x}_k - \mathbf{Q}_k\mathbf{y}_k)\mathbf{y}_k^T \tag{24}$$

$(\mu_k > 0)$ for learning the matrix $\mathbf{Q}$. Here the coefficient 2 has been absorbed in the learning parameter μ_k for convenience. This algorithm can be used for estimating the basis vectors

of ICA in context with any suitable separation algorithm. Naturally, the matrix $\mathbf{Q}$ could be learned by minimizing the MSE error using more complicated but faster converging algorithms. In large dimensional problems or in cases where the directions of the some of the basis vectors of ICA are close to each other, it may be better to use the nonneural formulas (22) or (23) for achieving sufficient accuracy.

Finally, consider estimation of the basis vectors of ICA in the specific situation where the data are already white: $\mathrm{E}\{\mathbf{v}_k\mathbf{v}_k^T\} = \mathbf{I}_M$. Then $\mathbf{B} = \mathbf{W}^T$, which yields $\hat{\mathbf{A}} = \mathbf{W}$ due to the orthonormality of $\mathbf{W}$. Thus the basis vectors of ICA are directly the columns of the separating matrix $\mathbf{W}$ for whitened data. This explains the good exploratory projection pursuit results achieved in [17]. The authors first take whitened multidimensional Gaussian data, then make the data non-Gaussian in some direction, and finally estimate this direction using the robust PCA subspace rule (16). From the above discussion, we can conclude that the found direction is nothing but a basis vector of ICA. In these experiments, the projection of the data onto this direction is furthest from the Gaussian distribution.

Quite recently, the basis vectors of ICA have been estimated for real-world image and sound data in [5], [6], and [21], showing their relevance in characterizing natural data.

V. Mathematical Analysis

The bigradient algorithm (18) [as well as the robust PCA subspace rule (16)] has been derived by optimizing the criterion $\sum_{j=1}^{M} \mathrm{E}\{f[y(j)]\}$ under orthonormality constraints, and so it can be expected to converge to an orthogonal matrix $\mathbf{W}$ that will minimize or maximize the criterion (depending on the sign of the learning parameter μ_k). We have presented mathematical analysis supporting this in [52], showing that in the standard PCA/MCA case in which $f[y(j)] = y(j)^2$, the asymptotic convergence points are indeed the desired eigenvectors. The EASI algorithm (19) has been analyzed in [9]. However, the nonlinear PCA subspace rule (17) is only indirectly related to an optimization criterion [30], and so a convergence analysis should be given. This section provides some results on the asymptotic solutions of the nonlinear PCA subspace rule. We first present a mathematical theorem, and then study in more detail two interesting special cases, where the function $g(t)$ is either a polynomial or a sigmoid.

A. Asymptotic Analysis of the Nonlinear PCA Subspace Rule

We start from the learning rule (17)

$$\mathbf{W}_{k+1} = \mathbf{W}_k + \mu_k[\mathbf{v}_k - \mathbf{W}_k\mathbf{g}(\mathbf{y}_k)]\mathbf{g}(\mathbf{y}_k^T)$$

with $\mathbf{y}_k = \mathbf{W}_k^T\mathbf{v}_k$. The input vectors $\mathbf{v}_k$ are whitened: $\mathrm{E}\{\mathbf{v}_k\mathbf{v}_k^T\} = \mathbf{I}$, and we assume that there exists a square separating matrix $\mathbf{H}^T$ such that the vector $\mathbf{u}_k = \mathbf{H}^T\mathbf{v}_k$ has independent elements and also unit variances: $\mathrm{E}\{\mathbf{u}_k\mathbf{u}_k^T\} = \mathbf{I}$. This implies that the separating matrix $\mathbf{H}^T$ must be orthogonal. Because our aim is to show that the weight matrix $\mathbf{W}$ converges to a separating matrix (transposed), we do not make any prior assumptions on the separation properties of $\mathbf{W}$ here.

To make the analysis easier, we multiply both sides of the learning rule (17) by $\mathbf{H}^T$. We obtain [43]

$$\begin{aligned}\mathbf{H}^T\mathbf{W}_{k+1} &= \mathbf{H}^T\mathbf{W}_k + \mu_k\big[\mathbf{H}^T\mathbf{v}_k - \mathbf{H}^T\mathbf{W}_k\mathbf{g}\big(\mathbf{W}_k^T\mathbf{v}_k\big)\big] \\ &\quad \times \mathbf{g}\big(\mathbf{v}_k^T\mathbf{W}_k\big) \\ &= \mathbf{H}^T\mathbf{W}_k + \mu_k\big[\mathbf{H}^T\mathbf{v}_k - \mathbf{H}^T\mathbf{W}_k\mathbf{g}\big(\mathbf{W}_k^T\mathbf{H}\mathbf{H}^T\mathbf{v}_k\big)\big] \\ &\quad \times \mathbf{g}\big(\mathbf{v}_k^T\mathbf{H}\mathbf{H}^T\mathbf{W}_k\big) \qquad (25)\end{aligned}$$

where we have used the fact that $\mathbf{H}\mathbf{H}^T = \mathbf{I}$. Denoting for the moment $\mathbf{S}_k = \mathbf{H}^T\mathbf{W}_k$ and using the definition $\mathbf{u}_k = \mathbf{H}^T\mathbf{v}_k$ given above, we have

$$\mathbf{S}_{k+1} = \mathbf{S}_k + \mu_k\big[\mathbf{u}_k - \mathbf{S}_k\mathbf{g}\big(\mathbf{S}_k^T\mathbf{u}_k\big)\big]\mathbf{g}\big(\mathbf{u}_k^T\mathbf{S}_k\big). \qquad (26)$$

This equation has exactly the same form as the original one. Geometrically the transformation by the orthogonal matrix $\mathbf{H}$ simply means a rotation to a new set of coordinates such that the elements of the input vector expressed in these coordinates are statistically independent. If $\mathbf{S}_k$ tends to a scaled version of the unit matrix, then in the original nonlinear PCA subspace rule (17) $\mathbf{W}_k = \mathbf{H}\mathbf{S}_k$ tends to a similarly scaled version of the separating matrix $\mathbf{H}$.

To show this, the difference equation (26) can be further analyzed by writing down the corresponding averaged differential equation; for a discussion of this technique, see e.g., [39]. The limit of convergence of the difference equation is among the asymptotically stable solutions of the averaged differential equation. Taking averages in (26) with respect to the density of $\mathbf{u}_k$, and using $\mathbf{Z} = \mathbf{Z}(t)$ as the continuous-time counterpart of the transformed weight matrix $\mathbf{S}_k$, we obtain

$$d\mathbf{Z}/dt = \mathbf{G}(\mathbf{Z}) - \mathbf{Z}\mathbf{H}(\mathbf{Z}) \qquad (27)$$

with

$$\mathbf{G}(\mathbf{Z}) = \mathrm{E}\{\mathbf{u}\mathbf{g}(\mathbf{u}^T\mathbf{Z})\} \qquad (28)$$

$$\mathbf{H}(\mathbf{Z}) = \mathrm{E}\{\mathbf{g}(\mathbf{Z}^T\mathbf{u})\mathbf{g}(\mathbf{u}^T\mathbf{Z})\}. \qquad (29)$$

The expectations are over the (unknown) density of vector $\mathbf{u}$. We are ready to state the main result of this section, which is a simplified version of a more general theorem originally presented by one of the authors in [43]

Theorem: In the matrix differential equation (27), assume the following.

1) The random vector $\mathbf{u}$ has a symmetrical density with $\mathrm{E}\{\mathbf{u}\} = 0$.
2) The elements of $\mathbf{u}$, denoted here $u_1, \cdots, u_n$, are statistically mutually independent and all have the same density.
3) The function $g(.)$ is odd, that is $g(t) = -g(-t)$ for all t, and at least twice differentiable everywhere.
4) The function $g(.)$ and the density of $\mathbf{u}$ are such that the following conditions hold:

$$\begin{aligned}A &= \mathrm{E}\{u^2g'(\alpha u)\} - 2\alpha\mathrm{E}\{g(\alpha u)g'(\alpha u)u\} \\ &\quad - \mathrm{E}\{g^2(\alpha u)\} < 0 \qquad (30)\end{aligned}$$

where $g'(t)$ is the derivative $g(t)$ and α is a scalar satisfying

$$\mathrm{E}\{ug(\alpha u)\} = \alpha\mathrm{E}\{g^2(\alpha u)\}. \qquad (31)$$

5) The following condition holds:

$$\mathrm{E}\{u^2\}\mathrm{E}\{g'(\alpha u)\} - \mathrm{E}\{g^2(\alpha u)\} < 0. \tag{32}$$

Then the matrix

$$\mathbf{Z} = \mathbf{D} = \mathrm{diag}(\alpha, \cdots, \alpha) = \alpha\mathbf{I} \tag{33}$$

is an asymptotically stable stationary point of (27), where α is the positive solution to (31).

The proof is given in [43].

Note 1: We only consider a diagonal matrix $\mathbf{D} = \alpha\mathbf{I}$ as the asymptotically stable solution. However, any permutation of $\mathbf{D}$ can be shown to be an asymptotically stable solution, too, by making another orthogonal rotation of the coordinate axes that will permute some of them. This simply means reindexing of the vector elements u_i. Mathematically, by replacing $\mathbf{Z}(t)$ with $\mathbf{PZ}(t)$, where $\mathbf{P}$ is a permutation (an orthogonal matrix), an analogous differential equation is obtained, and the conditions of Theorem 1 are unaltered.

Note 2: Conditions 4) and 5) are the technical requirements for asymptotic stability. Clearly, substituting matrix (33) into the fixed point condition $\mathbf{G}(\mathbf{Z}) = \mathbf{ZH}(\mathbf{Z})$ gives (31). The inequalities (30) and (32) are sufficient for asymptotic stability, as shown in [43].

Note 3: Due to the oddity of function $g(t)$, the signs of the α cannot be determined from (31); if $+\alpha$ is a solution, then so is also $-\alpha$. If the weight matrix $\mathbf{S}_k$ of (26) converges to $\mathbf{D}$, then asymptotically the ith element of the vector $\mathbf{y}_k = \mathbf{Du}_k$ is the ith element of the vector $\mathbf{u}_k$ multiplied by $\pm\alpha$. The sign has no influence on the absolute magnitude. For the negative α, a similar result to the above holds.

Note 4: Theorem 1 allows nonmonotonic learning functions. However, if $g(t)$ is monotonic, then (31) in fact implies that it must be an increasing function. If $g(t)$ were monotonically decreasing and odd, then the left-hand side would be negative for positive α and positive for negative α; but then there could not be any solution because $\mathrm{E}\{g^2(\alpha u)\} > 0$.

The Theorem 1 will now be illustrated for two specific types of nonlinear learning functions: polynomial functions $g(t) = t^s$, with s an odd positive integer, and sigmoidal functions $g(t) = \tanh(\beta t)$, with β a positive slope parameter. All these functions obviously satisfy the condition 3) of the Theorem. For more details, see [43].

B. Special Case: Polynomials

The family of odd polynomial functions

$$g(t) = t^s, \quad s = 1, 3, 5, 7, \cdots \tag{34}$$

is interesting in the present context because all the relevant variables in the conditions 4) and 5) of Theorem 1 will become *moments* of u for any probability density. These functions include the linear function for which $s = 1$.

First, we get from (31) for α

$$\mathrm{E}\{u^{s+1}\} = \alpha^{s+1}\mathrm{E}\{u^{2s}\}. \tag{35}$$

Substituting this in (30), we find that the condition 4) is always satisfied.

The stability condition 5) of Theorem 1 now becomes

$$\mathrm{E}\{u^{s+1}\} - s\mathrm{E}\{u^2\}\mathrm{E}\{u^{s-1}\} > 0. \tag{36}$$

Consider first the case

$$s = 1, \quad g(u) = u. \tag{37}$$

Clearly, the condition (36) is not satisfied. *The linear function never gives asymptotic stability*. Consider next the case

$$s = 3, \quad g(u) = u^3. \tag{38}$$

Now (36) gives

$$\mathrm{E}\{u^4\} - 3(\mathrm{E}\{u^2\})^2 > 0. \tag{39}$$

This expression is exactly the *kurtosis* or the fourth-order cumulant of u. If and only if the density is *positively kurtotic* or *super-Gaussian*, this condition is satisfied and the cubic polynomial $g(u) = u^3$ *gives asymptotic stability*.

Likewise, for $s = 5$ we get the condition

$$\mathrm{E}\{u^6\} - 5\mathrm{E}\{u^2\}\mathrm{E}\{u^4\} > 0 \tag{40}$$

and so on.

C. Special Case: Hyperbolic Tangents

Consider then the sigmoidal learning function $g(t) = \tanh(\beta t)$ where $\beta > 0$. It asymptotically approaches the hard-limiting function sign(t) as $\beta \to \infty$. Assuming $\sigma^2 = 1$, the stability condition 5) of the Theorem 1 becomes $\mathrm{E}\{g'(\alpha u)\} < \mathrm{E}\{g^2(\alpha u)\}$. For the hyperbolic tangent, $g'(t)$ has a peak around the origin and decreases to both sides, while $g^2(t)$ is zero at the origin and increases to both sides. In this case it is clear that a peaked super-Gaussian density of u makes $\mathrm{E}\{g'(\alpha u)\}$ large and $\mathrm{E}\{g^2(\alpha u)\}$ small, while a flat sub-Gaussian density does just the opposite. The latter case is then more stable.

A simple example of a sub-Gaussian density is the uniform density on the interval [−1,1]. Let us assume this for the elements of the vector $\mathbf{u}$ to illustrate the Theorem 1. Condition 1 of Theorem 1 is then satisfied. It remains to check the stability conditions 4) and 5) of Theorem 1. Now, a closed form solution for α in (31) is not feasible, and numerical methods must be used. It turns out that condition 5) holds for $\beta > 0$ (for details, see [43]), and condition 4) is always satisfied. The conclusion is that *for the uniform density the sigmoidal function gives asymptotic stability* when $\beta > 0$.

Asymptotic stability is a local effect, and Theorem 1 does not say anything about the basin of attraction of the asymptotic solution, that is, *global stability*. We have tested the global stability of the differential equation (27) numerically. In these simulations the input data were three-dimensional, each element having an identical uniform density, the sigmoid parameter had the value $\beta = 5.0$, and the initial deviation of $\mathbf{Z}(0)$ from the theoretical limit was varied. For this value of β, and the uniform densities used, $\alpha = 0.6998$. The deviation was increased up to 100.0 and the algorithm converged invariably to the asymptotically stable solution $\mathbf{D}$ predicted by Theorem 1 or to its variation. This means that when the initial deviation

is increased, it may happen that the asymptotic limit for $\mathbf{Z}(t)$ will not be $\mathbf{D} = \mathrm{diag}(\alpha, \alpha, \alpha)$ but a permutation with possibly changed signs. Thus for example for the initial value

$$\mathbf{Z}(0) = \begin{pmatrix} -6.1953 & 0.2556 & -0.0945 \\ 6.3493 & -2.1398 & -3.6694 \\ 0.0710 & 7.6358 & -5.7220 \end{pmatrix} \tag{41}$$

the asymptotic value turned out to be

$$\lim \mathbf{Z}(t) = \begin{pmatrix} -0.6998 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & -0.6998 \\ 0.0000 & 0.6998 & 0.0000 \end{pmatrix} \tag{42}$$

which is a permutation of matrix $\mathbf{D}$. Note also the negative sign in two of the nonzero elements.

The overall conclusion of this section is that, while the nonlinear PCA subspace rule (17) is not directly a gradient learning algorithm for a cost function, its limits of convergence are nevertheless separating matrices, if the nonlinear learning function $g(t)$ is adapted to the original densities of the source signals, especially to the sign of the kurtosis.

VI. Experimental Results

In this section, we demonstrate the performance of the ICA network of Fig. 3 using both artificial and real-world data. Artificially generated data is useful because it makes possible to compare the estimation results with theoretically correct values. In Section IV, several alternative learning algorithms or estimation procedures were given for each of the three layers in the network. We have not tested all combinations of them; however, at least two different learning methods have been used for each layer for confirming the generality of the proposed structure.

Comon's Data: Consider first a test example used earlier by Comon [12]. Here, the three original source signals $s_k(1)$, $s_k(2)$, and $s_k(3)$ in (3) consist of uniformly distributed noise, a ramp signal, and a pure sinusoid. Fig. 4 shows 100 samples of them. Actually two of the source signals are deterministic waveforms, allowing easy visual inspection of the separation results. All the three sources have a negative kurtosis. Fig. 5 depicts the respective components of the three-dimensional data vectors $\mathbf{x}_k$, which are linear mixtures of the source signals. They were formed using the linear ICA model (3), where the true normalized basis vectors of ICA were $\mathbf{a}(1) = [0.0891, -0.8909, 0.4454]^T$, $\mathbf{a}(2) = [0.3906, -0.6509, 0.6509]^T$, and $\mathbf{a}(3) = [-0.3408, 0.8519, -0.3976]^T$, and the noise term $\mathbf{n}_k$ was zero.

We chose the simplest learning algorithms, so that (15) was used for whitening, the nonlinear PCA subspace rule (17) for separation, and (24) for estimating the basis vectors of ICA. The 100 data vectors were used 60 times sequentially in teaching the ICA network of Fig. 3. The learning parameter μ_k was 0.01 both in (15) and (17). The learning function was $g(t) = \tanh(t)$. After teaching, the data vectors $\mathbf{x}_k, k = 1, \cdots, 100$, were inputted to the network of Fig. 3. Fig. 6 shows the separated signals $y_k(1)$, $y_k(2)$, and $y_k(3)$ (outputs of the second layer), which are good estimates of the original source signals. In the last layer of the ICA network, the algorithm (24) learned a matrix $\mathbf{Q}$ whose normalized columns $\hat{\mathbf{a}}(1) = [-0.1080, 0.8916, -0.4397]^T$, $\hat{\mathbf{a}}(2) = [0.3917, -0.6542, 0.6470]^T$, and $\hat{\mathbf{a}}(3) = [0.3325, -0.8507, 0.4072]^T$ estimate well the true basis vectors of ICA.

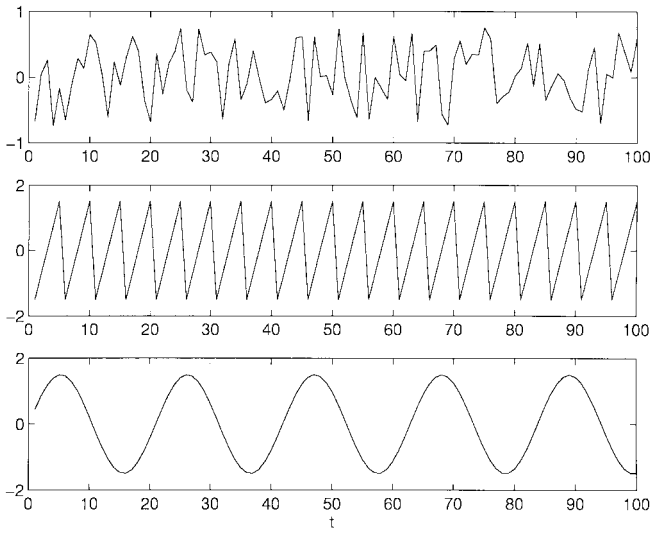

Fig. 4. One hundred samples of three source signals in Comon's example: Uniformly distributed noise, a ramp signal, and a pure sinusoid.

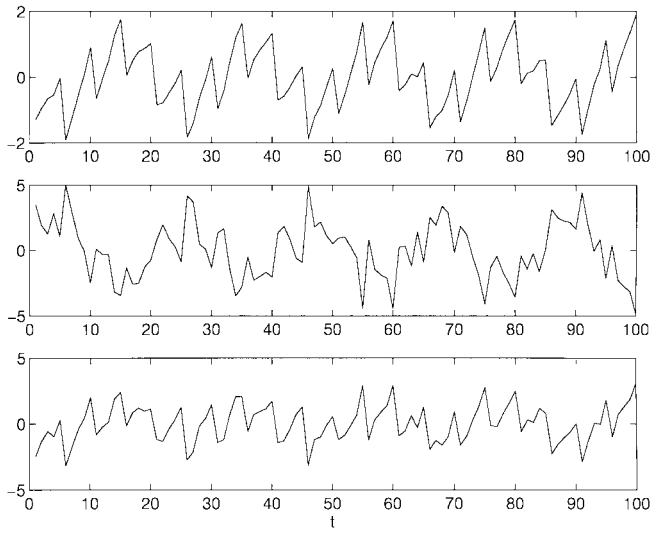

Fig. 5. Components of the 100 data vectors used in the simulation. They are linear combinations of the source signals shown in Fig. 4.

The results were roughly similar, when the bigradient algorithm (18) was used for estimating the separating matrix $\mathbf{W}^T$ with the same learning function and parameters, or alternatively using the learning function $g(t) = t^3$ and learning parameter $\mu_k = -0.003$. The other parameter γ was 0.9. Also the EASI (PFS) algorithm (19) performs well with suitable choices.

Parallelogram Data: As a second example, consider the performance of the generalized EASI-type algorithm (20) in estimating the basis vectors of ICA in a simple but illustrative test case. In Fig. 7, the dots are data vectors uniformly distributed inside the parallelogram, and the solid lines are the true basis vectors of ICA. In this case, the independent

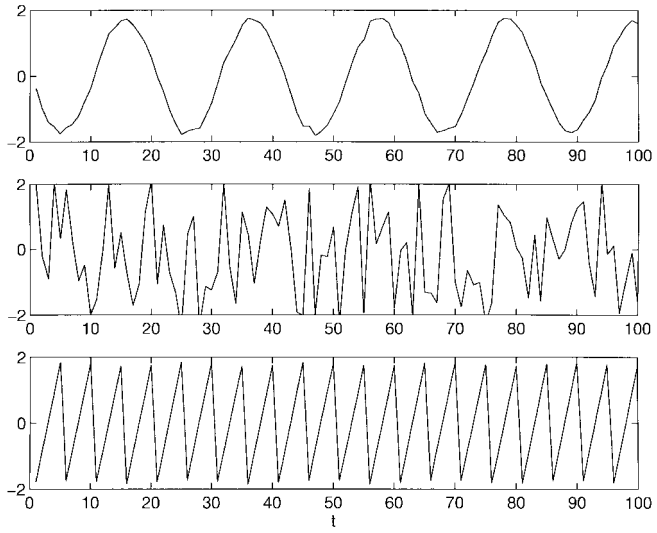

Fig. 6. The separated signals $y_k(1)$, $y_k(2)$, and $y_k(3)$ after learning.

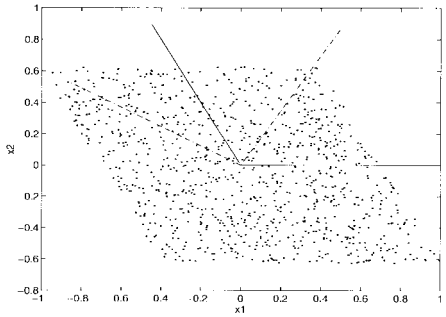

Fig. 7. The parallelogram data and the basis vectors of ICA (solid lines) and DCA (dashed lines).

components (projections of the data vectors onto the basis vectors of ICA) have a negative kurtosis. The dashed lines in Fig. 7 represent basis vectors which define a kind of opposite of ICA: when the data vectors are projected onto these directions, their components are maximally dependent. Let us call this solution dependent component analysis (DCA). In a sense, the DCA directions describe the data as well as the ICA directions. In this simple example, the first DCA basis vector happens to be the same as in PCA (see Fig. 1), pointing to the corner which is in the direction of maximum variance. However, the second DCA basis vector differs from the second PCA basis vector, which is restricted to be orthogonal to the first one.

Depending on the choice of the functions $g(t)$ and $h(t)$, the generalized separation algorithm (20) converges either to the ICA or to the DCA solution. The basis vectors were estimated using either the nonneural formula (22) or the neural learning rule (24). Table I shows the results for some choices of $g(t)$ and $h(t)$. In all these experiments, μ_k was a small constant, and a similar stabilization as in (19) was used for the cubic nonlinearity. Note that interchanging $g(t)$ and $h(t)$ is equivalent to changing the signs of the nonlinear terms in (20). The results also show that it is possible to use the nonlinearity $g(t) = \tanh(t)$ in the EASI algorithm (19). This has the advantage that the stabilizing terms in the denominator are not needed. If the sources are super-Gaussian (the kurtosis

TABLE I
TYPE OF THE SOLUTION PROVIDED BY THE GENERALIZED EASI-TYPE ALGORITHM WITH DIFFERENT CHOICES OF THE NONLINEAR FUNCTIONS $g(t)$ AND $h(t)$ IN THE PARALLELOGRAM EXAMPLE

			$h(t)$	
		t	$\tanh(t)$	t^3
	t	—	ICA	DCA
$g(t)$	$\tanh(t)$	DCA	—	DCA
	t^3	ICA	ICA	—

is positive), (19) provides separation; if the sources are sub-Gaussian (negative kurtosis), the signs of the nonlinear terms should be changed when $g(t) = \tanh(t)$ is used in (19) for achieving separation.

Image Data: Here we present a larger scale experiment with image data. The nine source signals were the digital images shown in Fig. 8. The first three source images (S1–S3) describe natural objects, the next three (S4–S6) are Brodatz textures, and the three last ones (S7–S9) are artificially generated. More specifically, S8 is a two-dimensional sinusoidal signal and S9 uniformly distributed noise. We have not tested the mutual independence of these sources in any way. All the sources except S3 had a negative kurtosis; the kurtosis of S3 had a small positive value, so that the sum of pairwise kurtoses for any two sources was always negative. The size of the source images was 387 $\times$ 306; they were coded as vectors with 118 422 elements. Each nine-dimensional source vector $\mathbf{s}_k$ in the ICA model (3) contained the kth components of the vectorized source images. These were multiplied by a nonorthogonal full-rank 9 $\times$ 9 ICA basis matrix $\mathbf{A}$, yielding the 118 422 data vectors $\mathbf{x}_k$ used in the simulation. The nine components of the mixtures $\mathbf{x}_k$ are depicted in the subimages of Fig. 9; they look almost similar, revealing not much about the structure of original source images.

Each of the nine images in Fig. 10 contains one component of the whitened vectors $\mathbf{v}_k, k = 1, \cdots, 118\,422$. In this experiment, we used the PCA whitening matrix (13), which was computed using standard numerical software. These images already show some structure, but are still far from the original sources. For separation, we used the nonlinear PCA subspace rule (17). The data vectors were used 20 times sequentially, and the gain parameter μ_k decreased slowly from its initial value 0.0005. The learning function was $g(t) = \tanh(t)$. Fig. 11 shows the component images of the vectors $\mathbf{y}_k, k = 1, \cdots, 118\,422$. These were obtained as responses to the data vectors $\mathbf{x}_k$, $k = 1, \cdots, 118\,422$, inputted to the ICA network after learning. The component images in Fig. 11 have been rescaled so that their gray level range is the same as in the original images in Fig. 8, and in some cases their sign has been changed to opposite. The separation results are good, even though some mixing is still visible especially in the subimages corresponding to the natural scenes S1–S3 in Fig. 8.

It is obvious that some of the original source images were not truly independent. However, experience shows that it is in practice often possible to achieve adequate blind separation results even though the sources are not statistically independent. Furthermore, the images S1–S3 are not even

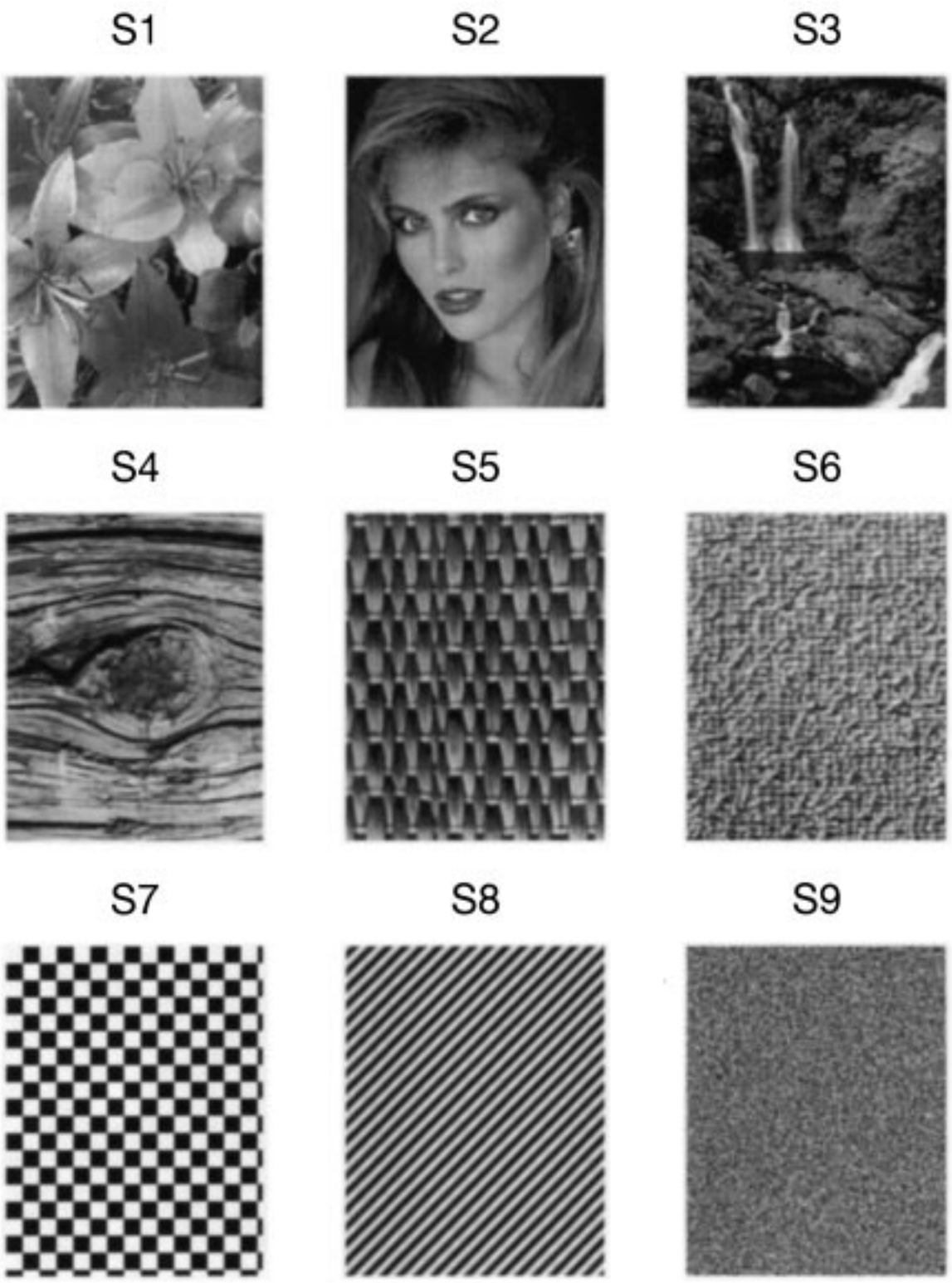

Fig. 8. The nine source images used in the image separation experiment.

stationary. The separation results could probably be improved by adding to the network of Fig. 3 another separation layer where a different nonlinearity is used for introducing more higher order statistics [11].

We also estimated the basis vectors $\mathbf{a}(i)$ of ICA using the formula (22). Fig. 12 shows the evolution of the average absolute error in the elements of the estimated mixing matrix $\hat{\mathbf{A}}$ (basis vectors) during learning. After a clear initial decrease, the error stabilizes to a roughly constant value. Furthermore, we computed the angles between the true and estimated basis vectors of ICA after learning. These angles were 0.0, 0.0, 0.8, 3.6, 4.8, 9.4, 15.3, 17.2, and 27.6°. It is apparent that the large angles correspond to the basis vectors of the natural images S1–S3 which are not truly independent and for which

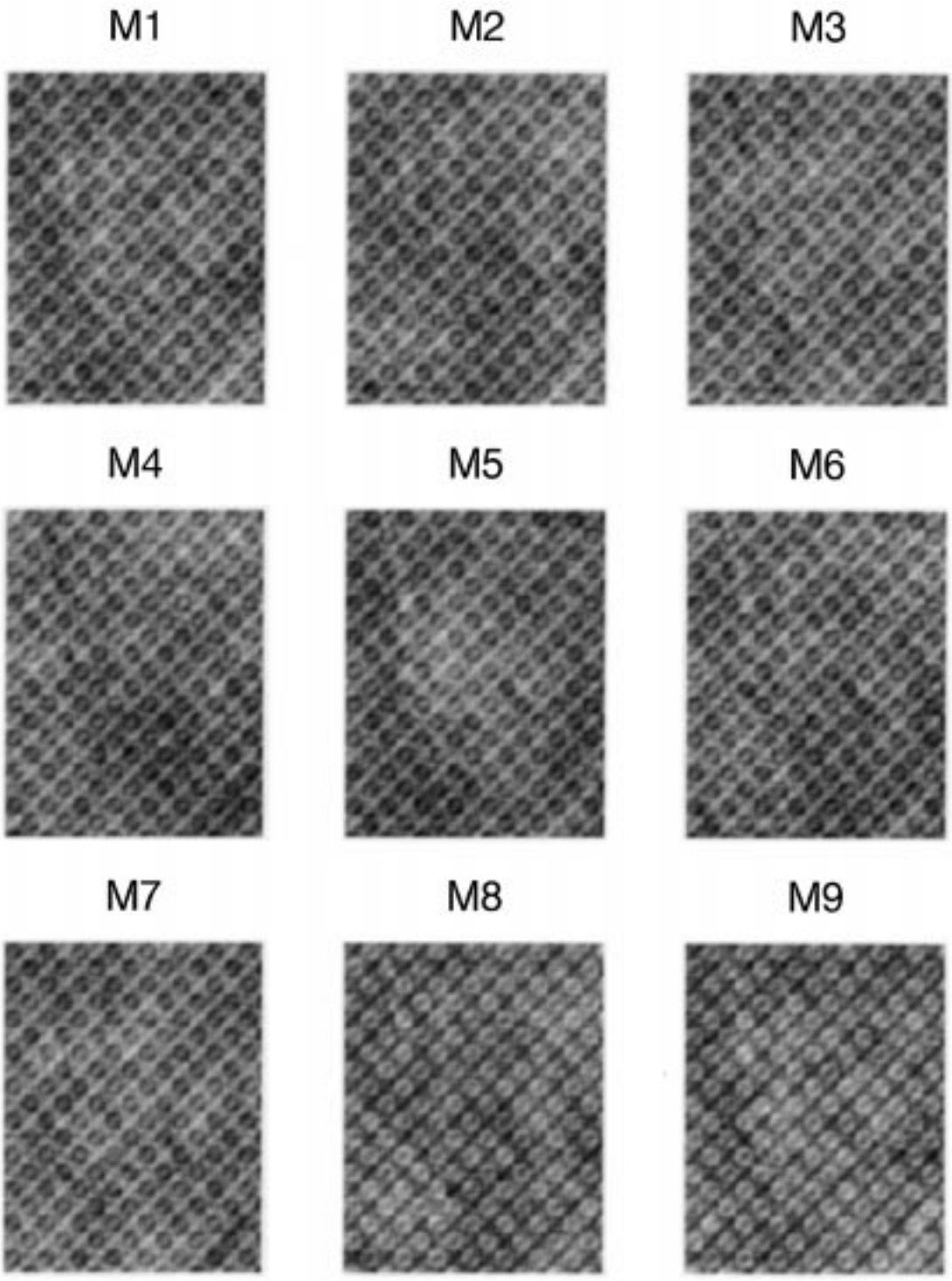

Fig. 9. The component images of the data vectors. They are linear mixtures of the source images in Fig. 8.

the separation results in Fig. 11 are not perfect. The smallest angles correspond to the artificial sources S7–S9; they are roughly independent and separated almost perfectly.

This example clearly demonstrates the usefulness of nonlinearities in PCA-type learning algorithms. The definitely poorer results in Fig. 10 show what standard PCA is typically able to achieve in this application.

Speech Data: In all the simulations described above, the sources were sub-Gaussian with a negative kurtosis. However, we have applied especially the bigradient algorithm (18) to super-Gaussian sources, too. In [53], an example similar to Comon's data is presented, where (18) successfully separates three artificially constructed super-Gaussian sources.

Furthermore, we have managed to separate up to ten real speech signals from their mixture using the bigradient algorithm. The speech signals are typically super-Gaussian [4]. In this experiment, we recorded 10 s of speech from ten different speakers. The sampling rate was 8 kHz, yielding

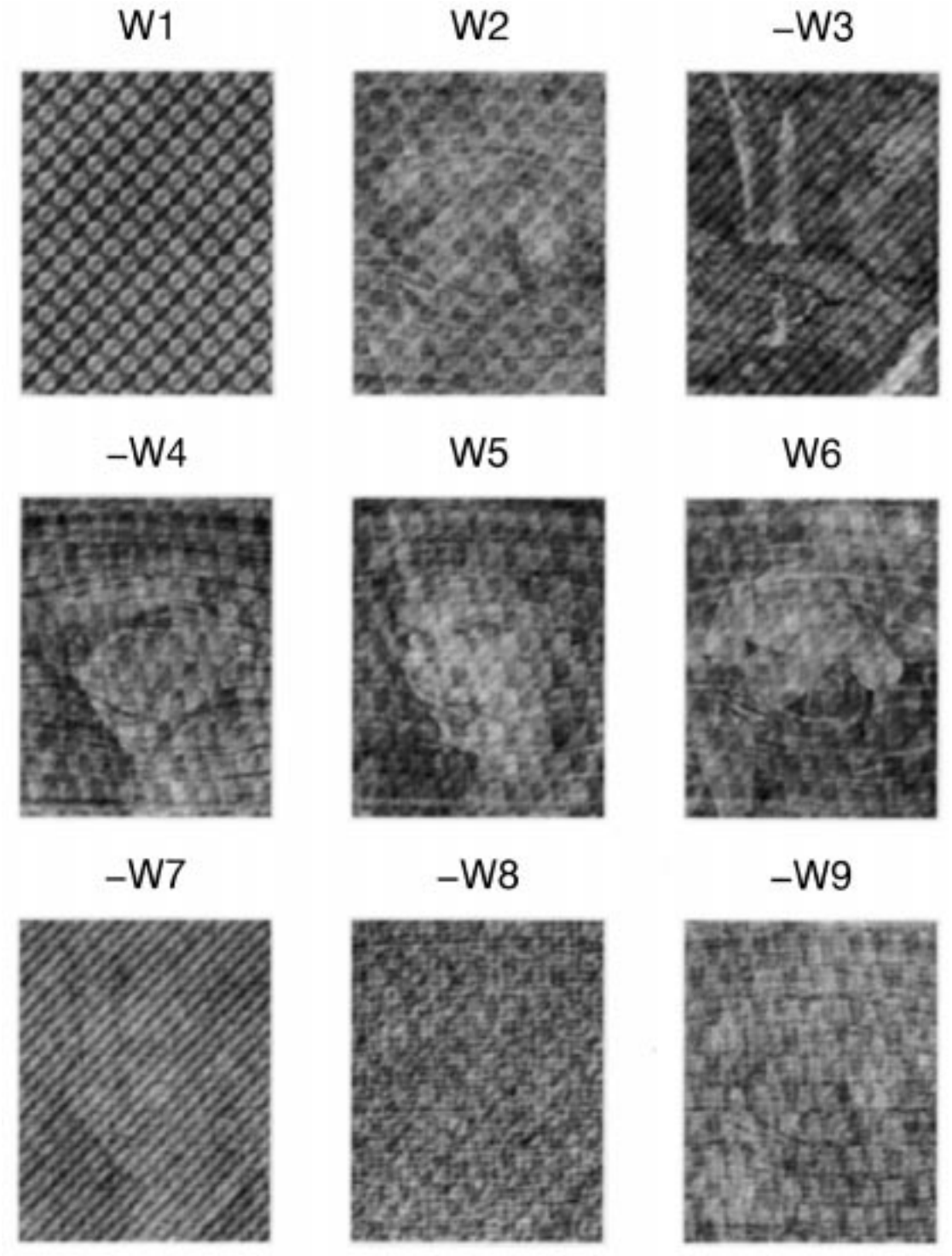

Fig. 10. The component images after PCA whitening.

80 000 samples of each speech waveform. These waveforms were preprocessed by normalizing their amplitudes. The input vectors were generated from (3) by using a 10×10 mixing matrix $\mathbf{A}$ whose elements were uniformly distributed random numbers. The noise term $\mathbf{n}_k$ was zero. Using the $\tanh(t)$ nonlinearity and the parameters $\mu = -0.01$, $\gamma = 0.9$, the bigradient algorithm (18) converged to an acceptable separation result with 10 000 training samples. This required about 5 min computing time on a SiliconGraphics INDY workstation using MATLAB code. The bigradient algorithm converges in general quickly provided that the learning parameters have been chosen appropriately.

Finally, we emphasize that preprocessing the input data by whitening them is essential for achieving good separation results using nonlinear PCA-type learning algorithms. Without whitening, the algorithms are able to separate sinusoidal signals somehow [27], but usually not other type of sources. The obvious reason is that without whitening, the algorithms

Fig. 11. The component images after separation. The separating matrix was learned using the nonlinear PCA subspace rule.

still largely respond to the second-order statistics in spite of using nonlinearities.

VII. Conclusion and Remarks

In this paper, we have introduced a class of neural networks for performing ICA. After learning, the networks have a standard feedforward structure. The basic ICA network consists of whitening, separation, and basis vector estimation layers. It can be used for both source separation and estimation of the basis vectors of ICA, which is useful for example in projection pursuit. We have presented several alternative learning proceduces for each layer, and modified our previous nonlinear PCA type learning algorithms so that their separation capabilities are greatly improved. The proposed class of networks yields good results in test examples.

Some new results continuing the work described in this paper can be found in [21]–[23], [32], [33],and [51]. In [22] and [23] computationally efficient, accurate fixed-point algo-

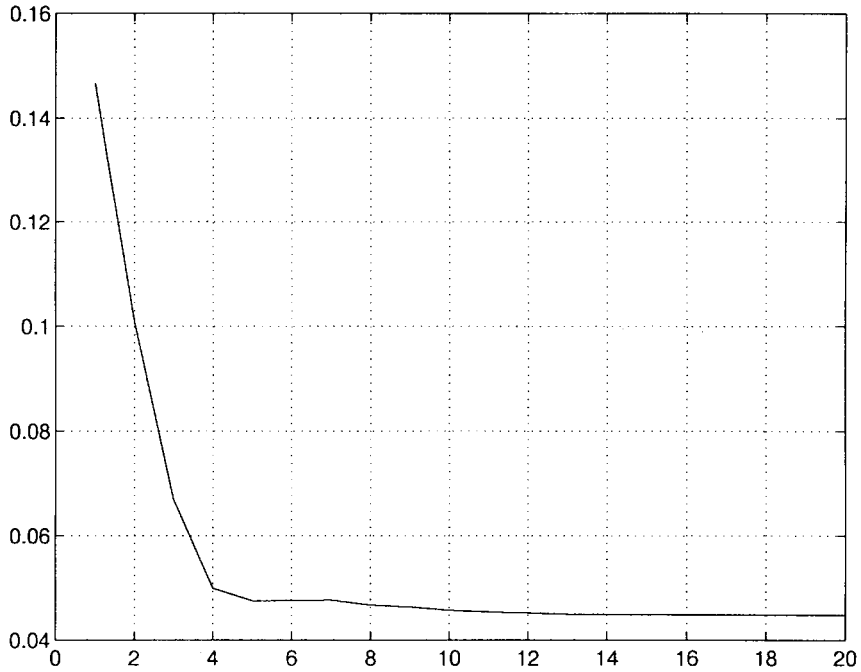

Fig. 12. The average absolute error of the elements of the estimated mixing matrix in the image data example during learning. The horizontal axis shows the number of epochs. During each epoch, the whole input data were used once in learning the separating matrix.

rithms have been introduced for estimating the independent components or source signals one at a time. These algorithms have been applied to large-scale practical problems in [21], [32], and [51]. Fast converging least-squares type adaptive or neural blind separation algorithms have been developed for a nonlinear PCA criterion in [33].

In any of the three layers of the complete ICA network, it is possible to use either a neural or a nonneural learning method. In practice, it may be advisable to learn neurally only the critical part, source separation, because efficient standard numerical methods are available for whitening and estimation of the basis vectors of ICA. On the other hand, simple truly neural learning algorithms can be used in each layer if desired.

Another remark concerns the linear ICA model (3), which is relatively simple. It would be of interest to extend the results of this paper to more general models, where the data are nonstationary, or the data model is nonlinear, or contains time delays, to mention a few possibilities. For example time delays should be included in the data model in practical speech separation. Some attempts to extend blind source separation and ICA into these directions have already been made for example in [7], [14], [36], [44], and [50]. A more detailed discussion and additional references can be found in the tutorial review [31].

ACKNOWLEDGMENT

The authors are grateful to the reviewers for their detailed and useful comments.

REFERENCES

[1] S. Amari, A. Cichocki, and H. Yang, "A new learning algorithm for blind signal separation," in *Advances in Neural Information Processing Systems 8*, D. S. Touretzky *et al*., Eds. Cambridge, MA: MIT Press, 1996, pp. 757–763.

[2] P. Baldi and K. Hornik, "Neural networks for principal component analysis: Learning from examples without local minima," *Neural Networks*, vol. 2, pp. 53–58, 1989.

[3] ——, "Learning in linear neural networks: A survey," *IEEE Trans. Neural Networks*, vol. 6, pp. 837–858, 1995.

[4] A. Bell and T. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computa.*, vol. 7, pp. 1129–1159, 1995.

[5] ——, "Learning the higher order structure of a natural sound," *Network: Computa. Neural Syst.*, vol. 7, pp. 261–267, 1996.

[6] ——, "Edges are the independent components of natural scenes," to appear in *Advances in Neural Information Processing Systems 9*. Cambridge, MA: MIT Press, 1997.

[7] G. Burel, "Blind separation of sources: A nonlinear neural algorithm," *Neural Networks*, vol. 5, pp. 937–947, 1992.

[8] G. Burel and N. Rondel, "Neural networks for array processing: From DOA estimation to blind separation of sources," in *Proc. 1993 IEEE Int. Conf. Syst., Man, Cybern.*, Le Touquet, France, Oct. 1993, pp. 601–606.

[9] J.-F. Cardoso and B. Laheld, "Equivariant adaptive source separation," *IEEE Trans. Signal Processing*, vol. 44, pp. 3017–3030, Dec. 1996.

[10] A. Cichocki and R. Unbehauen, *Neural Networks for Optimization and Signal Processing*. New York: Wiley, 1993.

[11] A. Cichocki, W. Kasprzak, and S. Amari, "Multilayer neural networks with a local adaptive learning rule for blind separation of source signals," in *Proc. 1995 Int. Symp. Nonlinear Theory Applicat., NOLTA-95*, Las Vegas, NV, vol. 1, Dec. 1995, pp. 61–66.

[12] P. Comon, "Separation of stochastic processes," in *Proc. Wkshp. Higher Order Spectral Analysis*, Vail, CO, June 1989, pp. 174–179.

[13] ——, "Independent component analysis—A new concept?," *Signal Processing*, vol. 36, pp. 287–314, 1994.

[14] G. Deco and D. Obradovic, *An Information-Theoretic Approach to Neural Computing*. New York: Springer-Verlag, 1996.

[15] K. Diamantaras and S. Kung, *Principal Component Networks—Theory and Applications*. New York: Wiley, 1996.

[16] J. Friedman, "Exploratory projection pursuit," *J. Amer. Statist. Assoc.*, vol. 82, no. 397, pp. 249–266, Mar. 1987.

[17] C. Fyfe and R. Baddeley, "Nonlinear data structure extraction using simple Hebbian networks," *Biol. Cybern.*, vol. 72, pp. 533–541, 1995.

[18] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New York: IEEE Computer Soc. Press and Macmillan, 1994.

[19] J. Herault, C. Jutten, and B. Ans, "Détection de grandeur primitives dans un message composite par une architecture de calcul neuromimétique en apprentissage nonsupervisé, " in *Proc. GRETSI Conference*, Nice, France, May 1985, pp. 1017–1022, in French.

[20] P. Huber, "Projection pursuit," *Ann. Statist.*, vol. 13, no. 2, pp. 435–475, 1985.

[21] J. Hurri, A. Hyvärinen, J. Karhunen, and E. Oja, "Image feature extraction using independent component analysis," in *Proc. 1996 IEEE Nordic Signal Processing Symp.*, Espoo, Finland, Sept. 1996, pp. 475–478.
[22] A. Hyvärinen and E. Oja, "One-unit learning rules for independent component analysis," to appear in *Advances in Neural Information Processing Systems 9*. Cambridge, MA: MIT Press, 1997.
[23] A. Hyvärinen, "A family of fixed-point algorithms for independent component analysis," to appear in *Proc. 1997 IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'97*, Munich, Germany, Apr. 1997.
[24] C. Jutten, "Calcul neuromimétique et traitement du signal, analyze en composantes indépendantes," Thése d'Etat, INPG, Univ. Grenoble, France, 1987 (in French).
[25] C. Jutten and J. Herault, "Independent components analysis (INCA) versus principal components analysis," in *Signal Processing IV: Theories and Applications, Proc. EUSIPCO-88*, J. Lacoume *et al.*, Eds. Amsterdam, The Netherlands: Elsevier, 1988, pp. 643–646.
[26] ———, "Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, July 1991.
[27] J. Karhunen and J. Joutsensalo, "Representation and separation of signals using nonlinear PCA type learning," *Neural Networks*, vol. 7, no. 1, pp. 113–127, 1994.
[28] J. Karhunen, "Optimization criteria and nonlinear PCA neural networks," in *Proc. 1994 IEEE Int. Conf. Neural Networks*, Orlando, FL, June 1994, pp. 1241–1246.
[29] ———, "Stability of Oja's PCA subspace rule," *Neural Computa.*, vol. 6, pp. 739–747, 1994.
[30] 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.
[31] J. Karhunen, "Neural approaches to independent component analysis and source separation," in *Proc. 4th European Symp. Artificial Neural Networks, ESANN'96*, Bruges, Belgium, Apr. 1996, pp. 249–266.
[32] J. Karhunen, A. Hyvärinen, R. Vigário, J. Hurri, and E. Oja, "Applications of neural blind separation to signal and image processing," to appear in *Proc. 1997 IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'97*, Munich, Germany, Apr. 1997.
[33] J. Karhunen and P. Pajunen, "Blind source separation using least-squares type adaptive algorithms," to appear in *Proc. 1997 IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'97*, Munich, Germany, Apr. 1997.
[34] P. Kotilainen, J. Saarinen, and K. Kaski, "Hardware implementations of PCA neural networks," in *Artificial Neural Networks, in Proc. ICANN-92*, I. Aleksander and J. Taylor, Eds. Amsterdam, The Netherlands: North-Holland, vol. 2, 1992, pp. 1427–1430.
[35] B. Laheld and J.-F. Cardoso, "Adaptive source separation with uniform performance," in *Signal Processing VII: Theories and Applications Proc. EUSIPCO-94*, M. Holt *et al.*, Eds. Lausanne, Switzerland: EURASIP, vol. 2, 1994, pp. 183–186.
[36] K. Matsuoka, M. Ohya, and M. Kawamoto, "A neural net for blind separation of nonstationary signals," *Neural Networks*, vol. 8, no. 3, pp. 411–419, 1995.
[37] E. Moreau and O. Macchi, "New self-adaptive algorithms for source separation based on contrast functions," in *Proc. IEEE Signal Proc. Wkshp. Higher Order Statistics*, Lake Tahoe, NV, June 1993, pp. 215–219.
[38] E. Oja, "A simplified neuron model as a principal components analyzer," *J. Math. Biol.*, vol. 15, pp. 267–273, 1982.
[39] ———, *Subspace Methods of Pattern Recognition*. Letchworth, U.K.: Res. Studies Press, 1983.
[40] E. Oja and J. Karhunen, "On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix," *J. Math. Anal. Applicat.*, vol. 106, pp. 69–84, 1985.
[41] E. Oja, H. Ogawa, and J. Wangviwattana, "Learning in nonlinear constrained Hebbian networks," in *Artificial Neural Networks Proc. ICANN-91*, T. Kohonen *et al.*, Eds. Amsterdam, The Netherlands: North-Holland, 1991, pp. 385–390.
[42] E. Oja, "Principal components, minor components, and linear neural networks," *Neural Networks*, vol. 5, pp. 927–935, 1992.
[43] ———, "The nonlinear PCA learning rule and signal separation—Mathematical analysis," Lab. Computer Inform. Sci., Helsinki Univ. Technology, Espoo, Finland, Tech. Rep. A26, Aug. 1995.
[44] J. Platt and F. Faggin, "Networks for the separation of sources that are superimposed and delayed," in *Advances in Neural Information Processing Systems 4*, J. Moody *et al.*, Eds. San Mateo, CA: Morgan Kaufmann, 1991, pp. 730–737.
[45] M. Plumbley, "A Hebbian/anti-Hebbian network which optimizes information capacity by orthonormalizing the principal subspace," in *Proc. IEE Conf. Artificial Neural Networks*, Brighton, U.K., May 1993, pp. 86–90.
[46] ———, "Efficient information transfer and anti-Hebbian neural networks," *Neural Networks*, vol. 6, pp. 823–833, 1993.
[47] F. Silva and L. Almeida, "A distributed solution for data orthonormalization," in *Artificial Neural Networks Proc. ICANN-91*, T. Kohonen *et al.*, Eds. Amsterdam, The Netherlands: North-Holland, 1991, pp. 943–948.
[48] J. Taylor and M. Plumbley, "Information theory and neural networks," in *Mathematical Approaches to Neural Networks*, J. Taylor, Ed. Amsterdam, The Netherlands: Elsevier, 1993, pp. 307–340.
[49] C. Therrien, *Discrete Random Signals and Statistical Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1992.
[50] K. Torkkola, "Blind separation of convolved sources based on information maximization," in *Proc. IEEE Wkshp. Neural Networks Signal Processing*, Kyoto, Japan, Sept. 1996, pp. 423–432.
[51] R. Vigário, A. Hyvärinen, and E. Oja, "ICA fixed-point algorithm in extraction of artifacts from EEG," in *Proc. 1996 IEEE Nordic Signal Processing Symp.*, Espoo, Finland, Sept. 1996, pp. 383–386.
[52] L. Wang and J. Karhunen, "A unified neural bigradient algorithm for robust PCA and MCA," *Int. J. Neural Syst.*, vol. 7, pp. 53–67, 1996.
[53] L. Wang, J. Karhunen, and E. Oja, "A bigradient optimization approach for robust PCA, MCA, and source separation," in *Proc. 1995 IEEE Int. Conf. Neural Networks*, Perth, Australia, Nov. 1995, pp. 1684–1689.

Juha Karhunen (M'90) received the Doctor of Technology degree from the Department of Technical Physics of Helsinki University of Technology, Espoo, Finland, in 1984. In 1994, he became Docent of statistical signal analysis.

Since 1976, he has been with the Laboratory of Computer and Information Science at Helsinki University of Technology, Finland, where he is currently Senior Teacher and Researcher (Assistant Professor). His current research interests include neural networks, unsupervised learning, and their applications to signal processing. He has published a number of conference and journal papers on these topics and has given invited talks in several international conferences.

Dr. Karhunen is a Member of INNS and the Pattern Recognition Society of Finland.

Erkki Oja (S'76–M'77–SM'90) received the Dr.Tech. degree (with distinction) from Helsinki University of Technology, Espoo, Finland, in 1977.

He is Professor of Computer Science at the Laboratory of Computer and Information Science, Helsinki University of Technology. He is the author of a number of journal papers and book chapters on pattern recognition, computer vision, and neural computing, and the book *Subspace Methods of Pattern Recognition*, which has been translated into Chinese and Japanese. His present research interests are in the study of subspace, PCA, self-organizing networks, and applying artificial neural networks to computer vision and signal processing.

Dr. Oja has served in the scientific and organization committees of a number of conferences, recently including ICANN-96 and ICNN-96. He is a Member of the Finnish Academy of Sciences, Past Chairman of the Finnish Pattern Recognition Society, Member of the Governing Board of the International Association of Pattern Recognition (IAPR), IAPR Fellow, Vice-President of the European Neural Network Society (ENNS), and member of the Neural Networks Technical Committee of the IEEE. He is Member of the editorial boards of the journals *Neural Networks*, *Neural Computation*, *Pattern Recognition Letters*, and the IEEE TRANSACTIONS ON NEURAL NETWORKS.

Liuyue Wang was born in Jiangxi, China, in February 1965. He received the M.Sc. degree in electrical engineering from Xi'an Jiaotong University, China, in 1990, and the Doctor of Technology degree from Helsinki University of Technology, Espoo, Finland, in 1996.

From 1993 to 1994 he was a Researcher in the Department of Information Technology at Lappeenranta University of Technology, Finland. Since 1994 he has been in various positions with the Laboratory of Computer and Information Science at Helsinki University of Technology, where he is now a Research Scientist. His current research interests include signal and image processing with neural networks and pattern recognition.

Ricardo Vigário received the B.Sc and M.Sc. degrees in applied and medical physics and biomedical engineering from the University of Lisbon, Portugal, in 1992 and 1994, respectively. Since 1993, he has been with the Laboratory of Computer and Information Science at Helsinki University of Technology, Espoo, Finland, where he is currently working toward the Dr.Tech. degree.

His current research interests include neural networks in signal processing, computer vision, and biomedical signal processing.

Jyrki Joutsensalo was born in Kiukainen, Finland, in July 1966. He received the Diploma Engineer, Licentiate of Technology, and Doctor of Technology degrees from the Helsinki University of Technology, Espoo, Finland, in 1992, 1993, and 1994, respectively.

Since 1988 he has been in various positions with the Laboratory of Computer and Information Science at the Helsinki University of Technology, where he is now a Research Scientist. His current research interest include statistical signal processing and neural networks, especially subspace approaches with application to spectral estimation, array signal processing, and mobile radio communications.