Robust Neural Networks with On-Line Learning for Blind Identification and Blind Separation of Sources

Andrzej Cichocki and Rolf Unbehauen, *Fellow, IEEE*

***Abstract*— Two unsupervised, self-normalizing, adaptive learning algorithms are developed for robust blind identification and/or blind separation of independent source signals from a linear mixture of them. One of these algorithms is developed for on-line learning of a single-layer feed-forward neural network model and a second one for a feedback (fully recurrent) neural network model. The proposed algorithms are robust, efficient, fast and suitable for real-time implementations. Moreover, they ensure the separation of extremely weak or badly scaled stationary signals, as well as a successful separation even if the mixture matrix is very ill-conditioned (near singular). The performance of the proposed algorithms is illustrated by computer simulation experiments.**

I. Introduction

BLIND separation of sources called also "waveform-preserved blind estimation of multiple independent sources" is an emerging field of fundamental research with many potential applications [1]–[31]. Generally speaking, the problem of blind separation of sources can be formulated as the problem of separating or estimating waveforms of primary sources (original input signals) from an array of sensors, without knowing the characteristics of the transmission channels [1]–[3], [9], [20]. In a large number of applications, the signals received by an array of sensors (e.g., sensors, antennas, transducers etc.) are a mixture of original source signals. These source signals are usually totally unknown as in the case of processing and enhancement of acoustic signals and array processing for radar or sonar signals [2], [9]–[15], [19], [20]. Also, in the biomedical domain the signals provided by sensors are often a mixture of many independent sources and it is necessary to extract the original source signals from their mixture [2], [9], [19].

The neural network models with learning capabilities for on-line blind separation of sources from linear mixture signals have been first developed by Herault and Jutten [1]–[3]. An extension of the adaptive learning algorithms for more complex cases, e.g., for convolutive mixing of sources with causal FIR filters [10]–[15], [19], [20] and nonlinear mixing [18] has been proposed more recently.

Manuscript received June 27, 1994; revised November 10, 1995. This paper was recommended by Associate Editor C. Jutten.

A. Cichocki is with the Institute of Physical and Chemical Research, FRP RIKEN Laboratory Artificial Brain Systems 2-1, Hirosawa, Wako, Saitama 351-01, Japan, on leave from the Warsaw University of Technology, Poland.

R. Unbehauen is with the University Erlangen-Nürnberg, D-91058 Erlangen, Germany.

Publisher Item Identifier S 1057-7122(96)07598-8.

The main objective of this paper is twofold.

1) To propose unsupervised, robust and efficient learning algorithms which are also suitable for badly scaled and ill-conditioned problems.
2) To show that the learning algorithms developed for a feedforward architecture can be converted into equivalent algorithms for a corresponding feedback neural network architecture.

II. Formulation of the Problem

The mixing process of the unknown input sources $s_j(t)$ $(j = 1, 2, \cdots, n)$ can have different mathematical or physical models dependent on the specific application or situation and different a priori information about the signals that is available. In this paper we will focus on the simplest but general case where no *a priori* information on the sources themselves is available and the mixed sensor signals $x_i(t)$ are a linear combination of the unknown statistically independent source signals $s_j(t)$, i.e.,

$$x_i(t) = \sum_{j=1}^{n} a_{ij} s_j(t) \tag{1a}$$

or

$$\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{s}(t) \tag{1b}$$

where $\boldsymbol{x}(t) = [x_1(t), x_2(t), \cdots, x_n(t)]^T$ are the observed sensor output signals, $\boldsymbol{s}(t) = [s_1(t), s_2(t), \cdots, s_n(t)]^T$ are the unknown mutually independent zero-mean source signals, and $\boldsymbol{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$ is a mixture matrix of the unknown real mixing parameters [2] (see Fig. 1). It is assumed that the parameters a_{ij} are fixed or slowly varying in the time. Moreover, it is assumed that the mixing matrix $\boldsymbol{A}$ is nonsingular, i.e., $\det(\boldsymbol{A}) \neq 0$ (although, as will be seen later it can be nearly singular, i.e., very ill-conditioned). We also assume for simplicity that the number of sensors is equal to the number of unknown sources.

The problem is formulated as follows: it is necessary to develop a neural network with a suitable learning algorithm which makes possible an on-line (i.e., in real-time) generation of output signals, say

$$\boldsymbol{y}(t) = [y_1(t), y_2(t), \cdots, y_n(t)]^T$$

which are estimates of the source signals

$$\boldsymbol{s}(t) = [s_1(t), s_2(t), \cdots, s_n(t)]^T.$$

For such a kind of problem there is inherited an indeterminacy [2], [8], [9], [22]. This indeterminacy is characterized by the

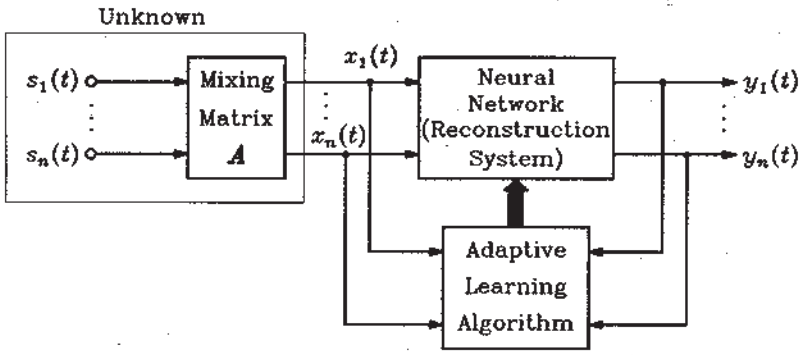


Fig. 1. A schematic diagram of blind separation of source signals.

magnitude scaling and the order in which the estimated signals $y_i(t)$ are arranged with respect to the original source signals $s_j(t)$. This indeterminacy can be expressed mathematically by the matrix equations [9], [22]

$$\boldsymbol{y}(t) = \boldsymbol{DPs}(t) = \tilde{\boldsymbol{P}}\boldsymbol{s}(t) \tag{2}$$

where $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ is a diagonal scaling matrix with nonzero entries and $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ is any permutation matrix and $\tilde{\boldsymbol{P}}$ is the generalized permutation matrix with exactly one nonzero element in each row and each column. This indeterminacy seems to be a rather severe limitation, but in a great number of applications this limitation is not essential since the most relevant information about the source signals is contained in the waveforms of the signals (for instance processing of multisensor biomedical recordings) rather than in their magnitudes and orders in which they are arranged.

A closely related problem to the blind separation of sources is the problem of blind identification whose objection is to estimate a mixture parameter matrix $\boldsymbol{A}$ from the sensor signals $\boldsymbol{x}(t)$ without knowing the source signals $\boldsymbol{s}(t)$ [8]. The neural networks considered in this paper allow to identify both the matrix $\boldsymbol{A}$ and the sources $\boldsymbol{s}(t)$ from the sensor signals $\boldsymbol{x}(t)$.

III. Herault-Jutten Neural Network Model

The problem stated in Section II was first formulated and completely solved by Herault and Jutten using the neural network approach [1]–[3]. In order to better understand our proposals and modifications we will review here very briefly the basic solution (algorithm) proposed by Herault and Jutten. In order to solve the problem they proposed a linear recursive neural network described by the set of equations

$$y_i(t) = x_i(t) - \sum_{\substack{j=1\\ j\neq i}}^{n} w_{ij}(t)y_j(t), \quad (i = 1, 2, \cdots, n) \tag{3a}$$

which can be written compactly in the matrix form [cf. Fig. 2(a) and (b)]

$$\boldsymbol{y}(t) = \boldsymbol{x}(t) - \boldsymbol{W}(t)\boldsymbol{y}(t) \tag{3b}$$

with the equilibrium value

$$\boldsymbol{y}(t) = [\boldsymbol{I} + \boldsymbol{W}(t)]^{-1}\boldsymbol{x}(t) \tag{4}$$

where $\boldsymbol{W}(t) = [w_{ij}(t)] \in \mathbb{R}^{n\times n}$ with $w_{ii}(t) \equiv 0\ \forall i$ is the matrix of the adaptive synaptic weights, $\boldsymbol{y}(t) \in \mathbb{R}^n$ is the output vector of the neural network and $\boldsymbol{I}$ means the $n \times n$ identity matrix. For such a neural network Herault and Jutten developed an original and ingenious unsupervised learning algorithm, which can be written in the form of a set of nonlinear differential equations

$$\frac{dw_{ij}(t)}{dt} = \mu(t)f[y_i(t)]g[y_j(t)], \quad \text{for} \quad i \neq j \tag{5}$$

where $\mu(t) > 0$ is the learning rate and $f(y), g(y)$ are nonlinear different odd activation functions, typically, $f(y) = y^3$ or $f(y) = y^2 \operatorname{sign}(y)$ and $g(y) = y, g(y) = \operatorname{sign}(y)$ or $g(y) = \alpha \tanh(\beta y)$ with $\alpha > 0, \beta > 0$. In fact, the Herault-Jutten algorithm [(5a) and (5b)] (henceforth called H-J algorithm) has appeared robust and a large variety of odd functions can be used to perform successfully the separation of independent sources. A functional block diagram illustrating the implementation of the H-J algorithm is shown in Fig. 2(b). The H-J algorithm has been successfully implemented by using electronic hardware as well as investigated by simulating it on digital computers [4], [5]. Although the H-J algorithm seems to be very robust with respect to the shape of the activation functions and useful in many applications several disadvantages occur in some cases. First of all, the H-J algorithm works rather poorly or even fails to separate sources if the signals are badly scaled (i.e., some of the source signals are very weak in comparison to others) and/or if the mixing matrix $\boldsymbol{A}$ is ill-conditioned (nearly singular). Moreover, the neural network [see Fig. 2(b)] does not ensure global and temporal stability. In fact, the stability of the neural network depends on many factors such as the initial conditions, the mixing coefficients, the kind of activation functions. Furthermore, in a purely software implementation of the algorithm on a digital computer the computation of the inverse matrix $\hat{\boldsymbol{W}}(k) \triangleq [\boldsymbol{I} + \boldsymbol{W}(\boldsymbol{k})]^{-1}$ is required at every iteration step k which is a rather computationally involved procedure. In analog hardware realization this procedure is inherently (automatically) performed by an appropriate network [4], [5].

In this paper we will propose two unsupervised, self-adaptive learning algorithms which partially alleviate or even completely eliminate the above mentioned drawbacks. It should be noted that the stability of (3a), (3b), and (4) is guaranteed if all the eigenvalues of $\boldsymbol{W}(t)$ remain inside the unit circle for each t [26]. To ensure stability we might require that the sum of the absolute values of each row of $\boldsymbol{W}(t)$ to be

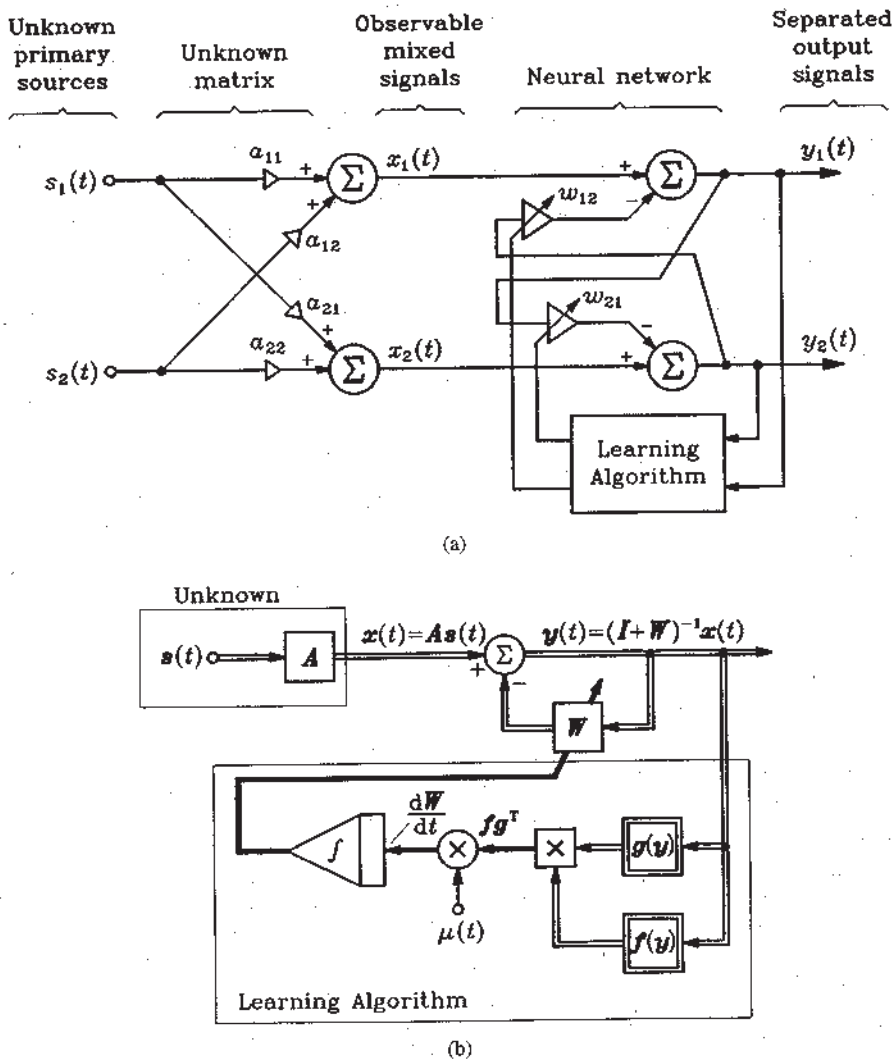


Fig. 2. (a) Illustration of the blind separation of sources in the case of two unknown primary sources ($n = 2$) using the Herault-Jutten approach. (b) Functional block diagram illustrating the Herault-Jutten learning algorithm.

less than one [13]. However, this is usually a too excessively strong sufficient condition. In general, the problem of stability requires a special attention although, as extensive experimental investigations have shown, the learning H-J algorithm is robust in the sense that it usually forces the network to get a stable behavior. This can be explained by the fact that the H-J algorithm or strictly speaking a subset of the Herault Jutten learning rules fulfills approximately the principle of minimum output power [2], [13]. A physical system which minimizes the output power tends to pull the poles inside the unit circle toward the origin. In order to derive the H-J algorithm let us minimize the output power of the neural network defined as energy (cost) function

$$E_c(\boldsymbol{W}) = E\left\{\sum_{i=1}^{n} \rho_i(y_i)\right\} \tag{6}$$

where $E\{\cdot\}$ means the expected value of its argument and the "power" functions of the output signals are defined as

$$\rho_i(y_i) = \begin{cases} \frac{1}{p}|y_i|^p & \text{for } p \text{ odd} \\ \frac{1}{p}y_i^p & \text{for } p \text{ even} \end{cases} \tag{7}$$

$p = 3, 4, \cdots$ typically $p = 3$ or $p = 4$ and the output signals are described by (3a). The minimization of the introduced energy function E_c according to the gradient descent rule leads to the relation

$$\frac{dw_{ij}(t)}{dt} = -\mu(t)\frac{\partial E_c(\boldsymbol{W})}{\partial w_{ij}} = -\mu(t)\frac{\partial E_c}{\partial y_i}\frac{\partial y_i}{\partial w_{ij}} \cong \mu(t)E\{f[y_i(t)]y_j(t)\} \tag{8a}$$

which can be written compactly in matrix form as

$$\frac{d\boldsymbol{W}(t)}{dt} \doteq \mu(t)E\{\boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{y}^T(t)\} \tag{8b}$$

where the symbol $\doteq$ indicates that nondiagonal elements only on its left side are equivalent to those on the right side, $\mu(t) > 0$ is the learning rate,

$$\boldsymbol{f}(\boldsymbol{y}) = [f_1[y_1(t)], f_2[y_2(t)], \cdots, f_n[y_n(t)]]^T$$

and

$$f_i(y_i) \triangleq \frac{\partial \rho_i(y_i)}{\partial y_i} = \begin{cases} y_i^{p-1}\,\text{sign}\,(y_i) & \text{for } p \text{ odd} \\ y_i^{p-1} & \text{for } p \text{ even.} \end{cases}$$

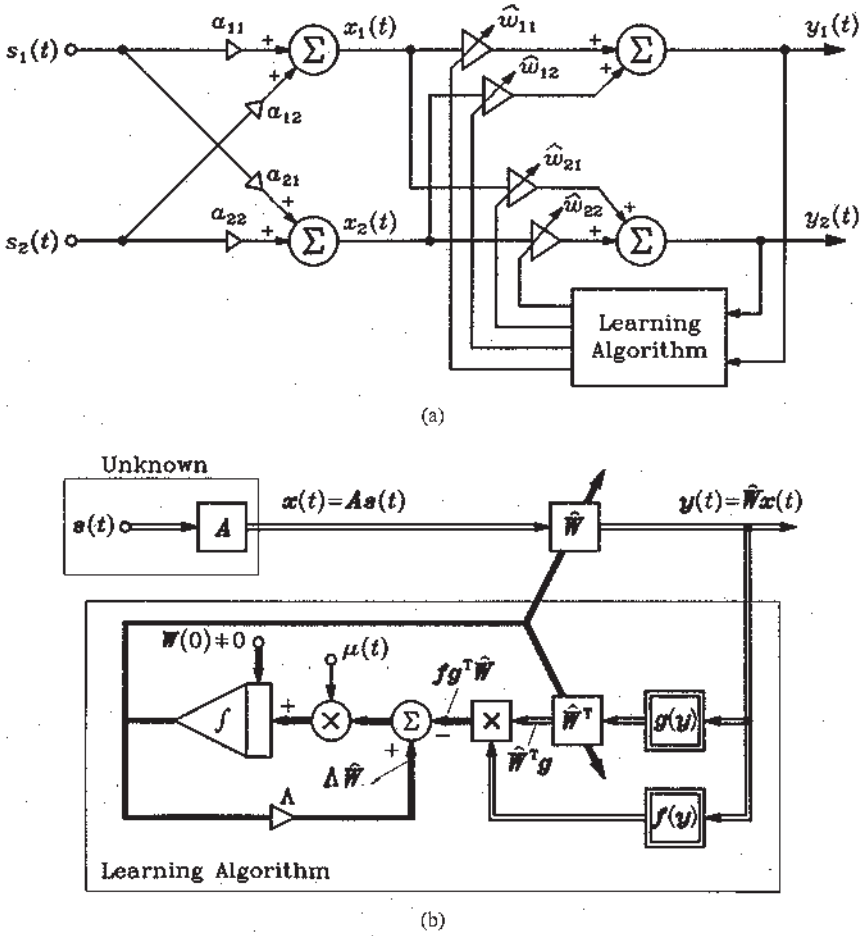


Fig. 3. Mixing and separation of sources using a feedforward neural network. (a) Detailed model for $n = 2$,. (b) Functional block diagram illustrating the implementation of the adaptive learning algorithm (15), (16).

Unfortunately, the expected values of $E\{\boldsymbol{f}(\boldsymbol{y})\boldsymbol{y}^T\}$ are not available and they can be approximated off-line by [3]

$$E\{\boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{y}^T(t)\} \cong \frac{1}{T}\sum_{t=1}^{T}[\boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{y}^T(t) \tag{9}$$

with

$$\boldsymbol{y}(t) = [\boldsymbol{I} + \boldsymbol{W}(t)]^{-1}\boldsymbol{x}(t)$$

for the discrete time instants $t = 1, 2, \cdots$. However, in practice the optimal solution can be reached on-line by applying a stochastic gradient descent approach assuming that the true gradient is replaced by a crude instantaneous gradient estimate, namely

$$E\{\boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{y}^T(t)\} \cong \boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{y}^T(t). \tag{10}$$

Hence

$$\frac{d\boldsymbol{W}(t)}{dt} \doteq \mu(t)\boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{y}^T(t). \tag{11}$$

Note that the above learning algorithm can be considered as a generalized anti-Hebbian rule [19], [22]. In fact, the algorithm (11) constitutes a subset of the H-J rules with $g(y) = y$ and $f(y)$ that equals for instance $y^2 \operatorname{sign} y$ for $p = 3$.

We have discussed briefly the Herault-Jutten learning algorithm because this algorithm has been for us the main inspiration and motivation for its further improvement and extension. In fact, the learning algorithms discussed in the next sections can be considered as modifications and/or slight extensions of the Herault-Jutten algorithm. However, these extensions provide dramatic improvements in performance and reliability.

IV. Robust Self-Normalizing Adaptive Learning Algorithms

4.1 Feedforward Neural Network

Let us consider a single layer feed-forward neural network consisting of n linear neurons (processing units) described by

$$y_i(t) = \sum_{j=1}^{n}\hat{w}_{ij}(t)x_j(t) \qquad (i = 1, 2, \cdots, n) \tag{12a}$$

or in matrix form by [cf. Fig. 3(a) and (b)]

$$\boldsymbol{y}(t) = \hat{\boldsymbol{W}}(t)\boldsymbol{x}(t) \tag{12b}$$

where $\hat{\boldsymbol{W}}(t) = [\hat{w}_{ij}(t)] \in \mathbb{R}^{n\times n}$ is the matrix of the adaptively adjusted synaptic weights, $\boldsymbol{x}(t) = [x_1(t), x_2(t), \cdots, x_n(t)]^T$ is the vector of the observed sensor signals and $\boldsymbol{y}(t) = [y_1(t), y_2(t), \cdots, y_n(t)]^T$ means the vector of the output signals which after the learning phase (i.e., after the synaptic

weights $w_{ij}(t)$ achieve the desired equilibrium point) must be proportional to the primary sources $s_j(t)$ $(j = 1, 2, \cdots, n)$. To achieve this the following relationship must be satisfied at the equilibrium point [cf. (2)]

$$\boldsymbol{y}(t) = \hat{\boldsymbol{W}}\boldsymbol{x}(t) = \hat{\boldsymbol{W}}\boldsymbol{A}\boldsymbol{s}(t) = \boldsymbol{D}\boldsymbol{P}\boldsymbol{s}(t). \tag{13}$$

Hence

$$\hat{\boldsymbol{W}} = \boldsymbol{D}\boldsymbol{P}\boldsymbol{A}^{-1}. \tag{14}$$

For the model described above we have developed a novel adaptive, on-line learning algorithm

$$\frac{d\hat{\boldsymbol{W}}(t)}{dt} = \mu(t)\{\boldsymbol{\Lambda} - \boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{g}^T[\boldsymbol{y}(t)]\}\hat{\boldsymbol{W}}(t)$$
$$\text{with} \quad \hat{\boldsymbol{W}}(0) \neq 0 \quad \text{and} \quad \det \hat{\boldsymbol{W}}(0) \neq 0$$
$$(\text{typically } \hat{\boldsymbol{W}}(0) = \boldsymbol{I}) \tag{15}$$

where $\mu(t) > 0$ is the learning rate, $\boldsymbol{\Lambda} = \text{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ is a diagonal matrix with the amplitude scaling factors $\lambda_i > 0$ $\forall i$ (typically $\boldsymbol{\Lambda} = \boldsymbol{I}$, i.e., $\lambda_i = 1, \forall i$).

For clarity of presentation a development and justification of the proposed learning algorithm is given in the Appendix.

A functional block diagram of the neural network with the on-chip implementable learning algorithm is depicted in Fig. 3(b). The learning algorithm can be written in scalar form as

$$\frac{d\hat{w}_{ij}(t)}{dt} = \mu(t)\left[\lambda_i \hat{w}_{ij}(t) - f_i[y_i(t)]\sum_{p=1}^{n} \hat{w}_{pj}(t) g_p[y_p(t)]\right]. \tag{16}$$

The learning rate $\mu(t) > 0$ can be fixed in the first learning phase (in the so-called "search" phase of learning) and then in the second (so-called "converge") phase of learning it can be exponentially decreased to zero. The proposed learning algorithm is slightly more complex than the H-J algorithm but it is more powerful and efficient in performance. This has been confirmed by extensive computer simulations. The learning algorithms (15), (16) can be easily realized by a VLSI analog circuit. It can also be adopted for a digital (time-discrete) realization or even implemented very easily on a standard digital computer. In this case the continuous-time algorithm (15) (system of differential equations) can be directly converted to iterative time-discrete learning rules (system of difference equations). Using for instance the Euler rule to (15) we obtain

$$\hat{\boldsymbol{W}}(k+1) = \hat{\boldsymbol{W}}(k) + \eta(k)[\boldsymbol{\Lambda} - \boldsymbol{f}[\boldsymbol{y}(k)]\boldsymbol{g}^T[\boldsymbol{y}(k)]]\hat{\boldsymbol{W}}(k)$$
$$k = (0, 1, 2, \cdots) \quad \text{with} \quad \hat{\boldsymbol{W}}(0) \neq 0$$
$$\text{and} \quad \det \hat{\boldsymbol{W}}(0) \neq 0 \tag{17}$$

where $\eta(k) = \mu(t)T > 0$ is the learning step and T is the sampling step. However, it should be noted that for the iterative learning rule (17), in contrast to the continuous-time algorithm, the learning rates $\eta(k) > 0$ must be upper-bounded to small values in order to ensure numerical stability of the algorithm [22]. It is also interesting to note that the learning algorithms (15) and (17) achieve an equilibrium point if the following conditions are satisfied

$$E\{f_i[y_i(t)]g_j[y_j(t)]\} = 0 \quad \text{for} \quad i \neq j \tag{18a}$$

and

$$\lambda_i = E\{f_i[y_i(t)]g_i[y_i(t)]\} \quad \text{for} \quad i = j. \tag{18b}$$

The new learning algorithm (15)–(17) is somewhat similar to the H-J algorithm. In fact it uses the same or similar odd activation functions $f(y), g(y)$. However, our development and proposal differ from the neural network models and associated learning algorithms (known from literature) in several respects. First, in contrast to the basic Herault-Jutten model (with a feedback architecture) our neural network has a feed-forward architecture (i.e., $\boldsymbol{y} = \hat{\boldsymbol{W}}\boldsymbol{x}$), and as shown by extensive computer simulation it is temporally stable independent of the initial conditions. Furthermore, our neural network does not require the computation of inverse matrices at every iteration step[1] [cf. (4)]. Second, our learning algorithm ensures a self-normalization of the amplitude (or strictly speaking energy) of the output signals $\boldsymbol{y}(t)$. In other words, the learning algorithm has an inherent self-adaptive gain control mechanism due to the self-adaptive synaptic weights $\hat{w}_{ii}$.[2] Third, our learning algorithm allows to separate the source signals with an extremely wide range of amplitudes (energies). Moreover, the mixing matrix can be very ill-conditioned, i.e., the sensors may have almost identical transmission parameters. This feature has been fully confirmed by extensive computer simulation experiments (see Section V).

Remark: Strictly speaking, the overall performance of the proposed learning algorithms (15)–(17) is independent of the scaling factors and/or condition number of the mixing matrix $\boldsymbol{A}$. It is easy to show this by multiplying on the right-hand side of (15) by a nonsingular mixing matrix $\boldsymbol{A}$, yielding

$$\frac{d\tilde{\boldsymbol{P}}(t)}{dt} = \mu(t)\{\boldsymbol{\Lambda} - \boldsymbol{f}[\tilde{\boldsymbol{P}}(t)\boldsymbol{s}(t)]\boldsymbol{g}^T[\tilde{\boldsymbol{P}}(t)\boldsymbol{s}(t)]\}\tilde{\boldsymbol{P}}(t)$$

where $\tilde{\boldsymbol{P}}(t) = \hat{\boldsymbol{W}}(t)\boldsymbol{A}$ will be called a performance matrix of the learning rule. The above matrix differential equation describes the dynamical behavior of the global "mixing–unmixing" system, which is completely independent of the mixing and scaling parameters. The same "equivariant" property have the algorithms developed independently very recently by J. F. Cardoso and his co-workers [6], [7], [30]. However, our algorithms are different from their proposals.

The open important theoretical problem is to prove rigorously that the performance matrix $\tilde{\boldsymbol{P}}(t)$ tends to the generalized permutation matrix [cf. (2)]. More generally, the question arises what conditions and constraints should satisfy a matrix $\boldsymbol{G}(\boldsymbol{y})$ in a generalized learning rule

$$\frac{d\hat{\boldsymbol{W}}(t)}{dt} = \mu(t)\boldsymbol{G}[\boldsymbol{y}(t)]\hat{\boldsymbol{W}}(t)$$

in order to ensure such a convergence? A closely related open problem is the choice of such activation functions $f_i(y_i)$ and

[1] This feature is not relevant in a continuous-time hardware realization.

[2] Note that in the original H-J model $w_{ii} = 0$ for any i.

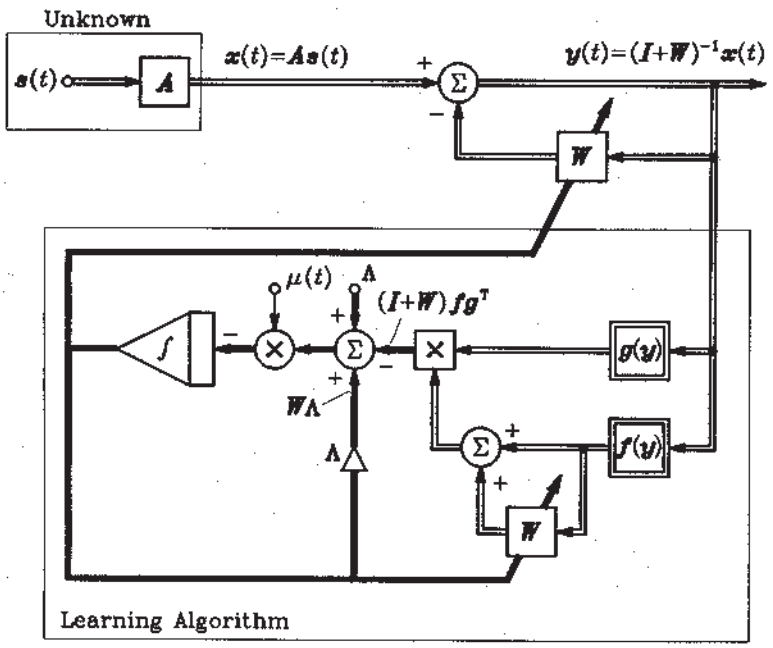


Fig. 4. Functional block diagram illustrating the implementation of the new adaptive learning algorithm (28) for a fully recurrent (feedback) architecture of the neural network.

$g_j(y_j)$ in (15)–(17) that ensure optimal convergence to one of the solutions.

From many possible kinds of odd activation functions which can be chosen in the algorithms (15)–(17) we have found by extensive computer simulations that the following pair of functions usually ensure convergence to a desired solution (i.e., successfully separate of independent sources for negative kurtosis):

$$f_i(y_i) = \begin{cases} y_i^p \operatorname{sign}(y_i) & \text{for } p \text{ even} \\ y_i^p & \text{for } p \text{ odd,} \end{cases} \quad \text{and}$$

$$g_j(y_j) = \tanh(10y_j) \quad \forall i,j \qquad (p = 1,2,3,4,5).$$

For a sper-Gaussian source the functions $f_i(y_i)$ and $g_j(y_j)$ should be interchanged. However, we do not claim that such choices are optimal and/or ensure convergence for any distribution of sources.

A partial solution of the above mentioned problems will be given in the future publications [25], [26]. However, this is out of the scope of this paper.

4.2 Feedback (Recurrent) Neural Network

The feedforward neural network model considered in the previous subsections is very simple, relatively easy to implement either in software or in VLSI technology and it is temporally stable, independent of the initial conditions. However, one serious disadvantage of this model is that in some cases the values of the synaptic weights $\hat{w}_{ij}$, especially, for ill-conditioned mixture matrices $\boldsymbol{A}$ and/or badly scaled or weak source signals, can take large values. This is rather inconvenient if the neural network is realized in hardware because some building blocks like op-amps can be saturated. Moreover, the realization of very large values of the synaptic weights is rather difficult for analog network implementations. This feature can be easily explained by analyzing (13) and (14). For example, for badly scaled signals some entries of the diagonal matrix $\boldsymbol{D}$ are large in comparison to others. On the other hand, for an ill-conditioned(nearly singular) mixing matrix $\boldsymbol{A}$ the entries of the inverse matrix $\boldsymbol{A}^{-1}$ can be very large.

The above disadvantage can be avoided by employing a feedback (fully recurrent) neural network described by the set of equations

$$y_i(t) = x_i(t) - \sum_{j=1}^{n} w_{ij}(t) y_j(t) \qquad (i = 1,2,\cdots,n) \quad (19a)$$

or in matrix form

$$\boldsymbol{y}(t) = \boldsymbol{x}(t) - \boldsymbol{W}(t)\boldsymbol{y}(t). \quad (19b)$$

Hence

$$\boldsymbol{y}(t) = [\boldsymbol{I} + \boldsymbol{W}(t)]^{-1}\boldsymbol{x}(t). \quad (20)$$

Note that in contrast to the Herault-Jutten neural network model we assume that our model is fully recurrent, i.e., it contains apart from the ordinary feedback loops also self-loops with $w_{ii}(t) \neq 0$. In other words, in the fully recurrent neural network there are connections between any pair of processing units and also from a unit to itself. The proposed fully recurrent neural network after training (learning phase) satisfies the following matrix relationships

$$\boldsymbol{y}(t) = [\boldsymbol{I} + \boldsymbol{W}]^{-1}\boldsymbol{x}(t) = (\boldsymbol{I} + \boldsymbol{W})^{-1}\boldsymbol{A}\boldsymbol{s}(t) = \boldsymbol{D}\boldsymbol{P}\boldsymbol{s}(t). \quad (21)$$

Hence

$$\boldsymbol{W} = \boldsymbol{A}\boldsymbol{P}^{-1}\boldsymbol{D}^{-1} - \boldsymbol{I} \quad (22)$$

or

$$\boldsymbol{A} = (\boldsymbol{I} + \boldsymbol{W})\boldsymbol{D}\boldsymbol{P}. \quad (23)$$

Now the problem arises how to develop a suitable learning algorithm for the recurrent (feedback) model? This problem

can be reformulated as follows: how to transform the learning algorithms proposed for a feed-forward neural network model [cf. (15), (17)] into equivalent learning rules for the feedback model discussed in this subsection? This can be made relatively easily by noting that

$$\hat{\boldsymbol{W}}(t) = [\boldsymbol{I} + \boldsymbol{W}(t)]^{-1} \tag{24}$$

where $\hat{\boldsymbol{W}}(t)$ and $\boldsymbol{W}(t)$ are the matrices of the synaptic weights of the feed-forward and feedback model, respectively. Note that the learning algorithm (15) can be transformed to

$$-\hat{\boldsymbol{W}}^{-1}(t)\frac{d\hat{\boldsymbol{W}}(t)}{dt}\hat{\boldsymbol{W}}^{-1}(t) = -\mu(t)\hat{\boldsymbol{W}}^{-1}(t)[\boldsymbol{\Lambda} - \boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{g}^T[\boldsymbol{y}(t)]] \tag{25}$$

on the assumption that the matrix $\hat{\boldsymbol{W}}(t)$ is nonsingular (i.e., $\det \hat{\boldsymbol{W}}(t) \neq 0$) for any t. Taking into account that

$$\hat{\boldsymbol{W}}^{-1}(t) = \boldsymbol{I} + \boldsymbol{W}(t) \tag{26}$$

and[3]

$$-\hat{\boldsymbol{W}}^{-1}(t)\frac{d\hat{\boldsymbol{W}}(t)}{dt}\hat{\boldsymbol{W}}^{-1}(t) = \frac{d\hat{\boldsymbol{W}}^{-1}(t)}{dt} = \frac{d\boldsymbol{W}(t)}{dt} \tag{27}$$

we obtain a new adaptive learning algorithm for the feedback neural network

$$\frac{d\boldsymbol{W}(t)}{dt} = -\mu(t)[\boldsymbol{I} + \boldsymbol{W}(t)]\{\boldsymbol{\Lambda} - \boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{g}^T\boldsymbol{y}(t)]\} \quad \text{with} \quad \boldsymbol{W}(0) \neq -\boldsymbol{I} \quad (\text{typically } \boldsymbol{W}(0) = \boldsymbol{0}). \tag{28}$$

The above learning algorithm can be written in scalar form as

$$\frac{dw_{ij}(t)}{dt} = -\mu(t)\left[\lambda_i w_{ij}(t) - \left(f_i[y_i(t)] + \sum_{p=1}^{n} w_{ip}(t) \cdot f_p[y_p(t)]\right) g_j[y_j(t)]\right], \quad \text{for} \quad i \neq j \tag{29a}$$

and

$$\frac{dw_{ij}(t)}{dt} = -\mu(t)\left[\lambda_i(w_{ii}(t) + 1) - \left(f_i[y_i(t)] + \sum_{p=1}^{n} w_{ip}(t) \cdot f_p[y_p(t)]\right) g_i[y_i(t)]\right], \quad \text{for} \quad i = j \tag{29b}$$

where $\mu(t) > 0, \lambda_i > 0$, typically $\lambda_i = 1, \forall i$. A functional block diagram illustrating the learning algorithm (28) is shown in Fig. 4.

Of course the above adaptive continuous-time learning algorithm can be easily transformed to a discrete-time algorithm by using, e.g., the Euler rule

$$\boldsymbol{W}(k+1) = \boldsymbol{W}(k) - \eta(k)[\boldsymbol{I} + \boldsymbol{W}(k)][\boldsymbol{\Lambda} - \boldsymbol{f}[\boldsymbol{y}(k)]\boldsymbol{g}^T[\boldsymbol{y}(k)]]. \tag{29c}$$

In this case the algorithm must be properly initialized (typically by the zero initial condition, i.e., $\boldsymbol{W}(0) = \boldsymbol{0}$) and the learning rate must be sufficiently small, otherwise an explosive (unstable) behavior is to be expected.

[3]The relationship (27) can be easily obtained by noting that $\frac{d[\hat{\boldsymbol{W}}^{-1}(t)\hat{\boldsymbol{W}}(t)]}{dt} = \hat{\boldsymbol{W}}^{-1}(t)\frac{d\hat{\boldsymbol{W}}(t)}{dt} + \frac{d\hat{\boldsymbol{W}}^{-1}(t)}{dt}\hat{\boldsymbol{W}}(t) = \boldsymbol{0}$

V. Experimental Results

The algorithms proposed in this paper have been extensively simulated on a computer. Computer simulations fully confirmed the validity and high performance of these algorithms for stationary signals[4] [21]. In this section we will illustrate the performance by presenting a few illustrative examples.

Example 1: Consider a set of two sensors receiving an unknown mixture of two unknown zero-mean independent signals. In the experiment a mixture of the following source signals are used

$$s_1(t) = 0.5 \sin[500t + 5\cos(60t)] \tag{30}$$

and

$$s_2(t) = 0.7 \sin(450t)\sin(40t). \tag{31}$$

A parameter (mixing) matrix $\boldsymbol{A}$ is chosen randomly as

$$\boldsymbol{A} = \begin{bmatrix} 0.57 & -0.43 \\ 0.89 & -0.92 \end{bmatrix}. \tag{32}$$

It was assumed that only the mixed (sensor) signals $x_1(t)$ and $x_2(t)$ are observable. We use here the neural network of Figs. 3(b) and 4 with initial conditions $\hat{\boldsymbol{W}}(0) = \boldsymbol{I}$ and $\boldsymbol{W}(0) = \boldsymbol{0}$, respectively. The scaling matrix was $\boldsymbol{\Lambda} = \boldsymbol{I}$. As activation function we used

$$f(y) = y^2 \operatorname{sign} y \quad \text{and} \quad g(y) = \tanh(10y).$$

The learning rate $\mu(t)$ was exponentially decreasing to zero. The computer simulation results are shown in Fig. 5(a), (b), and (c). After a few hundred milliseconds the neural networks achieve the following stationary (fixed) synaptic weights

$$\hat{\boldsymbol{W}} = \begin{bmatrix} 8.274 & -3.880 \\ -8.340 & 5.316 \end{bmatrix}, \quad \boldsymbol{W} = \begin{bmatrix} -0.544 & 0.330 \\ 0.715 & -0.296 \end{bmatrix}.$$

Consider now the case in which the mixing matrix $\boldsymbol{A}$ is very ill-conditioned:

$$\boldsymbol{A} = \begin{bmatrix} 2 & 3 \\ 2 & 3.0001 \end{bmatrix}. \tag{33}$$

Note that in this case the mixing (sensor) signals are almost identical. The both learning algorithms proposed (cf. Figs. 3(b) and 4) were successfully able to separate the primarily source signals $s_1(t)$ and $s_2(t)$. The feedforward and feedback (fully recurrent) neural networks have achieved the following synaptic weights

$$\hat{\boldsymbol{W}} = \begin{bmatrix} 18847.839 & -18847.207 \\ -12\,708.883 & 12\,708.882 \end{bmatrix} \tag{34}$$

and

$$\boldsymbol{W} = \begin{bmatrix} 0.585933 & 2.309848 \\ -1.585\,933 & 1.309\,925 \end{bmatrix}. \tag{35}$$

Exemplary computer simulation results are shown in Fig. 5(d).

We found that the neural networks shown in Figs. 3(b) and 4 perform the separation of signals surprisingly well even if

[4]For nonstationary signals the proposed algorithms may fail to converge because the synaptic weights w_{ij} depend not only on the characteristic of the mixture, but also on the energy of the signals. The authors are grateful to one of the reviewers for pointing out this problem.

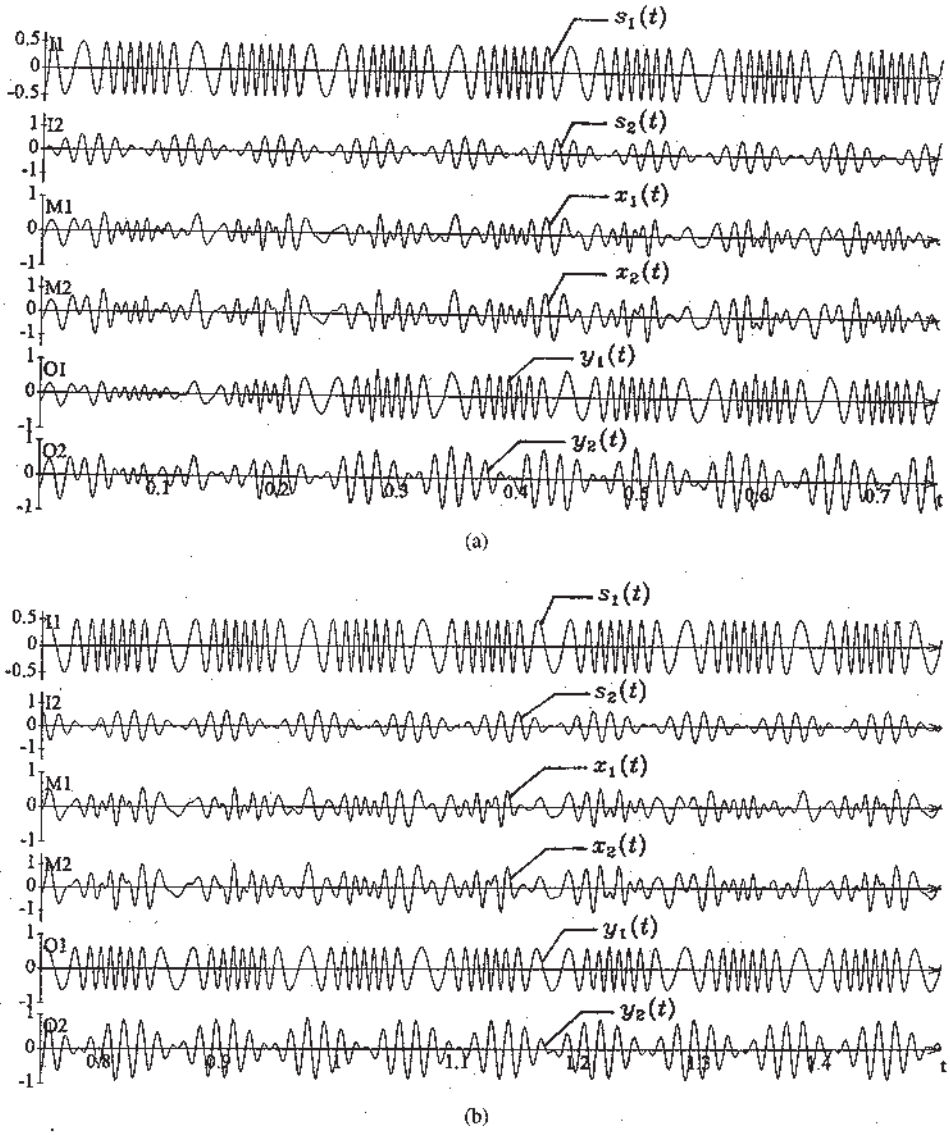


Fig. 5. Computer simulation results for Example 1. (a) Waveforms for the well-conditioned mixing matrix A simulated by employing a feedforward neural network. (b) Continuation of (a).

the source signals are extremely badly scaled and/or if the mixing matrix is very ill-conditioned. In the next experiment we used the ill-conditioned matrix (33) and two very badly scaled source signals: one with amplitude 1 nV (nanovolt) and the other with amplitude 1 V, i.e.,

$$s_1(t) = 10^{-9}\sin(500t + 5\cos 60t)$$
$$s_2(t) = \sin(450t)\sin(40t).$$

Note the amplitude ratio of the signals is $10^9 : 1$. Of course, in this case the small signal is absolutely not visible from the measured (observed) mixed signals $x_1(t)$ and $x_2(t)$. However, the weak signal as well as the large one are successfully and completely retrieved by the neural network of Fig. 4 as shown in Fig. 5(e) and (f).

By numerous computer simulations we have found that the proposed adaptive learning algorithm is extremely robust and provides a successful separation even with a larger amplitude ratio than $10^9 : 1$. In fact, this ratio depends only on the accuracy of the computer on which the simulations are performed.

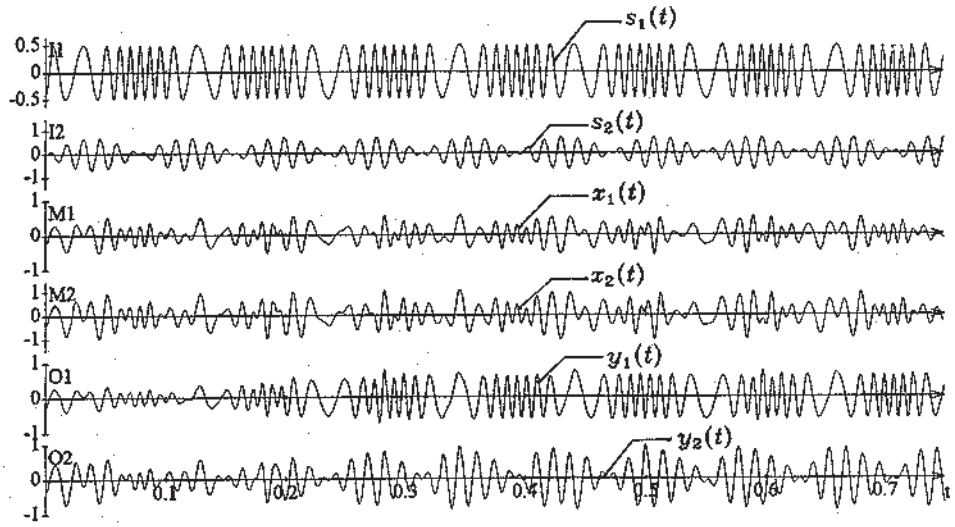
Example 2: The very badly scaled and weak source signals

$$s_1(t) = 10^{-5}\,\mathrm{sign}[\cos(155t)],$$
$$s_2(t) = 10^{-4}\sin(800t)\sin(60t),$$
$$s_3(t) = 10^{-3}\sin[300t + 6\cos(61t)],$$
$$s_4(t) = 10^{-2}\sin(90t),$$

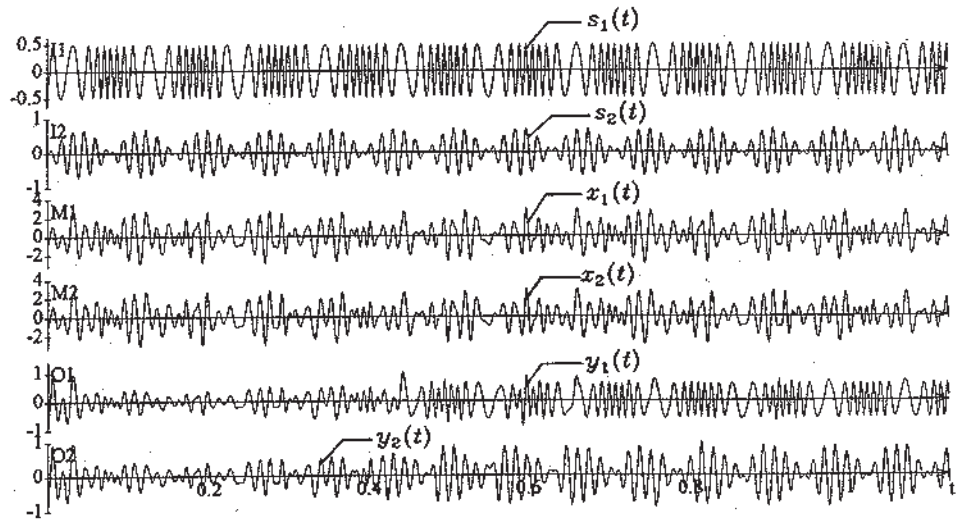
were mixed together with a large uniformly distributed noise ($s_5(t) = n(t)$) with amplitude 1. The mixing matrix A was chosen randomly as

$$A = \begin{bmatrix} 0.70 & 0.15 & -0.22 & 0.12 & -0.48 \\ -0.92 & -0.90 & 0.27 & -0.93 & -0.69 \\ 0.40 & -0.91 & 1.00 & 0.78 & 0.50 \\ -0.88 & 0.99 & 0.10 & -0.78 & 0.29 \\ 0.84 & -0.33 & 0.02 & -0.26 & 0.51 \end{bmatrix}. \quad (36)$$

It was assumed that only the sensor signals ($\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{s}(t)$) are available. In order to ensure a high speed convergence, we



(c)



(d)

Fig. 5. *(Continued.)* Computer simulation results for Example 1. (c) Waveforms for the well-conditioned mixing matrix simulated by using the fully recurrent neural network. (d) Waveforms for the ill-conditioned mixing matrix simulated by using fully recurrent neural networks. Note that the mixing signals $x_1(t)$ and $x_2(t)$ are in this case almost identical.

applied here the "search then converge" strategy. In the first learning phase the learning rate was set $\mu(t) = 200 = \text{const}$ for $0 \leqq t < 0.25s$ (search phase), then it was exponentially decreasing to zero according to the relation

$$\mu(t) = 200 \exp\left[-6(t - t_0)\right] \quad \text{for} \quad t \geqq t_0 = 0.25s.$$

The activation functions were chosen

$$f(y) = y^3 \quad \text{and} \quad g(y) = \operatorname{sign}(y).$$

The scaling coefficients were $\lambda_i = 0.25\ \forall i$. Exemplary computer simulation results are shown in Fig. 6(a), (b), and (c). Note that the neural network is able to estimate the source signals after a few hundred milliseconds.

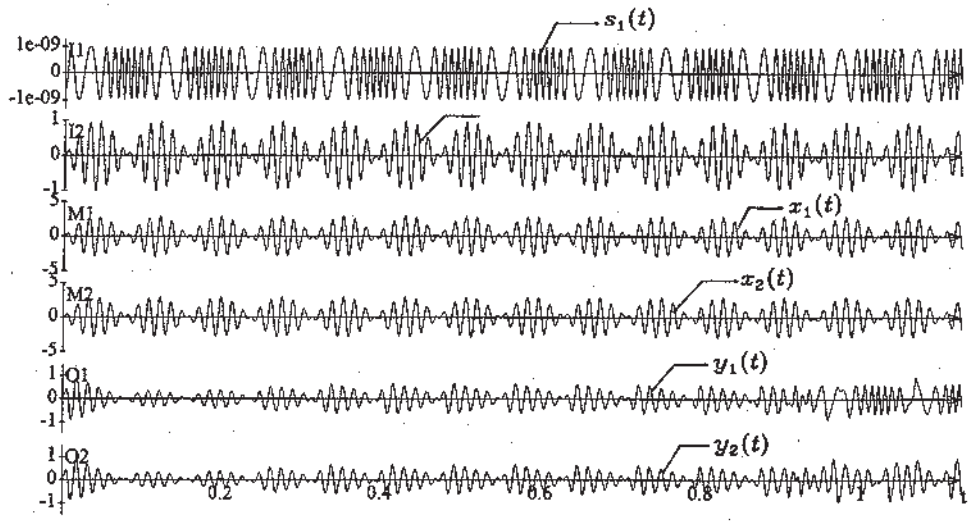
VI. Concluding Remarks

In this paper we have proposed two unsupervised, on-line, nonlinear, adaptive learning algorithms for blind identification and/or blind separation of independent sources. One algorithm is proposed for a linear single layer feedforward neural network, while the second one is derived for an equivalent feedback (fully recurrent) architecture of neural network. The proposed algorithms are on-line because they do not necessarily require storage of input data. Moreover, they are extremely robust because they satisfactorily perform a separation of signals even when the mixing matrix is very ill-conditioned. Extensive computer simulation experiments have shown that the proposed algorithms outperform most of the well-known on-line, adaptive learning algorithms, especially, for badly scaled signals and/or ill-conditioned problems.

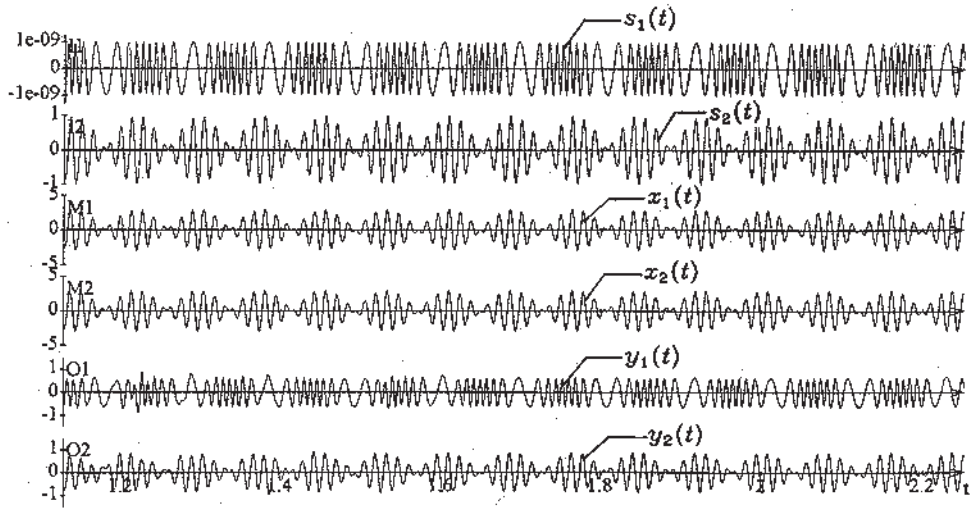
Appendix

In this Appendix we derive(or, strictly speaking, make some justification for) the learning algorithm proposed in Section III.

At the beginning let us assume that the source signals $s(t) = [s_1(t), s_2(t), \cdots, s_n(t)]^T$ are stationary zero-mean mutually



(e)



(f)

Fig. 5. *(Continued.)* Computer simulation results for Example 1. (e) Waveforms for the ill-conditioned mixing matrix and badly scaled source signals. (f) Continuation of (e).

decorrelated, i.e., the corresponding correlation matrix $\boldsymbol{R}_{ss}$ is a diagonal matrix

$$\boldsymbol{R}_{ss} \triangleq E[\boldsymbol{s}(t)\boldsymbol{s}^T(t)] = \boldsymbol{D}_s = \boldsymbol{D}_s^{1/2}\boldsymbol{D}_s^{1/2} \tag{A-1}$$

where E means the expectation value and $\boldsymbol{D}_s$ is a diagonal matrix with positive constant entries. The source signals are mixed together by the unknown nonsingular matrix $\boldsymbol{A}$ to give the observed (measured) vector $\boldsymbol{x}(t)$ as

$$\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{s}(t). \tag{A-2}$$

On the basis of the observed vector $\boldsymbol{x}(t)$ it is required to find a new matrix $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ which ensures that the output signals

$$\boldsymbol{y}(t) = \boldsymbol{W}\boldsymbol{x}(t) = \boldsymbol{W}\boldsymbol{A}\boldsymbol{s}(t) \tag{A-3}$$

are also mutually decorrelated, i.e.,

$$\begin{aligned}\boldsymbol{R}_{yy} &= E[\boldsymbol{y}(t)\boldsymbol{y}^T(t)] = E[\boldsymbol{W}\boldsymbol{x}(t)\boldsymbol{x}^T(t)\boldsymbol{W}^T] \\ &= \boldsymbol{W}\boldsymbol{R}_{xx}\boldsymbol{W}^T = \boldsymbol{W}\boldsymbol{A}\boldsymbol{R}_{ss}\boldsymbol{A}^T\boldsymbol{W}^T = \boldsymbol{W}\boldsymbol{A}\boldsymbol{R}_{ss}(\boldsymbol{W}\boldsymbol{A})^T \\ &= \boldsymbol{\Lambda}\end{aligned} \tag{A-4}$$

where $\boldsymbol{\Lambda}$ is the diagonal matrix $\boldsymbol{\Lambda} = \operatorname{diag}[\lambda_1, \lambda_2, \cdots, \lambda_n]$ with positive entries.

The autocorrelation matrix $\boldsymbol{R}_{xx}$ can be factorized as

$$\boldsymbol{R}_{xx} = E[\boldsymbol{x}(t)\boldsymbol{x}^T(t)] = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$$

where $\boldsymbol{D}$ is a diagonal matrix of the eigenvalues of $\boldsymbol{R}_{xx}$, and $\boldsymbol{V}$ is an orthogonal matrix of the associated eigenvectors. Setting $\boldsymbol{W} = \boldsymbol{V}^T$ we perform a principal component decomposition of the sensor signals $\boldsymbol{x}(t)$ with the output autocorrelation matrix

$$\boldsymbol{R}_{yy} = \boldsymbol{V}^T\boldsymbol{R}_{xx}\boldsymbol{V} = \boldsymbol{V}^T\boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{D} = \boldsymbol{\Lambda}.$$

In this paper we have applied a different approach in which the diagonal matrix $\boldsymbol{\Lambda}$ is assumed to be a given (specified) matrix, typically $\boldsymbol{\Lambda}$ is assumed to be the unit matrix, i.e., the main requirement is the orthogonalization and normalization of the output signals $y_i(t)$. It should be noted that the autocorrelation matrix $\boldsymbol{R}_{yy}$ can be expressed (factorized) as

$$\boldsymbol{R}_{yy} = \tilde{\boldsymbol{W}}\tilde{\boldsymbol{W}}^T \tag{A-5}$$

where $\tilde{\boldsymbol{W}} = \boldsymbol{W}\boldsymbol{A}\boldsymbol{D}_s^{1/2}$ and $\boldsymbol{D}_s^{1/2} = \boldsymbol{R}_{ss}^{1/2}$ is a diagonal matrix.

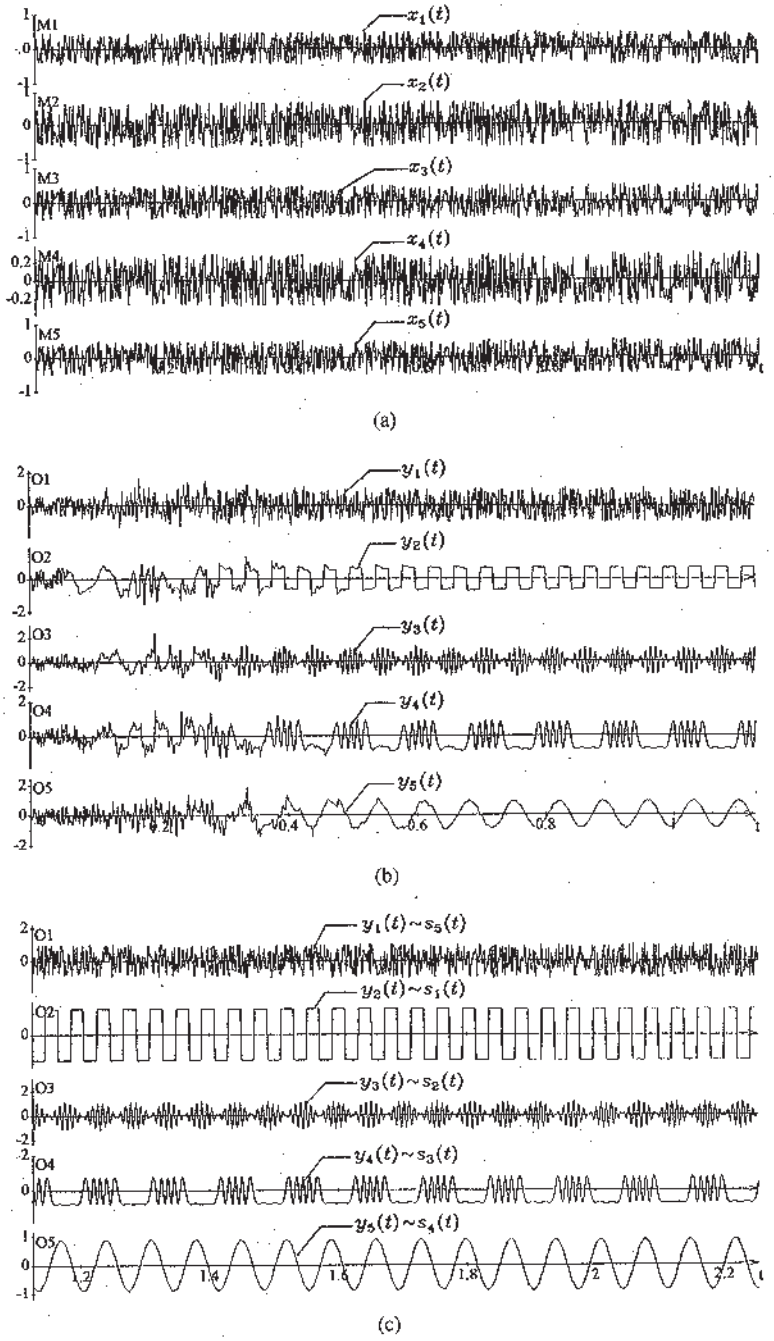


Fig. 6. Computer simulation results for Example 2. (a) Waveforms of the available mixing signals $x_1(t)$ to $x_5(t)$,. (b) Waveforms of the output signals $y_1(t) \div y_5(t)$ during the learning phase, employing a feedforward architecture. (c) Continuation of (b).

In order to develop an adaptive learning algorithm for iteratively updating the elements of the matrix $\boldsymbol{W}$ we can formulate the following cost (objective) function

$$E_c = \frac{1}{4}\left[\sum_{i=1}^{n}(r_{ii}-\lambda_i)^2 + \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n}|r_{ij}|^2\right] \tag{A-6}$$

where r_{ij} are elements of the correlation matrix $\boldsymbol{R}_{yy}$, i.e.,

$$r_{ij} = E[y_i(t)y_j(t)]. \tag{A-7}$$

The above cost function can be compactly written in matrix form as

$$E_c = \frac{1}{4}\,||\boldsymbol{R}_{yy} - \boldsymbol{\Lambda}||_F \tag{A-8}$$

where $||\cdot||_F$ means the Frobenius norm. Minimization of such a cost function forces that the correlation matrix $\boldsymbol{R}_{yy}$ tends to the diagonal matrix $\boldsymbol{\Lambda} = \text{diag}\,[\lambda_1, \lambda_2, \cdots, \lambda_n]$. As a first step let us determine the learning rule for the elements $\tilde{w}_{ij}$ of the matrix $\tilde{\boldsymbol{W}}$. Applying the standard gradient descent approach and next the chain rule we have

$$\frac{d\tilde{w}_{ij}}{dt} = -\mu\frac{\partial E_c}{\partial \tilde{w}_{ij}} = -\mu(t)\sum_{k=1}^{n}\sum_{l=1}^{n}\frac{\partial E_c}{\partial r_{kl}}\frac{\partial r_{kl}}{\partial \tilde{w}_{ij}} \tag{A-9}$$

where $\mu > 0$ is the learning rate. Taking into account that

$$\boldsymbol{R}_{yy} = \tilde{\boldsymbol{W}}\tilde{\boldsymbol{W}}^T, \tag{A-10}$$

where $\boldsymbol{R}_{yy} = [r_{ij}]_{n\times n}, \tilde{\boldsymbol{W}} = [\tilde{w}_{ij}]_{n\times n} \in \mathbb{R}^{n\times n}$, after some mathematical manipulations we obtain

$$\begin{aligned}\frac{d\tilde{w}_{ij}}{dt} &= \frac{\mu}{2}\left[\lambda_i \frac{\partial r_{ii}}{\partial \tilde{w}_{ij}} - \sum_{k=1}^{n}\sum_{l=1}^{n} r_{kl}\frac{\partial r_{kl}}{\partial \tilde{w}_{ij}}\right] \\ &= \frac{\mu}{2}\left[2\lambda_i \tilde{w}_{ij} - \sum_{k=1}^{n} r_{ik}\tilde{w}_{kj} - \sum_{l=1}^{n} r_{lj}\tilde{w}_{lj}\right]\end{aligned} \tag{A-11}$$

where $\lambda_i > 0$ typically $\lambda_i = 1\ \forall i$. The above formula can be simplified taking into account that the output correlation matrix is symmetric, i.e., $r_{ij} = r_{ji}$ as

$$\frac{d\tilde{w}_{ij}}{dt} = \mu\left[\lambda_i \tilde{w}_{ij} - \sum_{k=1}^{n} r_{ik}\tilde{w}_{kj}\right] \qquad (i,j = 1,2,\cdots,n). \tag{A-12}$$

The above differential equations can be written in compact matrix form as

$$\frac{d\tilde{\boldsymbol{W}}}{dt} = \mu\,[\boldsymbol{\Lambda} - \boldsymbol{R}_{yy}]\tilde{\boldsymbol{W}}. \tag{A-13}$$

Multiplying now the above equation by the nonsingular fixed matrices $\boldsymbol{D}_s^{-1/2}$ and $\boldsymbol{A}^{-1}$ we obtain

$$\frac{d\tilde{\boldsymbol{W}}}{dt}\boldsymbol{D}_s^{-1/2}\boldsymbol{A}^{-1} = \mu[\boldsymbol{\Lambda} - \boldsymbol{R}_{yy}]\tilde{\boldsymbol{W}}\boldsymbol{D}_s^{-1/2}\boldsymbol{A}^{-1}. \tag{A-14}$$

Hence taking into account (A-5) we get

$$\frac{d\boldsymbol{W}(t)}{dt} = \mu[\boldsymbol{\Lambda} - E(\boldsymbol{y}\boldsymbol{y}^T)]\boldsymbol{W}(t). \tag{A-15}$$

It should be noted that the cancellation of the term $[\boldsymbol{\Lambda} - E(\boldsymbol{y}\boldsymbol{y}^T)]$ is achieved when the vector $\boldsymbol{y}$ has uncorrelated components with corresponding variances λ_i. In the special case $\boldsymbol{\Lambda} = \boldsymbol{I}$ the cancellation of this term is obtained only if the vector $\boldsymbol{y}(t)$ will have uncorrelated unit-variance components.

Let us assume, without loss of generality, that the source signals have unit-variance, i.e., the autocorrelation matrix $\boldsymbol{R}_{ss} = E[\boldsymbol{s}(t)\boldsymbol{s}^T(t)] = \boldsymbol{I}$. Such an assumption can be made because scalar factors can be exchanged between the source signals and the corresponding columns of the mixing matrix $\boldsymbol{A}$ [cf. (2)].

For the unit-variance source signals the autocorrelation matrix $\boldsymbol{R}_{yy}$ of the output signals can be expressed as

$$\boldsymbol{R}_{yy} = \boldsymbol{W}\boldsymbol{A}\boldsymbol{R}_{ss}\boldsymbol{A}^T\boldsymbol{W} = (\boldsymbol{W}\boldsymbol{A})(\boldsymbol{W}\boldsymbol{A})^T = \hat{\boldsymbol{P}}\hat{\boldsymbol{P}}^T \tag{A-16}$$

where

$$\hat{\boldsymbol{P}} = \boldsymbol{W}\boldsymbol{A}.$$

For this case the learning algorithm (A-15) can be expressed as

$$\begin{aligned}\frac{d\boldsymbol{W}(t)}{dt} &= \mu(\boldsymbol{I} - \hat{\boldsymbol{P}}(t)\hat{\boldsymbol{P}}^T(t))\boldsymbol{W}(t) \\ &\text{with} \quad \boldsymbol{\Lambda} = \boldsymbol{I}, \quad \text{and} \quad \hat{\boldsymbol{P}}(t) = \boldsymbol{W}(t)\boldsymbol{A}.\end{aligned} \tag{A-17}$$

Multiplying (A-17) by the mixing matrix $\boldsymbol{A}$ we get

$$\frac{d\hat{\boldsymbol{P}}(t)}{dt} = \mu(\boldsymbol{I} - \hat{\boldsymbol{P}}(t)\hat{\boldsymbol{P}}^T(t))\hat{\boldsymbol{P}}(t) \tag{A-18}$$

or

$$\frac{d\hat{\boldsymbol{P}}(t)}{dt} = \mu\hat{\boldsymbol{P}}(t)(\boldsymbol{I} - \hat{\boldsymbol{P}}^T(t)\hat{\boldsymbol{P}}(t)). \tag{A-19}$$

The above equation achieves an equilibrium point if the left-hand side of (A-18) or (A-19) goes to zero as time goes to infinity, i.e., if the matrix $\hat{\boldsymbol{P}}(t) = \boldsymbol{W}(t)\boldsymbol{A}$ becomes an orthogonal matrix satisfying the relations

$$\hat{\boldsymbol{P}}\hat{\boldsymbol{P}}^T = \hat{\boldsymbol{P}}^T\hat{\boldsymbol{P}} = \boldsymbol{I} \tag{A-20}$$

or

$$\hat{\boldsymbol{P}}^{-1} = \hat{\boldsymbol{P}}^T.$$

The orthogonal matrix $\hat{\boldsymbol{P}}$, in general, is not equal to the generalized permutation matrix $\tilde{\boldsymbol{P}}$ [cf. (2)], therefore, the algorithm (A-15) does not ensure a blind separation of the sources, but only a decorrelation and normalization of the output signals. In other words, the conditions given by (A-20) are necessary but not sufficient to provide mutual independence of the normalized (unit-variance) output signals. Note that the learning algorithm (A-15) utilizes the second-order statistics $(E(y_iy_j))$ (producing uncorrelated output signals) which are not sufficient to ensure their independence. In order to achieve mutual independence of the output signals it is necessary to replace the linear functions in $E(y_iy_j)$ by nonlinear and different odd functions $f(y)$ and $g(y)$ as [cf. (15)]

$$\frac{d\boldsymbol{W}(t)}{dt} = \mu[\boldsymbol{\Lambda} - E[\boldsymbol{f}[\boldsymbol{y}(t)]\boldsymbol{g}^T[\boldsymbol{y}(t)]]]\boldsymbol{W}(t). \tag{A-21}$$

The main justification of using the nonlinear function $\boldsymbol{f}[\boldsymbol{y}(t)]$ and $\boldsymbol{g}[\boldsymbol{y}(t)]$ in the algorithm is that they introduce higher-order statistics into the computations [1]–[3], [16], [17]. In fact, the general condition of independence of the the signals is to cancel higher order cross cumulants [6]. It is not easy to verify the independence of signals exactly, because one should know or estimate the associated probability densities. In general, higher-order moments or more exactly, the generalized moments $E[f(y_i)g(y_j)]$ for $i \neq j$ are required to vanish in order to ensure independence of the signals. In practice, the expectation values $E[f(y_i)g(y_j)]$ are not available and they are approximated by their instantaneous values in the final stochastic gradient algorithm (15) and (16).

ACKNOWLEDGMENT

The authors wish to thank Prof. C. Jutten, Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS and the anonymous reviewers of this paper for their constructive criticism and suggestions for improvements. The authors are also grateful to E. Rummert and L. Moszczynski for helpful suggestions and for making extensive computer simulation experiments. One of the authors (A. Cichocki) would like to thank Prof. S. Amari for very valuable discussions, helpful guidance, and kind interest.

REFERENCES

[1] J. Herault and C. Jutten, "Space or time adaptive signal processing by neural network models," in *Proc. AIP Conf.* Snowbird, UT, 1986, in *Neural Networks for Computing*, J. S. Denker, Ed. New York: Amer. Inst. Phys., 1986, pp. 206–211.

[2] C. Jutten and J. Herault, "Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, pp. 1–20, 1991.

[3] P. Comon, C. Jutten, and J. Herault, "Blind separation of sources, Part II: Problem statement," *Signal Processing*, vol. 24, pp. 11–20, 1991.

[4] E. V. Vittoz and X. Arreguit, "CMOS integration of Herault-Jutten cells for separation of sources," in *Workshop of Analog VLSI and Neural Systems*, Portland, OR. Norwell, MA, Kluwer Academic, 1989, pp. 57–83.

[5] M. H. Cohen and A. G. Andreou, "Current-mode subthreshold MOS implementation of Herault-Jutten autoadaptive network," *IEEE J. Solid-State Circuits*, vol. 27, pp. 714–727, May 1992.

[6] J. F. Cardoso, A. Belouchrani, and B. Laheld, "A new composite criterion for adaptive and iterative blind source separation," in *Proc. ICASSP-94*, Adelaide, Australia, Apr. 1994, pp. 273–276.

[7] B. Laheld and J. F. Cardoso, "Adaptive source separation without prewhitening," in *Proc. EUSIPCO*, Edinburgh, Sept. 1994 pp. 183–186.

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

[9] L. Tong, R. Liu, V. C. Soon, and Y.-F. Huang, "Indeterminacy and identifiability of blind identification," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 499–509, May 1991.

[10] J. L. Lacoume and P. Ruiz, "Separation of independent sources from correlated inputs," *IEEE Trans. Signal Processing* vol. 40, pp. 3074–3078, Dec. 1992.

[11] C. Jutten, H. L. Nguyen Thi, E. Dijkstra, E. Vittoz, J. Caelen, "Blind separation of sources: an algorithm for separation of convolutive mixtures," in *Int. Wkshp. High Order Statistics*, Chamrousse, France, July 1991, pp. 273–276.

[12] H. L. Nguyen Thi and C. Jutten, "Blind separation of sources: Algorithms for convolutive mixture of large bandwidth signals," *Signal Processing*, vol. 45, no. 2, 1995, pp. 209–229.

[13] J. C. Platt and F. Faggin, "Networks for the separation of sources that are superimposed and delayed," *Advances in Neural Information Processing Systems*, vol. 4. San Mateo, CA: Morgan Kaufman, 1992, pp. 730–737.

[14] E. Weinstein, M. Feder, and A.V. Oppenheim, "Multi-channel separation by decorrelation," *IEEE Trans. Speech Audio Processing*, vol. 1, pp. 405–413, 1993.

[15] D. Van Compernolle and S. Van Gerven, "Signal separation in a symmetric adaptive noise canceler by output decorrelation," in *Proc. ICASSP92*, San Francisco, CA, 1992, vol. IV, pp. 221–224.

[16] E. Oja and J. Karhunen, "Signal separation by nonlinear Hebbian learning," in *Computational Intelligence*. New York: New York Press, 1995, pp. 83–97.

[17] J. Karhunen and J. Joutsensalo, "Representation and separation of signals using PCA type learning", *Neural Networks*, vol. 7, pp. 113–127, 1994.

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

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

[20] S. Li and T. Sejnowski, "Adaptive separation of mixed sound sources with delays by a beamforming Herault-Jutten network," *IEEE J. Oceanic Eng.*, vol. 20, 1995, pp. 73–79.

[21] E. Rummert, "Neural networks for blind separation of signals," (in German) M.S. degree thesis at the University Erlangen-Nüernberg, Oct. 1993 under guidance and supervision of the authors.

[22] A. Cichocki and R. Unbehauen, *Neural Network for Optimization and Signal Processing*. New York: Teubner-Wiley, 1994, pp. 461–471.

[23] A. Cichocki and L. Moszczynski, "New learning algorithm for blind separation of sources," *Electron. Lett.*, vol. 28, pp. 1986–1987, 1992.

[24] A. Cichocki, R. Unbehauen, L. Moszczynski, and E. Rummert, " A new on-line adaptive learning algorithm for blind separation of source signals," in *Proc. 1994 Int. Symp. on Artificial Neural Networks*, ISANN '94, Tainan, Taiwan, Dec. 1994, pp. 406–411.

[25] S. Amari, A. Cichocki, and H. H. Yang, "A new learning algorithm for blind signal separation," *Advances in Neural Information Processing Systems, NIPS*, vol. 7. Cambridge, MA: MIT Press 1996, pp. 657–663.

[26] ——, "Recurrent neural networks for blind separation of sources," in *Proc. Int. Symp. Nonlinear Theory Applicat.*, NOLTA '95 Las Vegas, NV, Dec. 10–14, 1995.

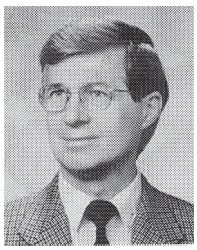
[27] A. Cichocki, W. Kasprzak, and S. Amari, "Multi-layer neural networks with a local adaptive learning rule for blind separation of source signals," in *Proc. Int. Symp. Nonlinear Theory Applicat.*, NOLTA '95 Las Vegas, NV, Dec. 10–14, 1995.

[28] A. Cichocki, R. E. Bogner, and L. Moszczynski, "Improved adaptive algorithms for blind separation of sources," in *Proc. Conf. Electron. Circuits Syst.*, KKTOIUE-95, Zakopane, Poland, Oct. 25–28, 1995, pp. 647–652.

[29] A. Cichocki, R. Unbehauen, and E. Rummert, "Robust learning algorithm for blind separation of signals," *Electron. Lett.*, vol. 30, no. 17, 1994, pp. 1386–1387.

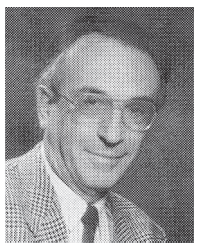
[30] J.-F. Cardoso and B. Laheld, "Equivariant adaptive source separation," *IEEE Trans. Signal Processing*, to be published.

[31] E. Moreau and O. Macchi, "A complex self-adaptive algorithm for source separation based on higher order contrast," in *Signal Processing VII*, EUSIPCO-94, 1994, pp. 1157–1160.



Andrzej Cichocki received the M.Sc. (with honors), Ph.D., and Habilitate Doctorate (Dr.Sc.) degrees, all in electrical engineering, from Warsaw University of Technology, Poland, in 1972, 1975, and 1982, respectively.

Since 1972, he has been with the Institute of Theory of Electrical Engineering and Electrical Measurements at the Warsaw University of Technology, where he became a full Professor in 1991. He is the coauthor of two books: *MOS Switched-Capacitor and Continuous-Time Integrated Circuits and Systems* (Springer-Verlag, 1989) and *Neural Networks for Optimization and Signal Processing* (Teubner-Wiley, 1993). He spent a few years at the University Erlangen-Nuernberg (Germany) as an Alexander Humboldt Research Fellow and Guest Professor, at Lehrstuhl fuer Allgemeine und Theortische Elektrotechnik conducted by Professor R. Unbehauen. Currently he is working as a team leader of the Laboratory for Artificial Brain Systems, Frontier Research Program RIKEN, Japan, in the Brain Information Processing Group conducted by Professor S. Amari. His current research interests include neural networks and nonlinear dynamic systems theory.



Rolf Unbehauen (M'61–SM'82–F'91) received the diploma in mathematics, the Ph.D. degree in electrical engineering and the Habilitate Doctorate in Electrical Engineering from Stuttgart University, Stuttgart, Germany, in 1954, 1957, and 1964, respectively.

From 1965 and 1966 he was a member of the Institute of Mathematics, the Computer Center and the Institute of Electrical Engineering at Stuttgart University, where he was appointed Associate Professor in 1965. Since 1966 he has been Full Professor of Electrical Engineering at the University of Erlangen-Nürnberg, Erlangen, Germany. His teaching and research interests are in the areas of network theory and its applications, system theory and electromagnetics. He has published many papers on his research results. He has authored four books in German and is coauthor of *MOS Switched-Capacitor and Continuous-Time Integrated Circuits and Systems* (Springer, 1989) and *Neural Networks for Optimization and Signal Processing* (Teubner Verlag and Wiley, 1993).

Dr. Unbehauen is a Member of the Informationstechnische Gesellschaft of Germany and of URSI, Commission C: Signals and Systems. From 1990 to 1991 he was an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS. Currently he is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS and of *Multidimensional Systems and Signal Processing*. In 1959 he received the NTG-Best-Paper-Award and in 1994 an honorary doctor from Technical University Cluj-Napoca (Romania).