Blind Separation of Uniformly Distributed Signals: A General Approach

Jayanta Basak, *Senior Member, IEEE,* and Shun-ichi Amari, *Fellow, IEEE*

***Abstract*— A general algorithm for blind separation of uniformly distributed signals is presented. First maximum likelihood equations are obtained for dealing with this task. It is difficult to obtain a closed form maximum likelihood solution for arbitrary mixing matrix. The learning rules are obtained based on the geometric interpretation of the maximum likelihood estimator. The algorithm, under special constraint of orthogonal mixing matrix, is the same as the $O(1/T^2)$ convergent algorithm. Special noise correction mechanisms are incorporated in the algorithm, and it has been found that the algorithm exhibits stable performance even in the presence of large amount of noise.**

***Index Terms*— Blind separation, maximum likelihood, natural gradient, neural networks.**

I. Introduction

BLIND separation [2]–[6], [24], [9]–[16], [18], [20]–[22] refers to the task of separating independent signal sources from the sensor outputs in which the signals are mixed in an unknown channel, a multiple-input multiple-output linear system. This problem is relevant and important in many applications including speech recognition, data communication, signal processing, and medical science.

Many algorithms have been proposed for dealing with the task of blind separation. These existing algorithms can be categorized into three major approaches: 1) independent component analysis (ICA); 2) entropy maximization; and 3) nonlinear principal component analysis. In the first approach, the signals are transformed in such a way that the dependency between individual signal components is minimized. The independent component analysis (ICA) was proposed by Comon [12] for this purpose (see also [5] and [24]). Different algorithms have been designed considering the different criteria measures for independence between the signals including [10].

In the second approach, the part of the information content of the output which is dependent on the input, as measured by the entropy, is maximized [9]. The output components are transformed by a nonlinear transfer function, so that the output distribution is contained within a finite hypercube. The maximization of the entropy forces the output components to be as uniformly spread over the hypercube as possible. The entropy maximization also leads to a similar measure of independence between the signal components.

Manuscript received February 12, 1998; revised January 4, 1999 and March 25, 1999.

J. Basak is with the Machine Intelligence Unit, Indian Statistical Institute, Calcutta-700 035, India.

S. Amari is with the Laboratory for Information Synthesis, RIKEN Brain Science Institute, 2-1 Hirosawa, Wako, Saitama 351-01, Japan.

Publisher Item Identifier S 1045-9227(99)05984-6.

These two approaches can be unified from the viewpoint of information geometry of the Kullback-Leibler divergence measure [17]. These algorithms, their statistical efficiency and dynamical stability are discussed in [2]–[6].

In the third approach, developed by Karhunen, Oja *et al.* [15], [16], [13], [21], [20], nonlinear principal component analysis algorithm is applied for separating the source signals. In the linear principal component analysis (PCA) algorithm [1], [19], the weight vectors in a connectionist framework, are stochastically approximated to the first eigenvector (with highest eigenvalue) of the correlation matrix of the input data, i.e., the principal component of the input. In the nonlinear principal component analysis, the output vector is generated by a nonlinear function of the weighted sum of the inputs (unlike the linear PCA where only the weighted sum is considered as the output). The nonlinear extension of the PCA rule is able to perform the task of source separation under a strong assumption that the signal components are mixed by some orthogonal mixing matrix. It has also been theoretically proved that the nonlinear PCA algorithm is able to perform the source separation under the orthogonality constraint.

In a completely different approach [22], assuming bounded input distributions, source signals were separated based on some geometric properties. However, no theoretical justification as to the rate of convergence of the algorithm was provided in [22].

In all of the major general purpose blind separation algorithms, it is assumed (implicitly or explicitly) that the source distributions have smooth differentiable form. However, if the source distributions are not differentiable (e.g., uniform distribution), some of the existing algorithms do not converge to the solution. Moreover, since the Fisher information diverges for such kind of distributions, the efficiency bound of $O(1/T)$ convergence (as provided by Cramer–Rao theorem [23]) is no more applicable. In such cases, it may be possible to design a much more efficient algorithm.

In [7] and [8], an algorithm has been presented for a specific case of uniform distribution with orthogonal mixing matrix. It has been theoretically proved that the algorithm is very similar to an $O(1/T^2)$ convergent superefficient algorithm. However, the algorithm in [7] is highly noise sensitive and applicable only to special orthogonal mixing matrices.

In the present article, we propose a general algorithm to separate a mixture of uniformly distributed sources mixed with arbitrary mixing matrix. It has been found that the existing general purpose algorithms including the Kullback–Leibler (K–L) divergence measure-based method [2]–[6] and EASI

algorithm [10] fail to converge in the case of arbitrary mixing matrix with uniformly distributed source signals. However there exist some algorithms including [13], [22] which can take care of uniformly distributed signals. In the proposed algorithm, the learning rule is derived in analogy with the maximum likelihood equations. In the limiting condition of orthogonal mixing matrix and zero noise, the algorithm becomes equivalent to the $O(1/t^2)$ convergent algorithm proposed in [7] and [8]. It has also been shown here that the maximum likelihood solution under these special constraints becomes exactly the same as in [7]. The algorithm also exhibits stability even in the presence of noise. This special property of the proposed algorithm in dealing with large amount of noise may also be extended in the case of the general purpose algorithms.

II. Problem

Let there be n independent signal sources $s_i(t); i = 1, 2, \cdots, n$ which are mixed by an unknown mixing matrix $\boldsymbol{A}$ to give rise to another n signal components $x_i(t); i = 1, 2, \cdots, n$, i.e.,

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

where $\boldsymbol{x}(t) = [x_1(t), x_2(t), \cdots, x_n(t)]'$ and $\boldsymbol{s}(t) = [s_1(t), s_2(t), \cdots, s_n(t)]'$. $[\cdot]'$ indicates the transpose of a vector or a matrix. The task is to estimate $\boldsymbol{A}$ only from the given signals $\boldsymbol{x}(t)$. In other words, a linear transformation $\boldsymbol{W}$ is to be estimated in such a way that

$$\boldsymbol{y}(t) = \boldsymbol{W}\boldsymbol{x}(t) \tag{2}$$

becomes a scaled permutation of $\boldsymbol{s}(t)$, i.e., $\boldsymbol{W}\boldsymbol{A} = \boldsymbol{C}$, $\boldsymbol{C}$ being an arbitrary scaled permutation matrix.

A basic assumption in the task of blind separation is that the source signals are considered to be independent, i.e.,

$$p(\boldsymbol{s}) = \prod_{i=1}^{n} q_i(s_i) \tag{3}$$

where $p(\boldsymbol{s})$ is the joint distribution of $\boldsymbol{s}$ and $q_i(s_i)$ is the marginal distribution of the individual (ith) components. Therefore, for perfect separation, one should have $p(\boldsymbol{y}) = \Pi_{i=1}^{n} q_i(y_i)$. In the independent component analysis, the error between the joint probability and the product of the marginal probability distributions is minimized.

The error between $p(\boldsymbol{y})$ and $\Pi_{i=1}^{n} q_i(y_i)$ can be measured by the K–L divergence measure (which is also referred to as the relative entropy). The K–L divergence measure between any two distributions $p_1(\boldsymbol{y})$ and $p_2(\boldsymbol{y})$ is defined as

$$\mathcal{D}[p_1 : p_2] = \int p_1(\boldsymbol{y}) \log \frac{p_1(\boldsymbol{y})}{p_2(\boldsymbol{y})}\, d\boldsymbol{y}. \tag{4}$$

The K–L divergence $\mathcal{D}[p(\boldsymbol{y}): \Pi_{i=1}^{n} q_i(y_i)]$ which is the same as the difference between the joint entropy and the sum of the marginal entropies of the output components, provides a measure of dependency between the output signal components. $\mathcal{D}[p(\boldsymbol{y}): \Pi_{i=1}^{n} q_i(y_i)]$ goes to zero when the output components are totally independent. The K–L divergence measure has been used in many algorithms including those developed by Amari *et al.* [2]–[6], [24]. In these algorithms, $\mathcal{D}[p(\boldsymbol{y}): \Pi_{i=1}^{n} q_i(y_i)]$ is adaptively minimized according to the natural gradient descent algorithm [2] without knowing the true probability distributions of the outputs. The learning rule in this method is given as

$$\frac{d\boldsymbol{W}}{dt} = \eta(\boldsymbol{I} - \boldsymbol{\phi}(\boldsymbol{y})\boldsymbol{y}')\boldsymbol{W} \tag{5}$$

where η is a small constant, and

$$\boldsymbol{\phi}(\boldsymbol{y}) = [\phi_1(y_1), \phi_2(y_2), \cdots, \phi_n(y_n)]'.$$

ϕ_i is given as $\phi_i(y_i) = -(\dot{p}_i/p_i)$ where p_i is an adequate probability density function, hopefully to be equal to the true source distribution. The entropy maximization criterion as proposed by Bell and Sejnowski [9] is also equivalent to minimizing the K–L divergence $\mathcal{D}[p : q]$ where q is an independent distribution.

In Cardoso and Laheld [10], the independence is achieved by first prewhitening (decorrelating) the input signal vector $\boldsymbol{x}$, i.e., transforming $\boldsymbol{x}$ to another vector $\boldsymbol{z}$ such that $\langle \boldsymbol{z}\boldsymbol{z}' \rangle = \boldsymbol{I}$. This is performed by minimizing the K–L divergence between two zero-mean normal distributions with covariance matrices $\langle \boldsymbol{z}\boldsymbol{z}' \rangle$ and $\boldsymbol{I}$, respectively. In the next stage, $\boldsymbol{z}$ is transformed to the output $\boldsymbol{y}$ by an orthogonal weight matrix. The learning rule obtained by combining these two stage is

$$\Delta\boldsymbol{W} = \eta[\boldsymbol{I} - \boldsymbol{y}\boldsymbol{y}' + \boldsymbol{y}\psi(\boldsymbol{y})' - \psi(\boldsymbol{y})\boldsymbol{y}']\boldsymbol{W} \tag{6}$$

where $\psi(\boldsymbol{y})$ is a nonlinear function of $\boldsymbol{y}$ and η is the learning rate. The learning rule has been derived based on the relative gradient minimization [10] which is similar to the natural gradient minimization [2].

The present paper treats a special case where the probability distribution of $\boldsymbol{s}(t)$ is independently identically distributed (i.i.d.) subject to the uniform distribution

$$p(\boldsymbol{s}) = \begin{cases} \dfrac{1}{2^n} & |s_i| \le 1, \text{ for all } i \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Here it is assumed that the signals are bounded in $[-1,1]$. In general, we can consider each s_i to be bounded in $[-k_i, k_i]$ where the each signal has a zero-mean distribution. However, we can validly assume that $y_i \in [-1,1]$ after perfect separation. This is because $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{A}\boldsymbol{s}$, and any scaling in $\boldsymbol{s}$ can be considered as a multiplying factor of $\boldsymbol{W}$, i.e., $\boldsymbol{W} = \boldsymbol{C}\boldsymbol{A}^{-1}$ where $\boldsymbol{C}$ is an arbitrary scaled permutation matrix. Therefore, in order to estimate $\boldsymbol{W}$ we can assume each s_i (which is actually y_i after perfect separation) is bounded in $[-1,1]$.

In the case of the uniform distribution, $\dot{p}_i$ does not exist. Therefore, it is not possible to apply the K–L divergence based method for uniformly distributed signals with arbitrary mixing matrix.

In [7] and [8], the special case of uniform distribution has been dealt with by considering only orthogonal mixing matrix, i.e., it was assumed that the input signals have already been prewhitened. Under such assumption of orthogonal mixing matrix and uniform source distribution, the learning rule was designed to update $\boldsymbol{W}$ such a way that the hypercube containing the output signals gets effectively rotated.

A measure analogous to the variational distance between the output distribution and the source distribution has been minimized. It has been proved that the algorithm in batch mode exhibits superefficiency of $O(1/T^2)$ convergence. However, the algorithm has been designed only for orthogonal mixing matrices and its performance in the noisy condition was poor.

In the rest of the article, we present a new algorithm for separating uniformly distributed sources for any arbitrary mixing matrix. It will be shown that the algorithm behaves reasonably well even in the presence of large amount of noise. It goes to the superefficiency of $O(1/T^2)$ in the limiting condition.

III. Maximum Likelihood Solution

The marginal distribution of the uniformly distributed signals in the presence of Gaussian noise can be written as

$$q_i(s_i) = \frac{1}{2}\left[\operatorname{erf}\left(\frac{s_i+1}{\sqrt{2}N}\right) + \operatorname{erf}\left(\frac{1-s_i}{\sqrt{2}N}\right) - 1\right] \tag{8}$$

where N is the noise amplitude, and $\operatorname{erf}(x)$ is defined as

$$\operatorname{erf}(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} \exp\left(-\frac{u^2}{2}\right) du.$$

The joint density is given as

$$p(\boldsymbol{s}) = \prod_{i=1}^{n} q_i(s_i). \tag{9}$$

For small noise amplitude, the Fisher information is mostly concentrated near the boundary of the hypercube. Therefore, the distribution can be approximately given as

$$p(\boldsymbol{s}) = K \exp\left(-\frac{D(\boldsymbol{s})}{\epsilon^2}\right) \tag{10}$$

where ϵ is a parameter dependent on the noise amplitude. K is the normalization parameter. $D(\boldsymbol{s})$ is given as

$$D(\boldsymbol{s}) = \sum_{i:|s_i|>1} (|s_i|-1)^2. \tag{11}$$

We get the uniform distribution by letting $\epsilon \to 0$.

Note that, it is possible to calculate the likelihood and the Fisher information formally by using the generalized function or distribution in the sense of Schwarts, even in the case of a uniform distribution which is not differentiable in the usual sense. However, this does not help us, because the Fisher information diverges to infinity. This is because the logarithm and the square of delta functions do not belong to the class of distributions. Moreover, one cannot derive a learning algorithm in terms of delta functions, because it is numerically unstable and inefficient. Under the assumption of uniformly distributed source signals, the mixing matrix can be estimated from the observed mixed signals by maximum likelihood estimate which is described as follows.

The mixed signals are given as

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s}. \tag{12}$$

Therefore,

$$p(\boldsymbol{x}) = \frac{p(\boldsymbol{s})}{|\det(\boldsymbol{A})|}. \tag{13}$$

Let there be T observations of $\boldsymbol{x}$ i.i.d. over $p(\boldsymbol{x})$ such that the log-likelihood function is given as

$$\begin{aligned} l(\boldsymbol{A}) &= \frac{1}{T}\log\left(\prod_{t=1}^{T} p(\boldsymbol{x}^{(t)})\right) \\ &= \frac{1}{T}\sum_{t=1}^{T}\log\left(p(\boldsymbol{x}^{(t)})\right) \\ &= -\log|\det(\boldsymbol{A})| + \frac{1}{T}\sum_{t=1}^{T}\log\left(p(\boldsymbol{s}^{(t)})\right). \end{aligned} \tag{14}$$

Therefore the maximum likelihood solution can be obtained by letting

$$\frac{\partial l(\boldsymbol{A})}{\partial \boldsymbol{A}} = 0 \tag{15}$$

and solving the simultaneous equations for the parameter values. The set of equations are given as

$$-(\boldsymbol{A}^{-1})' + \frac{1}{T}\frac{\partial}{\partial \boldsymbol{A}}\sum_{t=1}^{T}\log\left(p(\boldsymbol{s}^{(t)})\right) = 0. \tag{16}$$

From (10) and (12) we get

$$(\boldsymbol{A}^{-1})' + \frac{1}{T\epsilon^2}\sum_{t=1}^{T}\frac{\partial}{\partial \boldsymbol{A}} D(\boldsymbol{A}^{-1}\boldsymbol{x}^{(t)}) = 0. \tag{17}$$

Instead of estimating $\boldsymbol{A}$, we can estimate $\boldsymbol{W}$ such that $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ becomes a permutation of the original signal vector $\boldsymbol{s}$. Ideally $\boldsymbol{W} = \boldsymbol{A}^{-1}$. Instead of finding out a solution for (17), we can update $\boldsymbol{W}$ in such a way that (17) is satisfied for $\boldsymbol{W} + \boldsymbol{\Delta W}$, i.e., $\boldsymbol{W_0} = \boldsymbol{W} + \boldsymbol{\Delta W}$ where $\boldsymbol{W_0}$ is the true solution. In other words, in the vicinity of the true solution, we can find $\boldsymbol{\Delta W}$, the change in the weight matrix such that for the updated $\boldsymbol{W}$, (17) is satisfied. Therefore

$$\boldsymbol{A}^{-1}\boldsymbol{x} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{\Delta W}\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{\Delta y} \tag{18}$$

where

$$\boldsymbol{\Delta y} = \boldsymbol{\Delta}(\boldsymbol{W}\boldsymbol{x}) = \boldsymbol{\Delta W}\boldsymbol{x}$$

is the change in the output due to the change in the weight matrix. Under the noiseless condition, since $\epsilon \to 0$, we must have

$$\frac{1}{T}\sum_{t=1}^{T}\frac{\partial}{\partial \boldsymbol{A}} D(\boldsymbol{A}^{-1}\boldsymbol{x}^{(t)}) = 0. \tag{19}$$

In other words

$$\frac{1}{T}\sum_{t=1}^{T}\frac{\partial}{\partial \boldsymbol{W}} D(\boldsymbol{y} + \boldsymbol{\Delta y}) = 0. \tag{20}$$

From (11), we can write

$$\frac{\partial}{\partial w_{ij}} D(\boldsymbol{y}) = \begin{cases} (|y_i|-1)\operatorname{sgn}(y_i)x_j, & \text{if } |y_i|>1 \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

Let us denote (21) as

$$\frac{\partial}{\partial \boldsymbol{W}} D(\boldsymbol{y}) = \boldsymbol{E}(\boldsymbol{y}) \tag{22}$$

where $\boldsymbol{E}$ is a matrix whose ijth element is given by (21). From (20) and (22), we write

$$\langle \boldsymbol{E}(\boldsymbol{y} + \boldsymbol{\Delta y}) \rangle = 0 \tag{23}$$

where $\langle \cdot \rangle$ is the sample average. In the proximity of the true solution, we can expand $\boldsymbol{E}(\boldsymbol{y} + \boldsymbol{\Delta y})$ as a Taylor series expansion as

$$\boldsymbol{E}(\boldsymbol{y} + \boldsymbol{\Delta y}) = \boldsymbol{E}(\boldsymbol{y}) + \boldsymbol{\Delta E}(\boldsymbol{y}) + \text{higher order terms.} \tag{24}$$

Again from (21), we can write

$$[\boldsymbol{\Delta E}(\boldsymbol{y})]_{ij} = \begin{cases} \Delta y_i x_j, & \text{if } |y_i| > 1 \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

Note that the directional derivative $[\boldsymbol{\Delta E}(\boldsymbol{y})]_{ij}$ exists for $|y_i| > 1$ and $|y_i| < 1$. The function is continuous at $|y_i| = 1$ but not smooth at that point.

Since we have

$$\boldsymbol{\Delta y} = \boldsymbol{\Delta W x} \tag{26}$$

the (25) can be written as

$$\boldsymbol{\Delta E}(\boldsymbol{y}) = \boldsymbol{\Delta W x x'}. \tag{27}$$

Therefore, from (23), (24), and (27) we get (ignoring the higher order terms)

$$\boldsymbol{\Delta W} \langle \boldsymbol{x x'} \rangle = -\langle \boldsymbol{E}(\boldsymbol{y}) \rangle. \tag{28}$$

After perfect separation, $\boldsymbol{y_0} = \boldsymbol{W_0 x}$, where $\boldsymbol{W_0} = CA^{-1}$. As discussed in Section II, $y_{0i} \in [-1, 1]$, we have $\langle \boldsymbol{y_0 y_0'} \rangle = \boldsymbol{I}$ (the output signals after perfect separation are independent). Therefore

$$\begin{aligned} \langle \boldsymbol{x x'} \rangle &= \boldsymbol{A C^{-1}} \langle \boldsymbol{y_0 y_0'} \rangle (\boldsymbol{A C^{-1}})' \\ &= \boldsymbol{A C}^{-1} (\boldsymbol{A C^{-1}})'. \end{aligned} \tag{29}$$

Therefore, from (28)

$$\begin{aligned} \boldsymbol{\Delta W} &= -\langle \boldsymbol{E}(\boldsymbol{y}) \rangle ((\boldsymbol{A C^{-1}})(\boldsymbol{A C^{-1}})')^{-1} \\ &\quad -\langle \boldsymbol{E}(\boldsymbol{y}) \rangle (\boldsymbol{C A^{-1}})' (\boldsymbol{C A^{-1}}) \\ &= -\langle \boldsymbol{E}(\boldsymbol{y}) \rangle \boldsymbol{W_0'} \boldsymbol{W_0} \\ &= \langle \boldsymbol{E}(\boldsymbol{y}) \rangle (\boldsymbol{W} + \boldsymbol{\Delta W})' (\boldsymbol{W} + \boldsymbol{\Delta W}). \end{aligned} \tag{30}$$

Restoring only the first-order terms we get

$$\boldsymbol{\Delta W} = -\langle \boldsymbol{E}(\boldsymbol{y}) \rangle (\boldsymbol{W'W} + \boldsymbol{\Delta W'W} + \boldsymbol{W'\Delta W}). \tag{31}$$

Considering $\boldsymbol{\Delta W}$ has a form of $\boldsymbol{QW}$, we have

$$\boldsymbol{Q} + \boldsymbol{P}(\boldsymbol{Q} + \boldsymbol{Q'}) = -\boldsymbol{P} \tag{32}$$

where

$$\boldsymbol{P} = \langle \boldsymbol{E}(\boldsymbol{y}) \rangle \boldsymbol{W'} \tag{33}$$

i.e.,

$$P_{ij} = \begin{cases} (|y_i| - 1) \operatorname{sgn}(y_i) y_j, & \text{if } |y_i| > 1 \\ 0, & \text{otherwise.} \end{cases} \tag{34}$$

From the maximum likelihood equations, we can find that any row of the matrix $\boldsymbol{P}$ has nonzero entry only if the corresponding component of the output vector $\boldsymbol{y}$ has an absolute value greater than unity (i.e., $|y_i| > 1$). After perfect separation ($\boldsymbol{W_0} = \boldsymbol{C A^{-1}}$), all output vectors will be contained within a hypercube $H(\boldsymbol{W_0})$ which is given as

$$H(\boldsymbol{W_0}) = \{\boldsymbol{y} | |y_i(t)| < 1, \forall i, \forall t\}. \tag{35}$$

In the case of maximum likelihood estimate, $\boldsymbol{W}$ changes only when some component of $\boldsymbol{y}$ has a value greater than unity, i.e, $\boldsymbol{y}$ is outside $H(\boldsymbol{W_0})$. In other words, geometrically, we can view the maximum likelihood estimator as a minimizer of the error occurred due to the presence of points outside $H(\boldsymbol{W_0})$.

As a special case, if we consider $\boldsymbol{Q}$ to be antisymmetric, i.e., $\boldsymbol{Q} = -\boldsymbol{Q'}$ which implies only a rotation of the hyperbox spanned by the weight matrix $\boldsymbol{W}$ ($H(\boldsymbol{W})$) then

$$\boldsymbol{Q} = -[\boldsymbol{P}]_A \tag{36}$$

where $[\cdot]_A$ indicates the antisymmetric part of the matrix. The hyperbox $H(\boldsymbol{W})$ is defined as

$$H(\boldsymbol{W}) = \{\boldsymbol{y}(t) | |(\boldsymbol{WA})^{-1} \boldsymbol{y}(t)|_i \le 1, \forall i, \forall t\} \tag{37}$$

is the hyperbox consisting of all output vectors $\boldsymbol{y}$. Therefore, the updating rule obtained by the maximum likelihood estimator rotates the hyperbox $H(\boldsymbol{W})$ to match it with $H(\boldsymbol{W_0})$. Note that for antisymmetric $\boldsymbol{Q}$, the maximum likelihood solution is exactly the same as that obtained in [7] and [8]. In [7] and [8], the updating rule has been derived by the natural gradient minimization of the error occurred due to the presence of an outlier.

Similarly if we consider $\boldsymbol{Q}$ to be symmetric, i.e, $\boldsymbol{Q} = \boldsymbol{Q'}$ which implies only shear and scaling of the hyperbox spanned by $\boldsymbol{W}$ ($H(\boldsymbol{W})$) and no rotation of the hyperbox then

$$\boldsymbol{Q} = -[\boldsymbol{P}(\boldsymbol{I} + 2\boldsymbol{P})^{-1}]_S \tag{38}$$

where $[\cdot]_S$ denotes the symmetric part of a matrix. It is also interesting to note that in the case of rotation or shear, the maximum likelihood solution naturally becomes equivalent to the natural gradient descent solution [2]. In the general case, without any assumption of symmetric or antisymmetric $\boldsymbol{Q}$, it is difficult to obtain a close form solution of the maximum likelihood equation (32). The solution may be obtained from the geometric analog of the maximum likelihood estimator. In the next section we present a similar geometric formulation which gives a closed form updating rule of the weight matrix which is implementable in the neural-network framework.

IV. Geometric Formulation of the Learning Rule

Since the original source signals are independently uniformly distributed, ideally the output signals should also be independent and uniformly distributed. Therefore, all the output signals will be contained on/within a hypercube of volume 2^n. In other words

$$|y_i(t)| \le 1, \qquad \text{for all } i \text{ and } t$$

where t is an occurrence of the i^{th} output signal and $\boldsymbol{y} = \boldsymbol{Wx} = \boldsymbol{WAs}$. The unknown mixing matrix $\boldsymbol{A}$ causes a linear

transformation of the original hypercube containing the source signals. The task is therefore to adaptively estimate the inverse of the transformation in order to recover the original hypercube $H(\boldsymbol{W}_0)$ from the transformed hyperbox $H(\boldsymbol{W})$. Any linear transformation of $H(\boldsymbol{W})$ in the n-dimensional space can be thought of as a combination of rotation, shear, and scaling of the hyperbox. Therefore, weights are to be updated in such a way that it causes the effect of rotation, shear, and scaling.

A. Empirical Learning Rule

1) Rotation of the Hyperbox: Let $SO(n)$ be the set of all special orthogonal matrices in n-dimension such that for any $\boldsymbol{B} \in SO(n)$, $\boldsymbol{B}'\boldsymbol{B} = \boldsymbol{I}$. Any $\boldsymbol{B} \in SO(n)$ can be expressed as $\exp(\eta \boldsymbol{Z})$ where $\boldsymbol{Z}$ is an antisymmetric matrix with $||\boldsymbol{Z}|| = 1$ and η is a constant. Therefore, if the updating of $\boldsymbol{W}$ is such that

$$d\boldsymbol{W} = (\exp(\eta \boldsymbol{Z}) - \boldsymbol{I})\boldsymbol{W} \tag{39}$$

then $\boldsymbol{W}'\boldsymbol{W}$ (i.e., $\boldsymbol{W}\boldsymbol{W}'$) remains unchanged. Therefore, the angle between any ith and jth hyperplanes of the n-dimensional hyperbox $H(\boldsymbol{W})$ remains constant. In other words, the transformation effectively rotates the hyperbox. A first-order approximation of the (39) is given as

$$d\boldsymbol{W} = \eta \boldsymbol{Z}\boldsymbol{W} \tag{40}$$

which is also explicitly used in Cardoso and Laheld [10]. A second-order approximation is given as

$$d\boldsymbol{W} = \left(\eta \boldsymbol{Z} + \frac{\eta^2}{2}\boldsymbol{Z}^2 - O(\eta^3)\right)\boldsymbol{W}. \tag{41}$$

2) Shear of the Hyperbox: Any shearing transformation can be represented as an upper triangular or lower triangular matrix. A pure shear can be represented as a transformation of $\boldsymbol{W}$ given as

$$\boldsymbol{W} \Rightarrow \boldsymbol{W} + \boldsymbol{S}_h\boldsymbol{W} \tag{42}$$

where $\boldsymbol{S}_h$ is a symmetric matrix with zero diagonal entries. If $\boldsymbol{S}_h$ has nonzero diagonal entries then there will be different scaling of $\boldsymbol{W}$ in different directions. Since the antisymmetric component of a symmetric matrix is zero, there is no rotation of the matrix. Again, any linear transformation is a composition of rotation, shear, and scaling (including reflection). The scaling is represented by the diagonal entries of the transformation matrix. A more general way of representing the shear along with the scaling is

$$d\boldsymbol{W} = (\exp(\eta \boldsymbol{V}) - \boldsymbol{I})\boldsymbol{W} \tag{43}$$

where $\boldsymbol{V}$ is a symmetric matrix with $||\boldsymbol{V}|| = 1$ and η is a constant. First- and second-order approximations of (43) are given as

$$d\boldsymbol{W} = \eta \boldsymbol{V}\boldsymbol{W} \tag{44}$$

and

$$d\boldsymbol{W} = \left(\eta \boldsymbol{V} + \frac{\eta^2}{2}\boldsymbol{V}^2\right)\boldsymbol{W}. \tag{45}$$

3) Scaling of the Hyperbox: Any scaling of the hyperbox $H(\boldsymbol{W})$ can be expressed as

$$d\boldsymbol{W} = (\exp(\eta \boldsymbol{\Lambda}) - \boldsymbol{I})\boldsymbol{W} \tag{46}$$

where $\boldsymbol{\Lambda} = \operatorname{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ is a diagonal matrix. A first-order approximation is simply

$$d\boldsymbol{W} = \eta \boldsymbol{\Lambda}\boldsymbol{W}. \tag{47}$$

$\diamondsuit$

Any change $d\boldsymbol{W} = \boldsymbol{Q}\boldsymbol{W}$ in the weight matrix $\boldsymbol{W}$ can be orthogonally decomposed as

$$\boldsymbol{Q}\boldsymbol{W} = [\boldsymbol{Q}]_A\boldsymbol{W} + \operatorname{diag}\{\boldsymbol{Q}\}\boldsymbol{W} + ([\boldsymbol{Q}]_S - \operatorname{diag}\{\boldsymbol{Q}\})\boldsymbol{W} \tag{48}$$

where $[\cdot]_A$ denotes the antisymmetric part, $[\cdot]_S$ denotes the symmetric part, and $\operatorname{diag}\{\cdot\}$ is the diagonal of a matrix. The first part indicates a rotation, the second part is a scaling, and the third part gives a pure shear of the hyperbox spanned by the weight matrix $\boldsymbol{W}$. A transformation can be performed by first rotating the hyperbox, then scaling and shearing it instead of directly changing all parameters together. It has been shown in Section VI that the orthogonal decomposition of the transformations provide much better performance as compared to a direct transformation of the weight matrix. The hyperbox spanned by $\boldsymbol{W}$ is to be transformed in such a way that the error occurred due to the presence of outliers is minimized. Here the word "outlier" means any sample vector falling outside the hyperbox spanned by $\boldsymbol{W}_0$ i.e., $H(\boldsymbol{W}_0)$ (35). In the noiseless condition, there should be no outlier after perfect separation. In general, outliers are the ones which have just very big or extreme values. In the noisy condition, in order to analyze this, we use a mixture model where outliers belong to a different distribution. However, in practice, a very large signal vector may influence the estimator badly.

B. Error Measure

The Kullback–Leibler divergence measure $\mathcal{D}[p : q]$ does not exist when the two distributions p and q are not absolutely continuous. One such typical situation arise in the case of uniform distribution. Since the probability distribution of the source signals are known, we can use variational or Hellinger distance to measure the error between output distribution $p(\boldsymbol{y}; \boldsymbol{W})$ and the original source distribution $p(\boldsymbol{y}; \boldsymbol{A}^{-1})$. The error in terms of variational distance is given as

$$\begin{aligned} E &= \mathcal{D}[p(\boldsymbol{y}; \boldsymbol{W}) : p(\boldsymbol{y}; \boldsymbol{A}^{-1})] \\ &= \int |p(\boldsymbol{y}; \boldsymbol{W}) - p(\boldsymbol{y}; \boldsymbol{A}^{-1})| d\boldsymbol{y}. \end{aligned} \tag{49}$$

Equivalently, the error can also be expressed in terms of the Hellinger distance as

$$E = \int (\sqrt{p(\boldsymbol{y}; \boldsymbol{W})} - \sqrt{p(\boldsymbol{y}; \boldsymbol{A}^{-1})})^2 d\boldsymbol{y}. \tag{50}$$

We express the error measure from the observed samples in an analogous way to the variational distance. The mixing matrix $\boldsymbol{A}$ defines a hyperbox in the input signal space of $\boldsymbol{x}$, and $\boldsymbol{W}$ defines the estimated hyperbox. All the input vectors

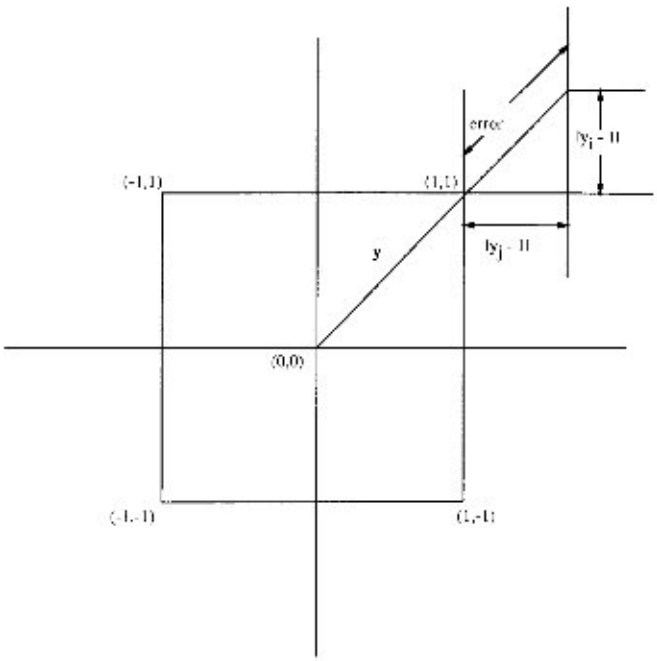

Fig. 1. A two-dimensional view of the hypercube (i-j plane) in the noiseless condition. The outlier $\boldsymbol{y}$ has two components outside the hypercube.

are contained on/within the hyperbox defined by $\boldsymbol{A}$. If the estimation of $\boldsymbol{W}$ is perfect, i.e., $\boldsymbol{W} = \boldsymbol{A}^{-1}$ then all the input signals are contained on/within the estimated hyperbox. In other words, all the output signals defined by $\boldsymbol{y} = \boldsymbol{WAs}$ will be contained in a hypercube such that

$$|y_{i_p}| \leq 1, \quad \forall p, i_p \in [1, n]. \tag{51}$$

At any instant t, if $\boldsymbol{x}(t)$ is not contained within/on the hyperbox defined by $\boldsymbol{W}^{-1}$ then the corresponding output $\boldsymbol{y}(t)$ will be outside the hypercube. Let us call such an instance of signal vector as an outlier. In analogy to the variational distance, the total error can be measured in the output space as

$$e = \sum_i (|y_i| - |\tilde{y}_i|) \tag{52}$$

where $\tilde{y}_i = (\boldsymbol{W_0 x})_i$.

Since $\boldsymbol{A}$ is unknown and needs to be adaptively estimated, $\tilde{\boldsymbol{y}}$ is unknown. We therefore consider that for an outlier $\boldsymbol{y}$, $\tilde{\boldsymbol{y}}$ is the point on the hyperbox closest to $\boldsymbol{y}$. In other words, at every instant, we assume that the estimation is close to the perfect. Thus the error in the output space is defined as

$$e = \sum_{i, |y_i| > 1} (|y_i| - 1). \tag{53}$$

The error e is geometrically represented in Fig. 1. The average error over all instances of the output is given as

$$\langle e \rangle = \frac{1}{T} \sum_t \sum_{i, |y_i(t)| > 1} (|y_i(t)| - 1) \tag{54}$$

where T is the number of observations.

Note that, the average error $\langle e \rangle$ is not the same as the variational distance since $\tilde{\boldsymbol{y}}$ is not necessarily the same as $\boldsymbol{A}^{-1}\boldsymbol{x}$. However, $\langle e \rangle$ provides an error measure (training error that is to be minimized based on the observed samples) analogous to the variational distance. With such an error measure, the estimated hyperbox is to be rotated and sheared only when there is an outlier and in such a way that the outlier just touches the closest bounding hyperplane of the hyperbox.

C. Formulation of the Learning Rule

The estimated hyperbox is to be adaptively transformed in such a way that the error $\langle e \rangle$ is minimized. According to the natural gradient descent algorithm [2], we can write the updating rule of $\boldsymbol{W}$ in terms of the instantaneous variables as

$$d\boldsymbol{W} \propto -\frac{\partial e}{\partial \boldsymbol{W}} \boldsymbol{W}'\boldsymbol{W}. \tag{55}$$

By definition, $\partial e/\partial \boldsymbol{W}$ is the rate of change of e with respect to $\boldsymbol{W}$, i.e.,

$$\Delta e = e(\boldsymbol{W} + d\boldsymbol{W}) - e(\boldsymbol{W}) = \mathrm{Tr}\left(\frac{\partial e}{\partial \boldsymbol{W}} d\boldsymbol{W}\right). \tag{56}$$

$\boldsymbol{W}$ can be updated in order to minimize e so that all n^2 variables in $\boldsymbol{W}$ are changed simultaneously. In other words, the shape of the hyperbox is transformed and also it is rotated. The change in shape and rotation depends on the local gradients.

Evaluating the partial derivative of e (considering $d\boldsymbol{W}$ has n^2 free parameters), we get

$$\frac{\partial e}{\partial W_{ij}} = \begin{cases} x_j \operatorname{sgn}(y_i), & \text{for } |y_i| > 1 \\ 0, & \text{otherwise.} \end{cases} \tag{57}$$

Note that, the directional derivative exists for $|y_i| > 1$ and $|y_i| < 1$ for each i. From (57), we have

$$\left(\frac{\partial e}{\partial \boldsymbol{W}} \boldsymbol{W}'\right)_{ij} = \begin{cases} y_j \operatorname{sgn}(y_i), & \text{for } |y_i| > 1 \\ 0, & \text{otherwise.} \end{cases} \tag{58}$$

Instead of transforming $\boldsymbol{W}$ directly based on the local gradients considering n^2 free variables, the transformation can be decomposed into rotation, scaling, and shear [(48)], and each of them can be performed separately.

In the case of rotation, $\boldsymbol{W}$ can change only in the direction of $\boldsymbol{ZW}$ [(40)] where $\boldsymbol{Z}$ is an antisymmetric matrix having only $n(n-1)/2$ free parameters. Therefore, the natural gradient descent algorithm for rotating the hyperbox $H(\boldsymbol{W})$ can be obtained by

$$d\boldsymbol{W} \propto -\left[\frac{\partial e}{\partial \boldsymbol{W}} \boldsymbol{W}'\right]_A \boldsymbol{W} \tag{59}$$

where $[\cdot]_A$ is the antisymmetric part. Similarly restoring the diagonal and the symmetric parts [(48)] we obtain the scaling and pure shear of the hyperbox. Therefore the natural gradient rules for scaling and shear can be, respectively, obtained as

$$d\boldsymbol{W} \propto -\mathrm{diag}\left\{\frac{\partial e}{\partial \boldsymbol{W}} \boldsymbol{W}'\right\} \boldsymbol{W} \tag{60}$$

and

$$d\boldsymbol{W} \propto -\left[\frac{\partial e}{\partial \boldsymbol{W}} \boldsymbol{W}'\right]_S \boldsymbol{W}. \tag{61}$$

From (59)–(61), we can obtain the separate rules for rotation, scaling, and shear. For rotation, the updating rule is given as

$$\Delta W = \eta_Z Z W \tag{62}$$

where

$$Z = yg(y)' - g(y)y' \tag{63}$$

and η_Z is the learning rate parameter for rotation. The nonlinear function $g(y) = [g(y_1), g(y_2), \cdots, g(y_n)]'$ is given as

$$g(y_i) = \begin{cases} \operatorname{sgn}(y_i), & \text{if } |y_i| > 1 \\ 0, & \text{otherwise.} \end{cases} \tag{64}$$

Similarly for scaling the updating rule is

$$\Delta W = \eta_\Lambda \Lambda W \tag{65}$$

where

$$\Lambda = -\operatorname{diag}\{g(y)y'\} \tag{66}$$

and η_Λ is the learning rate for scaling. The updating rule for shear is given as

$$\Delta W = \eta_V V W \tag{67}$$

where

$$V = -\left(\tfrac{1}{2}\{yg(y)' - g(y)y'\} - \operatorname{diag}\{g(y)y'\}\right) \tag{68}$$

and η_V is the learning rate for shear.

Considering the ensemble of outliers, we can obtain a batch learning by minimizing $\langle c \rangle$ with natural gradient descent algorithm. Let us represent a set of p outliers by $n \times p$ matrix Y such that

$$Y = [y^{(1)}, y^{(2)}, \cdots, y^{(p)}] \tag{69}$$

where $Y_{ij} = y_i^{(j)}$ is the jth component of the ith outlier. In that case, the batch learning rules can be obtained in the same way as in the case of on-line learning. For rotation, the updating matrix Z is given as

$$Z = YG(Y)' - G(Y)Y' \tag{70}$$

where $[G(Y)]_{ij} = g(y_i^{(j)})$, $g(\cdot)$ is given by (64). Similarly, the updating rules for scaling and shear, i.e., the updating matrices V and Λ can be obtained in the batch mode by replacing $g(y)$ with $G(Y)$ and y with Y.

D. Learning Rate

The learning rate parameter (η_Z, η_Λ, or η_V) determines the amount of transformation (rotation, scaling, and/or shear) of the hyperbox to be performed. The hyperbox is transformed in such a way that the error due to the presence of the outliers is minimized.

For each outlier, a correction vector is defined as

$$c = Qy \tag{71}$$

where $Q = Z$, Λ, or V depending on whether it is a rotation, scaling or a shear. The corresponding correction vector is c_Z or c_Λ, or c_V, respectively. The change in the output of each outlier due to the transformation of the hyperbox is given by

$$dy = \eta c \tag{72}$$

where $\eta = \eta_Z$ (the parameter for rotation) or η_Λ (the parameter for scaling) or η_V (the parameter for shear).

In the batch mode, we define a matrix C whose columns represent the correction vectors corresponding to the outliers. Thus

$$C = QY \tag{73}$$

where $C = [c^{(1)}, c^{(2)}, \cdots, c^{(p)}]$. The change in Y is then given as

$$dY = \eta C. \tag{74}$$

The hyperbox is to be transformed in such a way that in each transformation (rotation, scaling or shear), each outlier becomes as close to the hyperbox as possible. Ideally all the outliers should either fall on the hyperplane boundaries or should be contained within the hyperbox after the transformation. In the case of rotation, scaling, and shear performed separately no single operation may achieve the desired effect. Therefore, each operation is performed in such a way that the residual error due to the presence of the outliers is minimized. Here the residual error indicates the minimum change in the signal components necessary such that the signal vector is contained within the hyperbox spanned by W_0, i.e., $H(W_0)$. We define the desired change in the output of the outliers as (L_1-norm)

$$\begin{aligned} \Delta Y_{ij} &= \Delta y_i^{(j)} \\ &= \begin{cases} -(|Y_{ij}| - 1)\operatorname{sgn}(Y_{ij}), & \text{if } |Y_{ij}| > 1 \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{75}$$

In other words, ΔY provides the componentwise error occurred due to the presence of all outliers. The learning rate parameter is to be chosen in such a way that the cumulative effect of componentwise error is reduced optimally. Therefore, $||\Delta Y - dY||_F^2$ is minimized with respect to η where $||\cdot||_F$ is the Frobenius norm. At the minima,

$$\frac{d}{d\eta}||\Delta Y - \eta C||_F^2 = 0. \tag{76}$$

Solving for η, we get

$$\eta = \frac{\operatorname{Tr}(\Delta Y' C)}{||C||_F^2}. \tag{77}$$

Therefore,

$$\begin{aligned} \eta_Z &= \frac{\operatorname{Tr}(\Delta Y' Z Y)}{||ZY||_F^2} \\ \eta_\Lambda &= \frac{\operatorname{Tr}(\Delta Y' \Lambda Y)}{||\Lambda Y||_F^2} \\ \eta_V &= \frac{\operatorname{Tr}(\Delta Y' V Y)}{||VY||_F^2}. \end{aligned} \tag{78}$$

In other words, η (η_Z or η_Λ or η_V) is the resultant normalized dot product (i.e., the angle) between the correction vectors and the vectors representing the desired changes in the output corresponding to all the outliers.

E. Reduction of Free Parameters

The geometric description described in the previous sections lead to a weight updating rule analogous to that obtained by the maximum likelihood estimator. It is interesting to note that the maximum likelihood estimator naturally provides the natural gradient solution. However, in the maximum likelihood solution (32), each entry in the ith row of the weight updating matrix $\boldsymbol{P}$ is scaled by $(|y_i| - 1)$. This is because we considered a form of Euclidian distance of an outlier from the hypercube to obtain the density function (11). In the geometric interpretation, however, we considered the city block distance to measure the error. An analogous form of scaling effect in this learning rule is embedded by taking into account of the learning rate parameter. The learning rate parameter [η in (77)] takes the account of the cumulative effect of $(|y_i| - 1)$ of all components of all outliers into a scaler variable.

During rotation, the shape and volume of the hyperbox does not change. The shape and volume changes during scaling and shear. We start with some scaled identity weight matrix which spans a hypercube. The hyperbox is then transformed in such a way that maximum compactness of the hyperbox is preserved. In order to make the distortion as minimum as possible, the hyperbox is first rotated in order to minimize the error due to the presence of an outlier. After the optimal amount of rotation, the hyperbox is sheared and scaled based on the residual error due to the same observed samples.

The weight matrix is updated in such a way that $|\det(\boldsymbol{W})|$ remains constant in each transformation. This does not deteriorate the equivariance property of the algorithm. We consider the empirical form of weight updating as $\boldsymbol{W} = \boldsymbol{W} + \eta \boldsymbol{Q}\boldsymbol{W}$. Therefore

$$|\det(\boldsymbol{W})| = |\det(\boldsymbol{W})||\det(\boldsymbol{I} + \eta\boldsymbol{Q})|. \tag{79}$$

In order to keep $|\det(\boldsymbol{W})|$ constant, each entry in $\boldsymbol{W}$ is scaled by a normalizing constant such that new updating rule is

$$\boldsymbol{W} = (\boldsymbol{W} + \eta\boldsymbol{Z}\boldsymbol{W})/|\det(\boldsymbol{I} + \eta\boldsymbol{Q})|^{1/n}. \tag{80}$$

In the case of rotation, if we consider (39) then $|\det(\boldsymbol{W})|$ always remains constant. If we use the first-order approximation of the rotation matrix then the normalizing factor can be approximated as

$$1/|\det(\boldsymbol{I} - \eta_Z\boldsymbol{Z})|^{1/n} = 1 - \eta_Z^2\frac{\|\boldsymbol{Z}\|_F^2}{2n} \tag{81}$$

considering η_Z to be small. Similarly in the case of scaling, since we have only diagonal weight updating matrix and

$$1/|\det(\boldsymbol{I} + \eta_\Lambda\boldsymbol{\Lambda})|^{1/n} = 1 - \eta_\Lambda\frac{\mathrm{Tr}(\boldsymbol{\Lambda})}{n}. \tag{82}$$

In the case of pure shear, the normalizing constant can be approximated as

$$1/|\det(\boldsymbol{I} + \eta_V\boldsymbol{V})|^{1/n} = 1 + \eta_V^2\frac{\|\boldsymbol{V}\|_F^2}{2n}. \tag{83}$$

F. Separation in the Presence of Noise

In the presence of noise, the source signals are deviated from the true uniform distribution. As described in Section III, the noisy distribution can be approximated by (10) where ϵ depends on the noise amplitude. The approximation can be assumed to be valid for small noise amplitude and therefore for small ϵ. For large noise amplitude, the Fisher information is not only concentrated at the boundary of the hypercube but also takes nonzero value inside the hypercube. Therefore, the approximation by (10) is no more valid for large ϵ which considers nonzero Fisher information only outside the hypercube. Under the assumption of small noise amplitude, a maximum likelihood solution can be written as

$$\frac{\partial l(\boldsymbol{W})}{\partial \boldsymbol{W}} = 0 \tag{84}$$

which gives [from (17)]

$$(\boldsymbol{W} + \boldsymbol{\Delta W})' + \frac{1}{T\epsilon^2}\sum_t \frac{\partial}{\partial \boldsymbol{W}} D(\boldsymbol{y} + \boldsymbol{\Delta y}) = 0 \tag{85}$$

where we consider $\boldsymbol{W} + \boldsymbol{\Delta W} = \boldsymbol{C}\boldsymbol{A}^{-1}$ is the desired solution. Therefore, in the vicinity of the desired solution, i.e., for small $|\boldsymbol{\Delta W}|$, and for small amount of noise, i.e., small ϵ, we have [from (22) and (27)]

$$\boldsymbol{W}' + \frac{1}{\epsilon^2}[\langle E(\boldsymbol{y})\rangle + \boldsymbol{\Delta W}\langle \boldsymbol{x}\boldsymbol{x}'\rangle] = 0. \tag{86}$$

Considering $\langle \boldsymbol{x}\boldsymbol{x}'\rangle = ((\boldsymbol{W} + \boldsymbol{\Delta W})'(\boldsymbol{W} + \boldsymbol{\Delta W}))^{-1}$ [(29)], we get

$$\boldsymbol{\Delta W} = -(\langle E(\boldsymbol{y})\rangle + \epsilon^2\boldsymbol{W}')(\boldsymbol{W} + \boldsymbol{\Delta W})'(\boldsymbol{W} + \boldsymbol{\Delta W}). \tag{87}$$

Restoring only the first-order terms of $\boldsymbol{\Delta W}$ and considering $\boldsymbol{\Delta W}$ has a form of $\boldsymbol{Q}\boldsymbol{W}$, we get

$$\boldsymbol{Q} - (\boldsymbol{P} + \epsilon^2\boldsymbol{W}\boldsymbol{W}')(\boldsymbol{Q} + \boldsymbol{Q}') = -(\boldsymbol{P} + \epsilon^2\boldsymbol{W}\boldsymbol{W}') \tag{88}$$

where $\boldsymbol{P}$ is the same as in (34), i.e., $\boldsymbol{P} = \langle \boldsymbol{E}(\boldsymbol{y})\rangle\boldsymbol{W}'$.

Therefore, the maximum likelihood solution in the noisy case is exactly the same as in the noiseless condition when $\epsilon \to 0$. The parameter ϵ, dependent on the noise amplitude, controls the weight updating matrix $\boldsymbol{Q}$. It behaves like a feedback parameter controlling the increase or decrease of the weights.

In analogy to the maximum likelihood solution, the geometric transformations of the weight matrix are accordingly modified. Since $\epsilon^2\boldsymbol{W}\boldsymbol{W}'$ has no antisymmetric component, the rotation is not affected due to the presence of noise. This is physically interpretable because in the presence of noise also the distribution of the signals remain symmetric. Since we considered that additive noise components to the signal components have the same amplitude, the shape of the resultant distribution actually generates a dilated hypercube [7]. The scaling and shear operations are accordingly modified based on the controlling factor $\epsilon^2\boldsymbol{W}\boldsymbol{W}'$.

For scaling and shear, the modified weight updating rules can be obtained from (65) and (67), respectively, by replacing $\boldsymbol{g}(\boldsymbol{y})\boldsymbol{y}'$ with $\boldsymbol{g}(\boldsymbol{y})\boldsymbol{y}' + \epsilon^2\boldsymbol{W}\boldsymbol{W}'$. Therefore for scaling we have a learning rule

$$\boldsymbol{W} = \boldsymbol{W} + \eta_\Lambda(\boldsymbol{\Lambda} - \epsilon^2\,\mathrm{diag}\{\boldsymbol{W}\boldsymbol{W}'\})\boldsymbol{W}. \tag{89}$$

Similarly, for shear the modified learning rule can be written as

$$\boldsymbol{W} = \boldsymbol{W} + \eta_V(\boldsymbol{V} - \epsilon^2(\boldsymbol{W}\boldsymbol{W}' - \text{diag}\{\boldsymbol{W}\boldsymbol{W}'\}))\boldsymbol{W}. \quad (90)$$

The analogous expressions can be used for batch mode learning where the matrix $\boldsymbol{G}(\boldsymbol{Y})\boldsymbol{Y}' + \epsilon^2\boldsymbol{W}\boldsymbol{W}'$ is to be used instead of $\boldsymbol{G}(\boldsymbol{Y})\boldsymbol{Y}'$. The learning rate parameters η_Λ and η_V can be obtained from (77) by replacing $\boldsymbol{\Lambda}$ with $(\boldsymbol{\Lambda} - \epsilon^2\,\text{diag}\{\boldsymbol{W}\boldsymbol{W}'\})$ and $\boldsymbol{V}$ with $(\boldsymbol{V} - \epsilon^2(\boldsymbol{W}\boldsymbol{W}' - \text{diag}\{\boldsymbol{W}\boldsymbol{W}'\}))$, respectively.

V. Overall Algorithm

The updating rules are described for on-line and batch-mode in Section IV. In this section, we summarize the overall algorithm for a given batch size and the ϵ. In the batch-mode updating, a number of samples are to be observed at a time. Instead of observing a large set of samples at a time, the algorithm can be implemented in a semibatch-mode where the weights are updated in each iteration after observing each sample like an on-line algorithm but the effect of batch-mode updating is incorporated.

A shift register is used to store a previously defined number (say p) of observed samples. Whenever a new sample appears, it is checked whether the sample falls outside the hyperbox spanned by $\boldsymbol{W}$ (i.e., $H(\boldsymbol{W})$) or not, i.e., $|y_i(t)| > 1$ for any i. If the observed sample happens to be an outlier then it is stored in the register and the oldest sample stored in the register is taken out. The shift register is implemented as a first in first out (FIFO). Therefore, the register stores the last p samples which appeared as outliers.

At any instance therefore, the effective batch size is not necessarily fixed at p. The effective batch size, in this context, means the number of observed samples which at a time, effectively cause geometric transformation of the hyperbox. The effective batch size does not remain constant because some of the previous outliers stored in the register may be contained within the hyperbox $H(\boldsymbol{W})$ at the present instance. Therefore, these samples do not cause any geometric transformation of the hyperbox.

Let us now algorithmically describe the overall separation algorithm:

Step 1: Fix p, the size of the register
Fix ϵ, the noise parameter
Define $\boldsymbol{X}$: $n \times p$ matrix to store the observed samples
Define $\boldsymbol{Y}$: $n \times p$ matrix to store the output samples
Initialize $\boldsymbol{X}$ and $\boldsymbol{Y}$ to zeros
Define

$$\boldsymbol{R} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ & & \cdots & & \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

a $p \times p$ matrix
Define $\boldsymbol{a} = [100\cdots0]'$: a $p \times 1$ vector
Initialize $t = 1$. Initialize $\boldsymbol{W} = k\boldsymbol{I}$ where $k > 0$ is any constant, such that such that the absolute values of most of the output signal components are greater than unity.

Step 2: Observe the present sample $\boldsymbol{x}$ at instant t
Compute $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$

If $|y_i| > 1$ for any i then
begin

$$\boldsymbol{X} = \boldsymbol{X}\boldsymbol{R} + \boldsymbol{x}\boldsymbol{a}'$$

/* makes a right shift in the input register $\boldsymbol{X}$ and adds $\boldsymbol{x}$ to the register */
Set **transform** = rotation;
/* sets rotation as the first transformation operation */
end
else Goto *Step 5*.

Step 3: Compute $\boldsymbol{Y} = \boldsymbol{W}\boldsymbol{X}$
Compute $\boldsymbol{G}(\boldsymbol{Y})$: an $n \times p$ matrix

$$[\boldsymbol{G}(\boldsymbol{Y})]_{ij} = \begin{cases} \text{sgn}(Y_{ij}), & \text{if } |Y_{ij}| > 1 \\ 0, & \text{otherwise} \end{cases} \quad (91)$$

and $\boldsymbol{\Delta Y}$: an $n \times p$ matrix;

$$\Delta Y_{ij} = \begin{cases} -(|Y_{ij}| - 1)\ \text{sgn}(Y_{ij}), & \text{if } |Y_{ij}| > 1 \\ 0, & \text{otherwise} \end{cases} \quad (92)$$

Step 4: Case of:

transform = rotation :: Compute

$$\boldsymbol{Z} = \boldsymbol{Y}\boldsymbol{G}(\boldsymbol{Y})' - \boldsymbol{G}(\boldsymbol{Y})\boldsymbol{Y}'$$
$$\eta_Z = \frac{\text{Tr}(\boldsymbol{\Delta Y}'\boldsymbol{Z}\boldsymbol{Y})}{||\boldsymbol{Z}\boldsymbol{Y}||_F^2}$$
$$\boldsymbol{W} = \left(1 - \eta_Z^2\frac{||\boldsymbol{Z}_F|^2}{2n}\right)(\boldsymbol{I} + \eta_Z\boldsymbol{Z})\boldsymbol{W} \quad (93)$$

Set **transform** = shear;
goto *Step 3*.

transform = shear :: Compute

$$\boldsymbol{V} = -\frac{1}{2}(\boldsymbol{Y}\boldsymbol{G}(\boldsymbol{Y})' + \boldsymbol{G}(\boldsymbol{Y})\boldsymbol{Y}' + \epsilon^2\boldsymbol{W}\boldsymbol{W}') + \text{diag}\{\boldsymbol{G}(\boldsymbol{Y})\boldsymbol{Y}' + \epsilon^2\boldsymbol{W}\boldsymbol{W}'\}$$
$$\eta_V = \frac{\text{Tr}(\boldsymbol{\Delta Y}'\boldsymbol{V}\boldsymbol{Y})}{||\boldsymbol{V}\boldsymbol{Y}||_F^2}$$
$$\boldsymbol{W} = \left(1 + \eta_V^2\frac{||\boldsymbol{V}||_F^2}{2n}\right)(\boldsymbol{I} + \eta_V\boldsymbol{V})\boldsymbol{W} \quad (94)$$

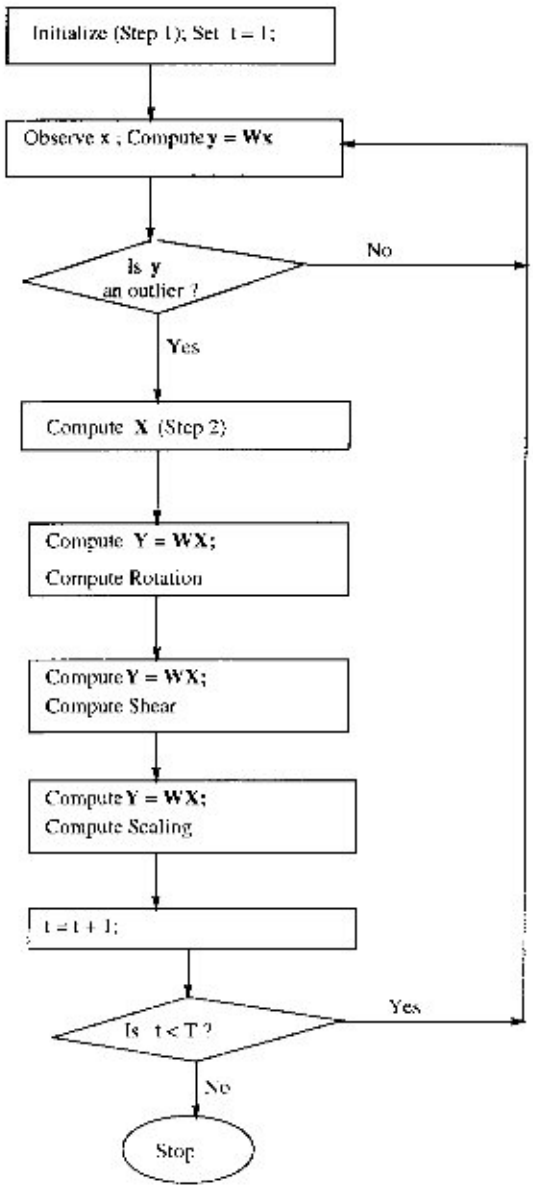

Fig. 2. A schematic block diagram sketching the algorithm. The "step 1" and "step 2" indicate the operations performed as described in the algorithm.

Set **transform** = scaling;
goto *Step 3*.

transform = scaling :: Compute

$$\Lambda = -\operatorname{diag}\{\boldsymbol{G}(\boldsymbol{Y})\boldsymbol{Y}' + \epsilon^2 \boldsymbol{W}\boldsymbol{W}'\}$$
$$\eta_\Lambda = \frac{\operatorname{Tr}(\boldsymbol{\Delta Y}'\boldsymbol{\Lambda Y})}{||\boldsymbol{\Lambda Y}||_F^2}$$
$$\boldsymbol{W} = \left(1 - \eta_\Lambda \frac{\operatorname{Tr}(\boldsymbol{\Lambda})}{n}\right)(\boldsymbol{I} + \eta_\Lambda \boldsymbol{\Lambda})\boldsymbol{W} \tag{95}$$

Step 5: $t = t + 1$;

If $t < T$ goto *Step 2*
else return.

Fig. 2 shows a block diagram schematically sketching the algorithm.

VI. Experimental Results

The effectiveness of the proposed method is demonstrated on the mixtures of five randomly generated source signals; i.e.,

$$\boldsymbol{s}(t) = [N_1(t), N_2(t), N_3(t), N_4(t), N_5(t)]'$$

where each $N_i(t)$ is uniformly distributed in $[-1,1]$.

The performance index is measured by (as proposed in [6] and [24])

$$\text{index} = \sum_{i=1}^{n}\left(\sum_{j=1}^{n}\frac{|C_{ij}|}{\max_k |C_{ik}|} - 1\right) + \sum_{j=1}^{n}\left(\sum_{i=1}^{n}\frac{|C_{ij}|}{\max_k |C_{kj}|} - 1\right) \tag{96}$$

where $\boldsymbol{C} = [C_{ij}] = \boldsymbol{WA}$.

During updating $\boldsymbol{W}$, the hyperbox is first rotated in order to minimize the error occurred due to the presence of an outlier. After the optimal rotation, the hyperbox is sheared and scaled based on the residual error due to the presence of the same outlier. It is to be noted here that no significant change in the performance of the algorithm has been observed if the sequence of the shear and scaling operations are interchanged.

Fig. 3(a) demonstrates the effectiveness of the algorithm on five randomly generated uniformly distributed signals under the noiseless condition with a batch size of $p = 50$. The mixing matrix is randomly generated with each entry $A_{ij} \in [-100\,100]$. The initial $\boldsymbol{W}$ is selected as $\boldsymbol{W} = k\boldsymbol{I}$ where k is some constant. The value of k should be such that the absolute values of most of the signal components are greater than unity. This is necessary because the algorithm is insensitive to the output signal vectors which are contained within a hypercube of unit dimension. In this experiment we have chosen $k = 0.05$. However, we experimented with larger values of k which provide equally good results. In order to test the equivariance property of the algorithm, we considered a general form of $\boldsymbol{\Lambda} = k_1\boldsymbol{M}$ where each entry $M_{ij} \in [-1,1]$ is a uniformly distributed number. The initial $\boldsymbol{W}$ is selected as $\boldsymbol{W} = k_2\boldsymbol{I}$. It has been found that the performance of the algorithm is insensitive to individual k_1 or k_2 so long as $k_1 k_2$ is constant. Given any unknown mixing matrix, k for $\boldsymbol{W} = k\boldsymbol{I}$ can be chosen sufficiently large. This can also be performed by observing a set of initial output samples for some k. Therefore, the algorithm is equivariant in a limited sense with respect to choice of k, so long as the initial absolute values of most of the signal components are greater than unity. However, in general sense, the algorithm is not affine equivariant.

Instead of orthogonal decomposition of the transformations, $\boldsymbol{W}$ can be updated by direct natural gradient descent of the geometric error (Section IV). We updated $\boldsymbol{W}$ by directly minimizing the error where in each iteration $|\det(\boldsymbol{W})|$ is kept constant. However, it has been found that the algorithm does not converge at all with the direct gradient minimization of the error as shown in Fig. 3(b).

This is due to the fact that the decomposition of the transformation into rotation, shear and scaling cause minimum amount of change in the shape of the hyperbox. Initially, we

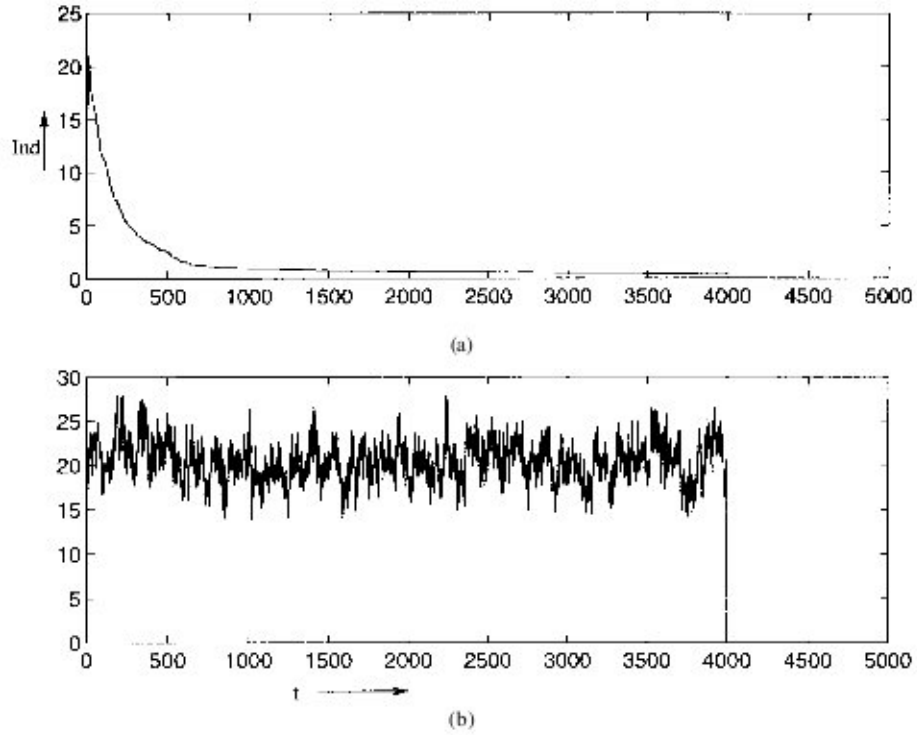

Fig. 3. The performance index is denoted by "Ind" and the number of observations by "t." (a) The performance of the proposed method with orthogonal decomposition of the transformations. The hyperbox is first rotated, then sheared, and finally scaled. (b) The performance of the algorithm by direct gradient minimization.

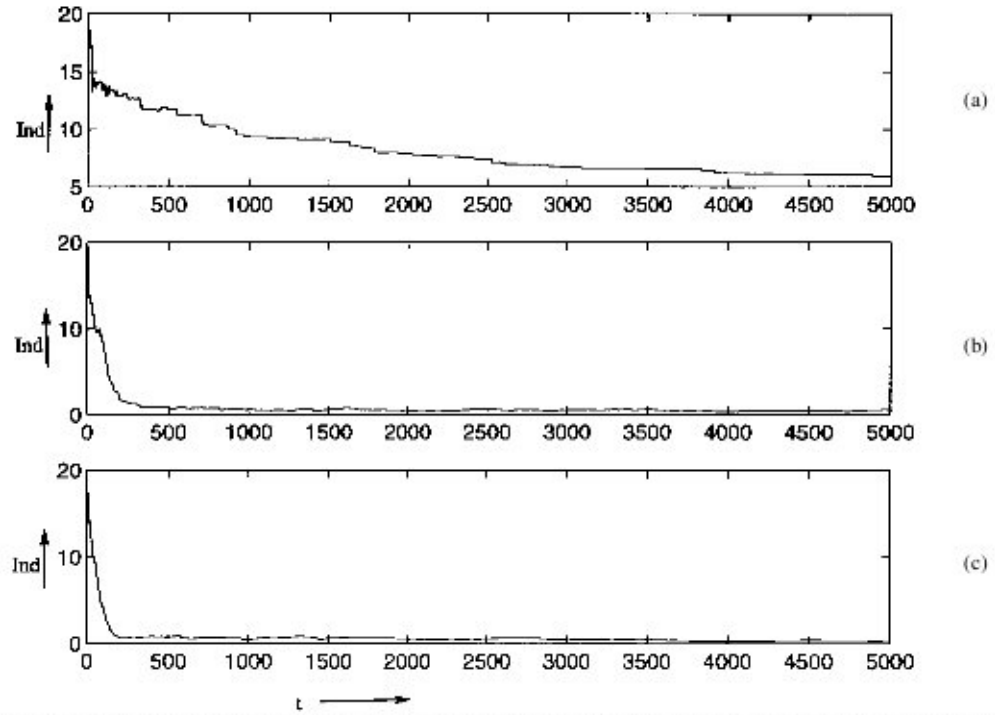

Fig. 4. The performance of the algorithm with different batch sizes of (a) $p = 1$, (b) $p = 10$, and (c) $p = 50$. The performance index is denoted by "Ind" and the number of observations by "t."

consider $\boldsymbol{W}$ as a scaled identity matrix which generates a hypercube. The minimum change in the shape in each iteration preserves the maximum compactness of the hyperbox. The direct gradient minimization, on the other hand, does not restrict the shape of the hyperbox toward minimum distortion.

The effect of batch size on the performance of the algorithm is also tested. Different batch size (Section IV) with $p = 1$, 10, and 50 are used on the same sequence of signal vectors. The mixing matrix and the initial $\boldsymbol{W}$ are considered to be the same in all three cases. The results are shown in Fig. 4(a)–(c),

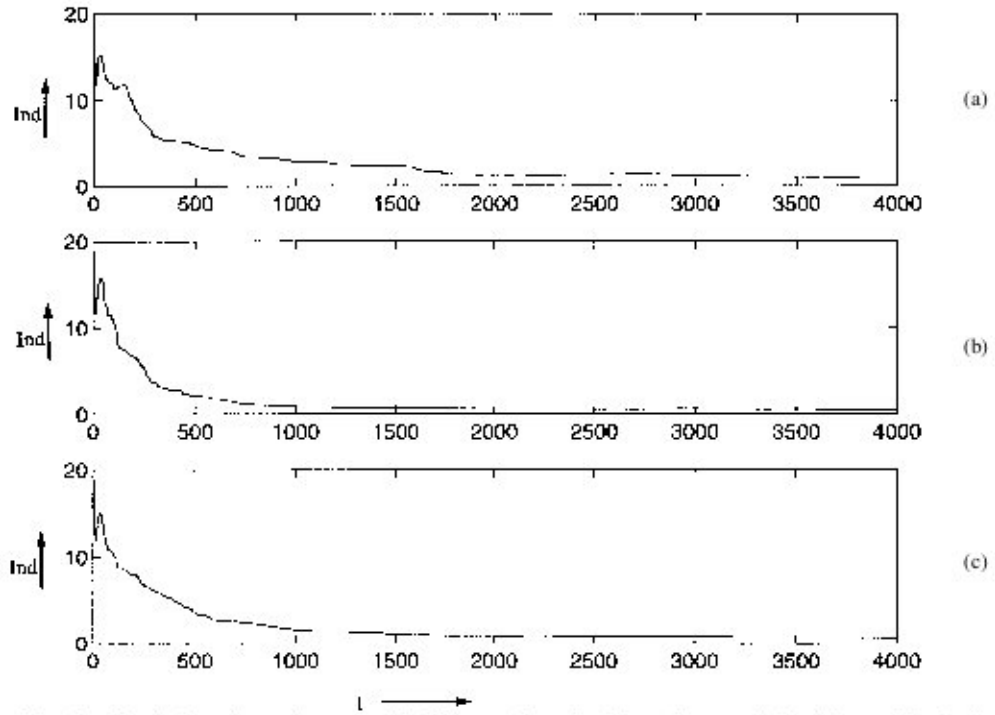

Fig. 5. The performance of the algorithm in the noisy environment with different values of ϵ. The performance index is denoted by "Ind" and the number of observations by "t." The Gaussian noise with noise amplitude = 0.5 is added to the signal components, i.e., the peak signal to noise ratio (PSNR) is 13.86 dB. (a) The performance of the algorithm with $\epsilon = 0.1$. (b) The performance with $\epsilon = 0.3$. (c) The performance with $\epsilon = 0.5$. The batch size p is equal to 30 in all cases.

respectively. It has been found that the algorithm converges in all three cases although the on-line algorithm with $p = 1$ is less stable as compared to $p = 10$ and $p = 50$. No significant enhancement of the algorithm has been found with the increase in the batch size beyond $p = 50$.

The performance of the new algorithm is also demonstrated in the presence of noise. The input is generated as

$$\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{s}(t) + N\boldsymbol{n}(t) \tag{97}$$

where $\boldsymbol{n}(t)$ is $\mathcal{N}(0, \boldsymbol{I})$ Gaussian noise, and N is the noise amplitude. The performance is tested with different values of ϵ, with a fixed batch size of $p = 30$. A sample performance is illustrated in Fig. 5 with a high noise amplitude of $N = 0.5$, i.e., with a peak signal to noise ratio (PSNR) of 13.86 dB. The values of ϵ are chosen as 0.1, 0.3, and 0.5, respectively, and the corresponding results are shown in Fig. 5(a)–(c), respectively. The results demonstrate the fact that the incorporation of a factor of $\epsilon^2 \boldsymbol{W}\boldsymbol{W}'$ in the updating rule (Section IV-F) enhances the performance of the algorithm in the presence of noise.

VII. Discussion and Conclusions

An algorithm is presented for the blind separation of a mixture of uniformly distributed signals. The algorithm is found to be successful in separating the mixed signals for arbitrary mixing matrix even in the presence of large amount of noise. The learning rule is defined based on the required geometric transformation of the hyperbox spanned by the weight matrix. For special cases like rotation, the solution obtained by minimizing the geometric error is identical to the maximum likelihood solution. In general condition, it is difficult to obtain a closed form maximum likelihood solution for an arbitrary mixing matrix, however, the maximum likelihood equation in the noisy condition, provides a guideline for formulating the learning rule of geometric error minimization in the presence of noise.

The effectiveness of the algorithm is demonstrated with randomly generated data in the different noisy conditions and noiseless condition. The hyperbox needs to be transformed by orthogonalizing the transformation operations, i.e., the hyperbox needs to be rotated, then sheared and scaled respectively, each time based on the residual geometric error due to the observed sample. The rotation causes a transformation of the hyperbox without changing its shape and volume, and it is performed first. The sequence of shear and scaling, however, can be interchanged without affecting the performance. One of the reasons behind the much better performance of the orthogonalization of transformation operations is that without orthogonalization, all n^2 parameters (weights) can change independently causing changes in shape, volume and orientation simultaneously. On the other hand, orthogonalization tries to preserve the maximum compactness of the hyperbox if we start from scaled identity matrix as the initial $\boldsymbol{W}$. A thorough theoretical explanation about why the orthogonalization performs much better still needs to be investigated.

The performance of the algorithm in the noisy condition is controlled by the parameter ϵ. Theoretically, if ϵ increases the Fisher information near the boundary of the hyperbox gets more widespread. It has been shown in [7] that when $\epsilon \to 0$ and there is only orthogonal mixing matrix (i.e., only rotation is necessary), the algorithm behaves like an $O(1/T^2)$ convergent algorithm. Ideally, with the increase in ϵ the algorithm should be more stable in the presence of noise

at the cost of speed and vice-versa. The experimental results illustrate some similar effects on the algorithm.

Finally, it is to be noted that the algorithm is successful in separating the uniformly distributed sources even in the presence of noise. On the contrary, the existing general algorithms like K–L divergence measure based algorithm [6], [24], EASI algorithm [10] completely fail to converge in the noiseless condition for the uniformly distributed signals.

REFERENCES

[1] S. Amari, "Neural theory of association and concept formation," *Biol. Cybern.*, vol. 26, pp. 175–185, 1977.

[2] ——, "Natural gradient works efficiently in learning," *Neural Comput.*, vol. 10, pp. 251–276, 1998.

[3] S. Amari, T. P. Chen, and A. Cichocki, "Stability analysis of adaptive blind source separation," *Neural Networks*, vol. 10, pp. 1345–1351, 1997.

[4] S.-I. Amari and J. F. Cardoso, "Blind source separation—Semiparametric statistical approach," *IEEE Trans. Signal Processing*, vol. 45, pp. 2692–2700, 1997.

[5] S.-I. Amari, A. Cichocki, and H. H. Yang, "Recurrent neural networks for blind separation of sources." in *Proc. Int. Symp. Nonlinear Theory Applicat.*, 1995, pp. 37–42.

[6] ——, "A new learning algorithm for blind signal separation." in *Neural Information Processing Systems: Natural and Synthetic, NIPS'96*, D. S. Touretzky, M. C. Mozer, and E. Hasselmo, Eds. Cambridge, MA: MIT Press, 1996, pp. 757–763.

[7] J. Basak and S. Amari, "Blind separation of a mixture of uniformly distributed signals," *Neural Comput.*, vol. 11, pp. 1011–1034, 1999.

[8] ——, "Blind separation of uniformly distributed source signals," in *Proc. Int. Symp. Nonlinear Theory Applicat.*, 1997, pp. 997–1000.

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

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

[11] A. Cichocki, S.-I. Amari, M. Adachi, and W. Kasprzak, "Self-adaptive neural networks for blind separation of sources," in *Proc. Int. Symp. Circuits Syst.*, Atlanta, GA, USA, 1996, pp. 157–161.

[12] P. Comon, "Independent component analysis, a new concept?," *Signal Processing*, vol. 36, pp. 287–314, 1994.

[13] A. Hyvarinen and E. Oja, "Independent component analysis by general nonlinear Hebbian-like learning rules," *Signal Processing*, vol. 64, pp. 301–313, 1998.

[14] 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.

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

[16] ——, "Genralizations of principal component analysis, optimization problems, and neural networks," *Neural Networks*, vol. 8, pp. 549–562, 1995.

[17] S. Kullback," *Information Theory and Statistics*. New York: Wiley, 1959.

[18] K. Matsuoka, M. Ohya, and M. Kawamoto, "A neural net for blind separation of nonstationary signals," *Neural Networks*, vol. 8, pp. 411–419, 1995.

[19] E. Oja, "A simplified neuron model as a principal component analyzer," *J. Math. Biol.*, vol. 15, pp. 267–273, 1982.

[20] E. Oja and J. Karhunen, "Signal separation by nonlinear hebbian learning." in *Computational Intelligence—A Dynamic Systems Perspective*, M. Palaniswami, Y. Attikiouzel, R. MarksII, D. Fogel, and T. Fukuda, Eds. New York: IEEE Press, 1995, pp. 83–97.

[21] E. Oja, J. Karhunen, L. Wang, and R. Vigario, "Principal and independent components in neural networks—Recent developments," in *Proc. Italian Wkshp. Neural Networks, WIRN'95*, Vietri, Italy.

[22] A. Prieto, C. G. Puntonet, and B. Prieto, "A neural learning algorithm for blind separation of sources based on geometric properties," *Signal Processing*, vol. 64, pp. 315–331, 1998.

[23] C. R. Rao," *Linear Statistical Inference and Its Applications*. New York: Wiley, 1973.

[24] H. H. Yang and S. Amari, "Adaptive on-line learning algorithms for blind separation—Maximum entropy and minimum mutual information," *Neural Comput.*, vol. 9, pp. 1457–1482, 1997.

Jayanta Basak (M'95–SM'99) was born in Calcutta, India, on September 25, 1965. He received the Bachelor's degree in electronics and telecommunication engineering from Jadavpur University, Calcutta, in 1987, and the Master's degree in computer science and engineering from the Indian Institute of Science (IISc), Bangalore, in 1989. He received the Ph.D. degree from the Indian Statistical Institute (ISI), Calcutta, in 1995.

He served as a Computer Engineer in the Knowledge-Based Computer Systems project of ISI, Calcutta, from 1989 to 1992. In 1992, he joined as a faculty of the Machine Intelligence Unit of ISI, Calcutta. Presently, he is an Associate Professor in the same unit of ISI. He was a Researcher in the RIKEN Brain Science Institute, Saitama, Japan during 1997 to 1998, and a Visiting Scientist in the Robotics Institute of Carnegie Mellon University, Pittsburgh, PA, during 1991 to 1992. His research interests include neural networks, pattern recognition, image analysis, and fuzzy sets.

Dr, Basak received the gold medal from Jadavpur University in 1987, the young scientist award in engineering sciences from Indian National Science Academy (INSA) in 1996, and junior scientist award in Computer Science from Indian Science Congress Association in 1994.

Shun-ichi Amari (M'71–SM'92–F'94) was born in Tokyo, Japan, on January 3, 1936. He received the bachelor's degree in mathematical engineering from the University of Tokyo in 1958 and the Dr.Eng. degree from the University of Tokyo in 1963.

He was an Associate Professor at Kyushu University, an Associate and then Full Professor at the Department of Mathematical Engineering and Information Physics, University of Tokyo, and is now Professor-Emeritus at the University of Tokyo. He is the Director of the Brain-Style Information Systems Group, RIKEN Brain Science Institute, Saitama, Japan. He has been engaged in research in wide areas of mathematical engineering and applied mathematics, such as topological network theory, differential geometry of continuum mechanics, pattern recognition, mathematical foundations of neural networks, and information geometry.

Dr. Amari served as President of the International Neural Network Society, Council Member of Bernoulli Society for Mathematical Statistics and Probability Theory, and Vice President of the Institute of Electrical, Information, and Communication Engineers. He was founding Coeditor-in-Chief of *Neural Networks*. He has been awarded the Japan Academy Award, the IEEE Neural Networks Pioneer Award, and the IEEE Emanuel R. Piore Award.