Neural Computation, vol.9, pp.1457-1482, 1997

Adaptive On-Line Learning Algorithms for Blind Separation — Maximum Entropy and Minimum Mutual Information

Howard Hua Yang and Shun-ichi Amari
Lab. for Information Representation, FRP, RIKEN
Hirosawa 2-1, Wako-shi, Saitama 351-01, JAPAN
FAX: +81 48462 4633
E-mails: hhy@koala.riken.go.jp, amari@zoo.riken.go.jp

Abstract

There are two major approaches for blind separation: Maximum Entropy (ME) and Minimum Mutual Information (MMI). Both can be implemented by the stochastic gradient descent method for obtaining the de-mixing matrix. The MI is the contrast function for blind separation while the entropy is not. To justify the ME, the relation between ME and MMI is firstly elucidated by calculating the first derivative of the entropy and proving that 1) the the mean-subtraction is necessary in applying the ME and 2) at the solution points determined by the MI the ME will not update the de-mixing matrix in the directions of increasing the cross-talking.

Secondly, the natural gradient instead of the ordinary gradient is introduced to obtain efficient algorithms, because the parameter space is a Riemannian space consisting of matrices. The mutual information is calculated by applying the Gram-Charlier expansion to approximate probability density functions of the outputs.

Finally, we propose an efficient learning algorithm which incorporates with an adaptive method of estimating the unknown cumulants. It is shown by computer simulation that the convergence of the stochastic descent algorithms is improved by using the natural gradient and the adaptively estimated cumulants.

1 Introduction

Let us consider the case where a number of observed signals are mixtures of the same number of stochastically independent source signals. It is desirable to recover the original source signals from the observed mixtures. The blind separation problem is to find a transform which recovers original signals without knowing how the sources are mixed. A learning algorithm for blind source separation belongs to the category of unsupervised factorial learning (Deco and Brauer 1996). It is related to the redundancy reduction principle (Barlow and Földiák 1989; Nadal and Parga 1994), one of the principles possibly employed by the nervous system to extract the statistically independent features for a better representation of the input without losing information. When the mixing model is arbitrarily non-linear, it is generally impossible to separate the sources from the mixture unless further knowledge about the sources and the mixing model is assumed. In this paper, we only discuss the linear mixing model for which the inverse transform (the de-mixing transform) is also linear.

Maximum Entropy (ME) (Bell and Sejnowski 1995a) and Minimum Mutual Information (MMI) or ICA (Amari et al. 1996; Comon 1994) are two major approaches to derive blind separation algorithms. By the MMI, the mutual information (MI) of the outputs is minimized to find the de-mixing matrix. This approach is based on the established ICA theory (Comon 1994). The MI is one of the best contrast functions since it is invariant under transforms such as scaling, permutation, and componentwise non-linear transforms. Since scaling and permutation are indeterminacy in the blind separation problem, therefore, it is desirable to choose a contrast function which is invariant to these transforms. All the global minima (zero points) of the MI are all possible solutions to the blind separation problem. By the ME approach, the outputs of the de-mixing system are first componentwise transformed by sigmoid functions, and then the joint entropy of the transformed outputs is maximized to find a de-mixing matrix. The on-line algorithm derived by the ME is concise and often effective in practice. But the ME has not yet been rigorously justified except for the case when the sigmoid functions happen to be the cumulative density functions of the unknown sources (Bell and Sejnowski 1995b). We study the relation between the ME and MMI, and give a justification to the ME approach.

In order to realize the MMI by on-line algorithms, we need to estimate the MI. To this end, we apply the Gram-Charlier expansion and the Edgeworth expansion to approximate the probability density functions of the outputs. We then have the stochastic gradient type on-line algorithm similar to the one obtained from the ME.

In order to speed up the algorithm, we show that the natural gradient descent method should be used to minimize the estimated MI rather than the conventional gradient. This is because the stochastic gradient descent takes place in the space of matrices. This space has a natural Riemannian structure from which the natural gradient is obtained. We also apply this idea to improve the on-line algorithms based on the ME.

Although the background philosophies of the ME and MMI are different, it is interesting to know that both approaches result in algorithms of a similar form with different non-linearity. Both of them include unknown factors to be fixed adequately or to be estimated. They are activation functions in the case of ME and cumulants κ_3 and κ_4 in the case of MMI. Their optimal values are determined by the unknown probability distributions of the sources. The point is that the algorithms still work even if the specification of the activation functions or cumulants are not accurate. However, efficiency of the algorithms becomes worse by misspecification.

In order to obtain efficient algorithms, we propose an adaptive method together with on-line learning algorithms in which the unknown factors are adequately estimated. The performances of the adaptive on-line algorithms are compared with the fixed on-line algorithms based on simulation results. It is shown by simulations that the adaptive algorithms works much better in various examples.

The paper is organized in the following way. The blind separation problem is described in Section 2. The relation between the ME and the MMI is discussed in Section 3. The gradient descent algorithms based on ME and MMI are derived in Section 4. The natural gradient is also shown but its derivation is given in Appendix B. The Gram-Charlier expansion and the Edgeworth expansion are applied in this section to estimate the MI. The adaptive on-line algorithms based on the MMI together with an adaptive method of estimating the unknown cumulants are proposed in Section 5. The performances of the adaptive algorithms are compared with the fixed ones by the simulations in Section 6. Finally, the conclusions are made in Section 7.

2 Problem

Let us consider n unknown source signals $S_i(t), i = 1, \cdots, n$, which are mutually independent at any fixed time t. We shall denote random variables by capital letters, and their specific values by the same letters in lower case. The bold capital letters denote random vectors or matrices. We assume that the sources $S_i(t)$ are stationary processes, each source has moments of any order and at most one source is Gaussian. We also treat visual signals where t should be replaced by the spatial coordinates (x, y) with $S_i(x, y)$ representing the brightness of the pixel at (x, y). The model for the sensor outputs is

$$\boldsymbol{X}(t) = \boldsymbol{A}\boldsymbol{S}(t)$$

where $\boldsymbol{A} \in \boldsymbol{R}^{n\times n}$ is an unknown non-singular mixing matrix, $\boldsymbol{S}(t) = [S_1(t), \cdots, S_n(t)]^T$ and $\boldsymbol{X}(t) = [X_1(t), \cdots, X_n(t)]^T$ and T denotes the transposition.

Without knowing the source signals and the mixing matrix, we want to recover the original signals from the observations $\boldsymbol{X}(t)$ by the following linear transform:

$$\boldsymbol{Y}(t) = \boldsymbol{W}\boldsymbol{X}(t)$$

where $\boldsymbol{Y}(t) = [Y_1(t), \cdots, Y_n(t)]^T$ and $\boldsymbol{W} \in \boldsymbol{R}^{n\times n}$ is a matrix. When $\boldsymbol{W}$ is equal to $\boldsymbol{A}^{-1}$ we have $\boldsymbol{Y}(t) = \boldsymbol{S}(t)$.

However, it is impossible to obtain the original sources $S_i(t)$ in exact order and amplitude because of the indeterminacy of permutation of $\{S_i\}$ and scaling due to the product of two unknowns: the mixing matrix $\boldsymbol{A}$ and the source vector $\boldsymbol{S}(t)$. Nevertheless, subject to a permutation of indices, it is possible to obtain the scaled sources $c_iS_i(t)$ where the constants c_i are nonzero scalar factors. The source signals are identifiable in this sense. Our goal is to find a de-mixing matrix $\boldsymbol{W}$ adaptively so that $[Y_1, \cdots, Y_n]$ coincides with a permutation of scaled $[S_1, \cdots, S_n]$. In this case, this de-mixing matrix $\boldsymbol{W}$ is written as

$$\boldsymbol{W} = \boldsymbol{\Lambda}\boldsymbol{P}\boldsymbol{A}^{-1}$$

where $\boldsymbol{\Lambda}$ is a non-singular diagonal matrix and $\boldsymbol{P}$ a permutation matrix.

3 Maximizing Entropy vs Minimizing Mutual Information

There are two well known methods for blind separation: 1) maximizing the entropy (ME) of the transformed outputs and 2) minimizing the mutual information (MMI) of $\boldsymbol{Y}$ so that its components become independent. We discuss the relation between ME and MMI.

3.1 Some properties of entropy and mutual information

The idea of ME originated from neural networks. Let us transform $y_a = \sum_j w_{aj}x_j$ by a sigmoid function $g_a(y)$ to $z_a = g_a(y_a)$ which is regarded as the output from an analog neuron.

Let $\boldsymbol{Z} = (g_1(Y_1), \cdots, g_n(Y_n))$ be the componentwise transformed output vector by sigmoid functions $g_a(y), a = 1, \cdots, n$. It is expected that the entropy of the output $\boldsymbol{Z}$ is maximized when the components Z_a of $\boldsymbol{Z}$ are mutually independent. The blind separation algorithm based on ME (Bell and Sejnowski 1995a) was derived by maximizing the joint entropy $H(\boldsymbol{Z}; \boldsymbol{W})$ with respect to $\boldsymbol{W}$ by using the stochastic gradient descent method. The joint entropy of $\boldsymbol{Z}$ is defined by

$$H(\boldsymbol{Z}; \boldsymbol{W}) = -\int p(\boldsymbol{z}; \boldsymbol{W}) \log p(\boldsymbol{z}; \boldsymbol{W}) d\boldsymbol{z}$$

where $p(\boldsymbol{z}; \boldsymbol{W})$ is the joint probability density function (pdf) of $\boldsymbol{Z}$ determined by $\boldsymbol{W}$ and $\{g_a\}$.

The non-linear transformations $g_a(y)$ are necessary for bounding the entropy in a finite range. Indeed, when $g(y)$ is bounded $c \leq g(y) \leq d$, for any random variable Y the entropy of $Z = g(Y)$ has an upper bound

$$H(Z) \leq \log(d-c).$$

Therefore, the entropy of the transformed output vector is upper bounded:

$$H(\boldsymbol{Z};\boldsymbol{W}) \leq \sum_{a=1}^{n} H(Z_a) \leq n\log(d-c). \tag{1}$$

In fact, the above inequality holds for any bounded transforms. So the global maximum of the entropy $H(\boldsymbol{Z};\boldsymbol{W})$ exists. $H(\boldsymbol{Z};\boldsymbol{W})$ may also have many local maxima determined by the functions $\{g_a\}$ used to transform $\boldsymbol{Y}$. By studying the relation between ME and MMI, we shall prove that some of these maxima are the de-mixing matrices with the form $\boldsymbol{\Lambda P A}^{-1}$.

The basic idea of MMI is to choose $\boldsymbol{W}$ that minimizes the dependence among the components of $\boldsymbol{Y}$. The dependence is measured by the Kullback-Leibler divergence $I(\boldsymbol{W})$ between the joint probability density function (pdf) $p(\boldsymbol{y};\boldsymbol{W})$ of $\boldsymbol{Y}$ and its factorized version $\tilde{p}(\boldsymbol{y};\boldsymbol{W})$ which is the product of the marginal probability density functions of $\boldsymbol{Y}$:

$$I(\boldsymbol{W}) = D[p(\boldsymbol{y};\boldsymbol{W}) \parallel \tilde{p}(\boldsymbol{y};\boldsymbol{W})] = \int p(\boldsymbol{y};\boldsymbol{W}) \log \frac{p(\boldsymbol{y};\boldsymbol{W})}{\tilde{p}(\boldsymbol{y},\boldsymbol{W})} d\boldsymbol{y} \tag{2}$$

where $\tilde{p}(\boldsymbol{y};\boldsymbol{W}) = \prod_{a=1}^{n} p_a(y_a;\boldsymbol{W})$ and $p_a(y_a;\boldsymbol{W})$ is the marginal pdf of $\boldsymbol{y}$,

$$p_a(y_a;\boldsymbol{W}) = \int p(\boldsymbol{y};\boldsymbol{W}) dy_1 \cdots \check{dy}_a \cdots dy_n,$$

$\check{dy}_a$ denoting that dy_a is missing from $d\boldsymbol{y} = dy_1 \cdots dy_n$.

The Kullback-Leibler divergence (2) gives the MI of $\boldsymbol{Y}$, and is written in terms of entropies

$$I(\boldsymbol{W}) = -H(\mathbf{Y};\boldsymbol{W}) + \sum_{a=1}^{n} H(Y_a;\boldsymbol{W}) \tag{3}$$

where

$H(\mathbf{Y};\boldsymbol{W}) = -\int p(\boldsymbol{y};\boldsymbol{W}) \log p(\boldsymbol{y};\boldsymbol{W}) d\boldsymbol{y}$,

and

$H(Y_a;\boldsymbol{W}) = -\int p_a(y_a;\boldsymbol{W}) \log p_a(y_a;\boldsymbol{W}) dy_a$

is the marginal entropy.

It is easy to show that $I(\boldsymbol{W}) \geq 0$ and is equal to zero if and only if Y_a are independent. As it is proved in (Comon 1994), $I(\boldsymbol{W})$ is a contrast function for the ICA, meaning

$$I(\boldsymbol{W}) = 0 \text{ iff } \boldsymbol{W} = \boldsymbol{\Lambda P A}^{-1}.$$

In contrast to the entropy criterion, the mutual information $I(\boldsymbol{W})$ is invariant under componentwise monotonic transformations, scaling and permutation of $\boldsymbol{Y}$. Let $\{g_i(y)\}$ be differentiable monotonic functions and $\boldsymbol{Z} = (g_1(Y_1), \cdots, g_n(Y_n))$ be the transformed outputs. It is easy to prove that the mutual information $I(\boldsymbol{W})$ is the same for $\boldsymbol{Y}$ and $\boldsymbol{Z}$. This implies that the componentwise nonlinear transformation is not necessary as a preprocessing if we use the MI as a criterion.

3.2 Relation between ME and MMI

The blind separation algorithm derived by the ME approach is concise and often very effective in practice. However, the ME is not rigorously justified except for the case in which the sigmoid transforms happen to be the cumulative distribution functions (cdfs) of the unknown sources.

It is discussed in (Nadal and Parga 1994; Bell and Sejnowski 1995a) that the ME does not necessarily lead to a statistically independent representation. To justify the ME approach, we study how ME is related to MMI. We prove that, when all sources have a zero mean, ME gives locally correct solutions, otherwise it is not true in general.

Let us first consider the relation between the entropy and the mutual information. Since $\boldsymbol{Z}$ is componentwise nonlinear transform of $\boldsymbol{Y}$, it is easy to obtain

$$\begin{aligned} H(\boldsymbol{Z};\boldsymbol{W}) &= H(\boldsymbol{Y};\boldsymbol{W}) + \sum_{a=1}^{n} \int dy_a p(y_a;\boldsymbol{W}) \log g_a'(y_a) \\ &= -I(\boldsymbol{W}) + \sum_{a=1}^{n} H(Y_a,\boldsymbol{W}) + \sum_{a=1}^{n} \int dy_a p(y_a;\boldsymbol{W}) \log g_a'(y_a) \\ &= -I(\boldsymbol{W}) - \sum_{a=1}^{n} D[p(y_a;\boldsymbol{W}) \parallel g'(y_a)] \end{aligned}$$

Hence, we have the following equation:

$$-H(\boldsymbol{Z};\boldsymbol{W}) = I(\boldsymbol{W}) + D[\tilde{p}(\boldsymbol{y};\boldsymbol{W}) \parallel g'(\boldsymbol{y})] \tag{4}$$

where

$$g'(\boldsymbol{y}) = \prod_{a=1}^{n} g_a'(y_a)$$

is regarded as a pdf of an independent random vector provided, for each a, $g_a(-\infty) = 0$, $g_a(\infty) = 1$ and $g_a'(y)$ is the derivative of $g_a(y)$.

Let $\{r_a(s_a)\}$ be the pdf's of the independent sources $S_a(t)$ at any t. The joint pdf of the sources is $r(\boldsymbol{s}) = \prod_{a=1}^{n} r_a(s_a)$. When $\boldsymbol{W} = \boldsymbol{A}^{-1}$, we have $\boldsymbol{Y} = \boldsymbol{S}$ so that $p(y_a;\boldsymbol{A}^{-1}) = r_a(y_a)$. Hence, from (4) it is straightforward that if $\{g_a(y_a)\}$ are the cdfs of the sources, $r(\boldsymbol{y}) = g'(\boldsymbol{y})$ so that $D[r(\boldsymbol{y}) \parallel g'(\boldsymbol{y})] = 0$. In this case, $H(\boldsymbol{Z};\boldsymbol{A}^{-1}) = -I(\boldsymbol{A}^{-1}) = 0$. Since $H(\boldsymbol{Z};\boldsymbol{W}) \leq 0$ from (1), the entropy $H(\boldsymbol{Z};\boldsymbol{W})$ achieves the global maximum at $\boldsymbol{W} = \boldsymbol{A}^{-1}$. Let $\tilde{r}(\boldsymbol{y})$ be the joint pdf of the scaled and permutated sources $\boldsymbol{y} = \boldsymbol{\Lambda P s}$. It is easy to prove that $H(\boldsymbol{Z};\boldsymbol{W})$ achieves the global maximum at $\boldsymbol{W} = \boldsymbol{\Lambda P A}^{-1}$, too. This was also discussed in (Nadal and Parga 1994; Bell and Sejnowski 1995b) from different perspectives. The above formula (4) can also be derived from the formula (24) in (Nadal and Parga 1994).

We now study the general case where $g'(\boldsymbol{y})$ is different from $r(\boldsymbol{y})$. We decompose $H(\boldsymbol{Z};\boldsymbol{W})$ as

$$-H(\boldsymbol{Z};\boldsymbol{W}) = I(\boldsymbol{W}) + D(\boldsymbol{W}) + C(\boldsymbol{W}) \tag{5}$$

where

$$\begin{aligned} I(\boldsymbol{W}) &= I(\boldsymbol{y};\boldsymbol{W}), \\ D(\boldsymbol{W}) &= D[\tilde{p}(\boldsymbol{y},\boldsymbol{W}) \parallel r(\boldsymbol{y})], \\ C(\boldsymbol{W}) &= \sum_{a=1}^{n} \int dy_a p(y_a;\boldsymbol{W}) \log k_a(y_a) = \sum_{a=1}^{n} \int d\boldsymbol{y} p(\boldsymbol{y};\boldsymbol{W}) \log k_a(y_a), \\ k_a(y_a) &= \frac{r_a(y_a)}{g_a'(y_a)}. \end{aligned}$$

Since $p(y_a, \boldsymbol{A}^{-1}) = r_a(y_a)$, $I(\boldsymbol{W})$ and $D(\boldsymbol{W})$ take the minimum value zero at $\boldsymbol{W} = \boldsymbol{A}^{-1}$. To understand the behavior of $C(\boldsymbol{W})$, we need to compute its gradient. To facilitate the process for calculating the gradient, we reparameterize $\boldsymbol{W}$ as

$$\boldsymbol{W} = (\boldsymbol{I} + \boldsymbol{B})\boldsymbol{A}^{-1}$$

around $\boldsymbol{W} = \boldsymbol{A}^{-1}$ so that $\boldsymbol{Y} = (\boldsymbol{I} + \boldsymbol{B})\boldsymbol{S}$. We call the diagonal elements $\{B_{bb}\}$ the scaling coordinates and the off-diagonal elements $\{B_{bc}, b \neq c\}$ the cross-talking coordinates. If $\frac{\partial C}{\partial B_{bc}} \neq 0$ for some $b \neq c$ at $\boldsymbol{W} = \boldsymbol{A}^{-1}$, then the gradient descent algorithm based on the ME would increase cross-talking around this point. If $\frac{\partial C}{\partial B_{bc}} = 0$ for all $b \neq c$, even if $\frac{\partial C}{\partial B_{bb}} \neq 0$ for some b, the ME method still gives a correct solution for any nonlinear functions $g_a(y)$. We have the following lemma.

Lemma 1 *At $\boldsymbol{B} = 0$ or $\boldsymbol{W} = \boldsymbol{A}^{-1}$, for $b \neq c$, the gradient of $C(\boldsymbol{W})$ is*

$$\frac{\partial C}{\partial B_{bc}} = -\left(\int dy_c \; y_c r(y_c)\right)\left(\int dy_b \; r_b'(y_b) r_b(y_b) \log k_b(y_b)\right). \tag{6}$$

The proof is given in Appendix A.

The diagonal terms $\frac{\partial C}{\partial B_{bb}}$ does not vanish in general. It is easy to see from (6) that at $\boldsymbol{W} = \boldsymbol{A}^{-1}$ the derivatives $\{\frac{\partial C}{\partial B_{bc}}, b \neq c\}$ vanish when all the sources have a zero mean,

$$\int ds_a \; s_a r(s_a) = 0,$$

or all the sigmoid functions g_a are equal to the cdfs of the sources, i.e., $g'(\boldsymbol{y}) = r(\boldsymbol{y})$.

Similarly, reparameterizing the function $C(\boldsymbol{W})$ around around $\boldsymbol{W} = \boldsymbol{\Lambda P A}^{-1}$ using

$$\boldsymbol{W} = (\boldsymbol{I} + \boldsymbol{B})\boldsymbol{\Lambda P A}^{-1},$$

we calculate the gradient $\frac{\partial C}{\partial \boldsymbol{B}}$ at $\boldsymbol{B} = 0$. It turns out that the equation (6) still holds after permutating indexes and the derivatives $\{\frac{\partial C}{\partial B_{bc}}, b \neq c\}$ vanish when all sources have a zero mean or $g'(\boldsymbol{y}) = \tilde{r}(\boldsymbol{y})$ which is the joint pdf of scaled and permutated sources.

Hence, we have the following theorem regarding the relation between the ME and the MMI.

Theorem 1 *When all sources are zero mean signals, at all solution points $\boldsymbol{W} = \boldsymbol{\Lambda P A}^{-1}$ determined by the contrast function MI, the ME algorithms will not update the de-mixing matrix in the directions of increasing the cross-talking. When one of sources has a non-zero mean, the entropy is not maximized in general at $\boldsymbol{W} = \boldsymbol{\Lambda P A}^{-1}$.*

Note the entropy is maximized at $\boldsymbol{W} = \boldsymbol{\Lambda P A}^{-1}$ when the sigmoid functions are equal to the cdfs of the scaled and permutated sources. But, usually we cannot choose the unknown cdfs as the sigmoid functions. In some blind separation problems such as the separation of visual signals, the means of mixture are positive. In such cases, we need a preprocessing step to centralize the mixture:

$$\boldsymbol{x}_t - \overline{\boldsymbol{x}}_t = \boldsymbol{A}(\boldsymbol{s}_t - \overline{\boldsymbol{s}}_t) \tag{7}$$

where $\overline{\boldsymbol{x}}_t = \frac{1}{t}\sum_{u=1}^{t} \boldsymbol{x}_u$ and $\overline{\boldsymbol{s}}_t$ is similarly defined. The mean subtraction can also be implemented by online thresholds for the sigmoid functions applied to the outputs.

Applying a blind separation algorithm, we find a matrix $\boldsymbol{W}$ close to one of solution points $\boldsymbol{\Lambda P A}^{-1}$ such that $\boldsymbol{W}(\boldsymbol{x}_t - \overline{\boldsymbol{x}}_t) \approx \boldsymbol{\Lambda P}(\boldsymbol{s}_t - \overline{\boldsymbol{s}}_t)$, and then obtain the sources by

$$\boldsymbol{W}\boldsymbol{x}_t = \boldsymbol{W}(\boldsymbol{x}_t - \overline{\boldsymbol{x}}_t) + \boldsymbol{W}\overline{\boldsymbol{x}}_t \approx \boldsymbol{\Lambda P}\boldsymbol{s}_t.$$

Since every linear mixture model can be reformulated as (7), without losing generality, we assume that all sources have a zero mean in the rest of this paper.

4 Gradient descent algorithms based on ME and MMI

The gradient descent algorithms based on ME and MMI are the following:

$$\frac{d\boldsymbol{W}}{dt} = \eta \frac{\partial H(\boldsymbol{Z};\boldsymbol{W})}{\partial \boldsymbol{W}}, \tag{8}$$

$$\frac{d\boldsymbol{W}}{dt} = -\eta \frac{\partial I(\boldsymbol{W})}{\partial \boldsymbol{W}}. \tag{9}$$

However, the algorithms work well if we put a positive-definite operator $\mathcal{G}$ on matrices:

$$\frac{d\boldsymbol{W}}{dt} = \eta \mathcal{G} \star \frac{\partial H(\boldsymbol{Z};\boldsymbol{W})}{\partial \boldsymbol{W}} \tag{10}$$

$$\frac{d\boldsymbol{W}}{dt} = -\eta \mathcal{G} \star \frac{\partial I(\boldsymbol{W})}{\partial \boldsymbol{W}} \tag{11}$$

The gradients themselves are not obtainable in general. In practice, they are replaced by the stochastic ones whose expectations give the true gradients. This method is called the stochastic gradient descent method known from the very old time in neural network community (e.g., Amari 1967). Some remarks on the the stochastic gradient and backpropagation are given in (Amari 1993).

4.1 Stochastic gradient descent method based on ME

Note that the entropy $H(\boldsymbol{Z};\boldsymbol{W})$ can be written as

$$H(\boldsymbol{Z};\boldsymbol{W}) = H(\boldsymbol{X}) + \log|\boldsymbol{W}| + \sum_{a=1}^{n} E[\log g_a'(Y_a)]$$

where $|\boldsymbol{W}| = |\det(\boldsymbol{W})|$. It is easy to show

$$\frac{\partial \log|\boldsymbol{W}|}{\partial \boldsymbol{W}} = \boldsymbol{W}^{-T}$$

where $\boldsymbol{W}^{-T} = (\boldsymbol{W}^{-1})^T$. Moreover,

$$\frac{\partial \sum_{a=1} \log g_a'(y_a)}{\partial \boldsymbol{W}} = -\boldsymbol{\Phi}(\boldsymbol{y})\boldsymbol{x}^T$$

where

$$\boldsymbol{\Phi}(\boldsymbol{y}) = (-\frac{g_1''(y_1)}{g_1'(y_1)} \cdots - \frac{g_n''(y_n)}{g_n'(y_n)})^T. \tag{12}$$

Hence, the expectation of the instantaneous values of the stochastic gradient

$$\frac{\widehat{\partial H}}{\partial \boldsymbol{W}} = \boldsymbol{W}^{-T} - \boldsymbol{\Phi}(\boldsymbol{y})\boldsymbol{x}^T \tag{13}$$

gives the gradient $\frac{\partial H}{\partial \boldsymbol{W}}$. Based on this, the following algorithm proposed by (Bell and Sejnowski 1995a) is obtained from (10) and (13):

$$\frac{d\boldsymbol{W}}{dt} = \eta(\boldsymbol{W}^{-T} - \boldsymbol{\Phi}(\boldsymbol{y})\boldsymbol{x}^T) \tag{14}$$

Here, the learning equation is in the form of (10) with the operator $\mathcal{G}$ equal to the identity matrix. We show a more adequate $\mathcal{G}$ later. Note that when the transforms are $\tanh(x)$ and $\int_{-\infty}^{x} \exp(-\frac{1}{4}u^4)du$, the corresponding activation functions g_a are $2\tanh(x)$ and x^3, respectively.

We show in Appendix B that the natural choice of the operator $\mathcal{G}$ should be

$$\mathcal{G} \star \frac{\partial H}{\partial \boldsymbol{W}} = \frac{\partial H}{\partial \boldsymbol{W}} \boldsymbol{W}^T \boldsymbol{W}.$$

This is given by the Riemannian structure of the parameter space of matrices $\boldsymbol{W}$ (Amari 1996). We then obtain the following algorithm from (14):

$$\frac{d\boldsymbol{W}}{dt} = \eta(\boldsymbol{I} - \boldsymbol{\Phi}(\boldsymbol{y})\boldsymbol{y}^T)\boldsymbol{W} \qquad (A)$$

which is not only computationally easy but also very efficient. The learning rule of form (A) was proposed in (Cichocki et al 1994) and later justified in (Amari et al. 1996).

The learning rule (A) has two important properties: the equivariant property (Cardoso and Laheld 1996) and the property of keeping $\boldsymbol{W}(t)$ from becoming singular, whereas the learning rule (14) does not have these properties. To prove the second property, we define $< \boldsymbol{X}, \boldsymbol{Y} >= \mathrm{tr}(\boldsymbol{X}^T\boldsymbol{Y})$ and calculate

$$\begin{aligned}\frac{d|\boldsymbol{W}|}{dt} &=< \frac{\partial|\boldsymbol{W}|}{\partial \boldsymbol{W}}, \frac{d\boldsymbol{W}}{dt} >=< |\boldsymbol{W}|\boldsymbol{W}^{-T},\ \eta(\boldsymbol{I} - \Phi(\boldsymbol{y})\boldsymbol{y}^T)\boldsymbol{W} > \\ &= \eta\mathrm{tr}(\boldsymbol{I} - \Phi(\boldsymbol{y})\boldsymbol{y}^T)|\boldsymbol{W}| = \eta(\sum_{i=1}^{n}(1 - \phi_i(y_i)y_i))|\boldsymbol{W}|.\end{aligned}$$

Then, we obtain an expression for $|\boldsymbol{W}(t)|$

$$|\boldsymbol{W}(t)| = |\boldsymbol{W}(0)| \exp\{\eta \int_0^t \sum_{i=1}^{n}(1 - \phi_i(y_i(\tau))y_i(\tau))d\tau\} \qquad (15)$$

from which we know that $|\boldsymbol{W}(t)| \neq 0$ if $|\boldsymbol{W}(0)| \neq 0$. This means that

$$\{\boldsymbol{X} \in R^{n\times n} : \quad |\boldsymbol{X}| \neq 0\}$$

is an invariant set of the flow $\boldsymbol{W}(t)$ driven by the learning rule (A).

It is worth pointing out that the algorithm (14) is equivalent to the sequential maximum likelihood algorithm when g_a are taken to be equal to r_a. For $r(\boldsymbol{s}) = r_1(s_1)\cdots r_n(s_n)$, the likelihood function is

$$\log p(x; \boldsymbol{W}) = \log(r(\boldsymbol{W}\boldsymbol{x})|\boldsymbol{W}|) = \sum_{a=1}^{n} \log(r_a(y_a)) + \log|\boldsymbol{W}|$$

from which we have

$$\frac{\partial \log p(x; \boldsymbol{W})}{\partial w_{ak}} = \frac{r'_a(y_a)}{r_a(y_a)} x_k + (\boldsymbol{W}^{-T})_{ak}$$

and in the matrix form

$$\frac{\partial \log p(x; \boldsymbol{W})}{\partial \boldsymbol{W}} = \boldsymbol{W}^{-T} - \Psi(\boldsymbol{y})\boldsymbol{x}^T$$

where $\Psi(\boldsymbol{y}) = (-\frac{r'_1(y_1)}{r_1(y_1)} \cdots - \frac{r'_n(y_n)}{r_n(y_n)})^T$ and w_{ak} is the (a,k) elements of $\boldsymbol{W}$. Using the natural (Riemannian) gradient descent method to maximize $\log p(x; \boldsymbol{W})$, we have the sequential maximum likelihood algorithm of the type (A):

$$\frac{d\boldsymbol{W}}{dt} = \mu(\boldsymbol{I} - \Psi(\boldsymbol{y})\boldsymbol{y}^T)\boldsymbol{W} \qquad (16)$$

From the point of view of asymptotic statistics, this gives the Fisher-efficient estimator. However, we do not know r_a or $\Psi(\boldsymbol{y})$, so that we need to use the adaptive method of estimating $\Psi(\boldsymbol{y})$ in order to implement (16).

4.2 Approximation of MI

To implement the MMI, we need some formulas to approximate the MI. Generally, it is difficult to obtain the explicit form of the MI since the pdf's of the outputs are unknown. The main difficulty is to calculate the marginal entropy $H(Y_a;\boldsymbol{W})$ explicitly. In order to estimate the marginal entropy, the Edgeworth expansion and the Gram-Charlier expansion were used in (Comon 1994) and (Amari et al. 1996), respectively, to approximate the pdf's of the outputs. The two expansions are formally identical except for the order of summation, but the truncated expansions are different. We show later by computer simulation that the Gram-Charlier expansion is superior to the Edgeworth expansion for blind separation.

In order to calculate each $H(Y_a;\boldsymbol{W})$ in (3), we shall apply the Gram-Charlier expansion to approximate the pdf $p_a(y_a)$. Since we assume $E[\boldsymbol{s}]=0$, we have $E[\mathbf{y}]=E[\boldsymbol{WA}\mathbf{s}]=0$ and $E[y_a]=0$. To simplify the calculations for the entropy $H(y_a;\boldsymbol{W})$ to be carried out later, we assume $m_2^a=E[y_a^2]=1$ for all a. Under this assumption, we approximate each marginal entropy first and then obtain a formula for the MI. The zero mean and unit variance assumption makes it easier for us to calculate the MI. However, the formula obtained for the MI can be used in more general cases. When the components of $\boldsymbol{Y}$ have non-zero means and different variances, we can shift and scale $\boldsymbol{Y}$ to obtain

$$\tilde{\boldsymbol{Y}}=\boldsymbol{\Sigma}^{-1}(\boldsymbol{Y}-\boldsymbol{m})$$

where $\boldsymbol{\Sigma}=\mathrm{diag}(\sigma_1,\cdots,\sigma_n)$, σ_a is the variance of Y_a and $\boldsymbol{m}=E[\boldsymbol{Y}]$. The components of $\tilde{\boldsymbol{Y}}$ have the zero mean and the unit variance. Due to the invariant property of the MI, the formula of computing the MI of $\tilde{\boldsymbol{Y}}$ is applicable to compute the MI of $\boldsymbol{Y}$.

We use the following truncated Gram-Charlier expansion (Stuart and Ord 1994) to approximate the pdf $p_a(y_a)$:

$$p_a(y_a)\approx\alpha(y_a)\{1+\frac{\kappa_3^a}{3!}H_3(y_a)+\frac{\kappa_4^a}{4!}H_4(y_a)\} \tag{17}$$

where $\kappa_3^a=m_3^a$ and $\kappa_4^a=m_4^a-3$ are the third and fourth order cumulants of Y_a, respectively, and $m_k^a=E[y_a^k]$ is the k-th order moment of Y_a, $\alpha(y)=\frac{1}{\sqrt{2\pi}}e^{-\frac{y^2}{2}}$, and $H_k(y), k=1,2,\cdots$, are the Chebyshev-Hermite polynomials defined by the identity

$$(-1)^k\frac{d^k\alpha(y)}{dy^k}=H_k(y)\alpha(y).$$

The Gram-Charlier expansion clearly shows how κ_3^a and κ_4^a affect the approximation of the pdf. The last two terms in (17) characterize the deviations from the Gaussian distributions. To apply (17) to calculate $H(Y_a)$, we need the following integrals:

$$\int\alpha(y)(H_3(y))^2H_4(y)dy=3!^3 \tag{18}$$

$$\int\alpha(y)(H_4(y))^3dy=12^3. \tag{19}$$

These integrals can be obtained easily from the following results for the moments of a Gaussian random variable N(0,1):

$$\int y^{2k+1}\alpha(y)dy = 0, \qquad \int y^{2k}\alpha(y)dy = 1\cdot 3\cdots(2k-1). \tag{20}$$

By using the expansion

$$\log(1+y) \approx y - \frac{y^2}{2} + O(y^3)$$

and taking account of the orthogonality relations of the Chebyshev-Hermite polynomials and (18)-(19), the entropy $H(Y_a;\boldsymbol{W})$ is approximated by

$$H(Y_a;\boldsymbol{W}) \approx \ \frac{1}{2}\log(2\pi e) - \frac{(\kappa_3^a)^2}{2\cdot 3!} - \frac{(\kappa_4^a)^2}{2\cdot 4!} + \frac{3}{8}(\kappa_3^a)^2\kappa_4^a + \frac{1}{16}(\kappa_4^a)^3. \tag{21}$$

Let $F(\kappa_3^a, \kappa_4^a)$ denote the right hand side of the approximation (21). From $\boldsymbol{Y} = \boldsymbol{W}\boldsymbol{X}$, we have $H(\boldsymbol{Y}) = H(\boldsymbol{X}) + \log|\boldsymbol{W}|$. Applying (21) and the above expressions to (3), we have

$$I(\boldsymbol{Y};\mathbf{W}) \approx \ -H(\boldsymbol{X}) - \log|\boldsymbol{W}| + \frac{n}{2}\log(2\pi e) + \sum_{a=1}^{n} F(\kappa_3^a, \kappa_4^a) \tag{22}$$

On the other hand, the following approximation of the marginal entropy (Comon 1994) is obtained by using the Edgeworth expansion of $p_a(y_a)$:

$$H(Y^a;\boldsymbol{W}) \approx \frac{1}{2}\log(2\pi e) - \frac{1}{2\cdot 3!}(\kappa_3^a)^2 - \frac{1}{2\cdot 4!}(\kappa_4^a)^2 - \frac{7}{48}(\kappa_3^a)^4 + \frac{1}{8}(\kappa_3^a)^2\kappa_4^a \tag{23}$$

The terms in the Edgeworth expansion are arranged in a decreasing order by assuming that κ_i^a is of order $n^{(2-i)/2}$. This is true when Y_a is a sum of n independent random variables and the number n is large. To obtain the formula (23), the truncated Edgeworth expansion is used by neglecting those high-order terms higher than $1/n^2$. This is a standard method of asymptotic statistics but is not valid in the present context because of the fixed n. Moreover, at around $\boldsymbol{W} = \boldsymbol{A}^{-1}$, Y_a is close to S_a which is not a sum of independent random variables, so that the Edgeworth expansion is not justified in this case. The learning algorithm derived from (23) does not work well in our simulations. The reason is that the cubic of the 4th order cumulant $(\kappa_4^a)^3$ plays an important role but is omitted in (23). It is a small term from the point view of the Edgeworth expansion but not a small term in the present context. In our simulations, we deal with nearly symmetric source signals. So $(\kappa_3^a)^4$ is small. In this case, we should use the following entropy approximation instead of (23):

$$H(Y_a;\boldsymbol{W}) \approx \frac{1}{2}\log(2\pi e) - \frac{1}{2\cdot 3!}(\kappa_3^a)^2 - \frac{1}{2\cdot 4!}(\kappa_4^a)^2 + \frac{1}{8}(\kappa_3^a)^2\kappa_4^a + \frac{1}{48}(\kappa_4^a)^3 \tag{24}$$

The learning algorithm derived from the above formula works almost equally well as (21).

Note that the expansion formula can be more general. Instead of the Gaussian kernel, we can use other standard distribution function as the kernel to expand $p_a(y_a)$ as:

$$p_a(y) = \beta(y)\{1 + \sum_i \mu_i K_i(y)\}$$

where $K_i(y)$ are the orthogonal polynomials with respect to the kernel $\beta(y)$. For example, the expansion corresponding to the kernel

$$\beta(y) = \begin{cases} e^{-y}, & y \geq 0 \\ 0, & y > 0 \end{cases}$$

will be better than the Gram-Charlier expansion in approximating the pdf of a positive random variable. The orthogonal polynomials corresponding to this kernel are Laguerre polynomials.

4.3 Stochastic gradient method based on MMI

To obtain the stochastic gradient descent algorithm to update $\boldsymbol{W}$ recursively, we need to calculate the gradient of $I(\boldsymbol{W})$ with respect to $\boldsymbol{W}$. Since the exact function form of $I(\boldsymbol{W})$ is unknown, we calculate the derivative using the approximated MI.

Since

$$\begin{aligned} \frac{\partial \kappa_3^a}{\partial w_{ak}} &= 3E[y_a^2 x_k] \text{ and} \\ \frac{\partial \kappa_4^a}{\partial w_{ak}} &= 4E[y_a^3 x_k], \end{aligned}$$

we obtain the following from (22),

$$\frac{\partial I(\boldsymbol{W})}{\partial w_{ak}} \approx -(\boldsymbol{W}^{-T})_{ak} + f(\kappa_3^a, \kappa_4^a)E[y_a^2 x_k] + g(\kappa_3^a, \kappa_4^a)E[y_a^3 x_k] \tag{25}$$

where

$$f(y, z) = -\frac{1}{2}y + \frac{9}{4}yz, \qquad g(y, z) = -\frac{1}{6}z + \frac{3}{2}y^2 + \frac{3}{4}z^2. \tag{26}$$

Removing the expectation symbol $E(\cdot)$ in (25) and writing it in a matrix form, we obtain the stochastic gradient:

$$\frac{\widehat{\partial I}}{\partial \boldsymbol{W}} = -\boldsymbol{W}^{-T} + (\boldsymbol{f}(\kappa_3, \kappa_4) \circ \boldsymbol{y}^2)\boldsymbol{x}^T + (\boldsymbol{g}(\kappa_3, \kappa_4) \circ \boldsymbol{y}^3)\boldsymbol{x}^T \tag{27}$$

whose expectation gives $\frac{\partial I}{\partial \boldsymbol{W}}$, where $\circ$ denotes the Hadamard product of two vectors

$\boldsymbol{f} \circ \boldsymbol{y} = (f_1 y_1, \cdots, f_n y_n)^T$, and
$\boldsymbol{y}^k = [(y_1)^k, \cdots, (y_n)^k]^T$ for $k = 2, 3$,
$\boldsymbol{f}(\kappa_3, \kappa_4) = [f(\kappa_3^1, \kappa_4^1), \cdots, f(\kappa_3^n, \kappa_4^n)]^T$,
$\boldsymbol{g}(\kappa_3, \kappa_4) = [g(\kappa_3^1, \kappa_4^1), \cdots, g(\kappa_3^n, \kappa_4^n)]^T$.

We need to evaluate κ_3 and κ_4 for implementing this idea. If κ_3 and κ_4 are known, using the natural gradient descent method to minimize $I(\boldsymbol{W})$, from (27) we obtain the algorithm based on MMI:

$$\frac{d\boldsymbol{W}}{dt} = \eta(t)\{\boldsymbol{I} - \boldsymbol{\Phi}_\kappa(\boldsymbol{y})\boldsymbol{y}^T\}\boldsymbol{W}, \tag{A'}$$

where

$$\boldsymbol{\Phi}_\kappa(\boldsymbol{y}) = \boldsymbol{h}(\boldsymbol{y}; \kappa_3, \kappa_4) = \boldsymbol{f}(\kappa_3, \kappa_4) \circ \boldsymbol{y}^2 + \boldsymbol{g}(\kappa_3, \kappa_4) \circ \boldsymbol{y}^3. \tag{28}$$

Note each component of $\boldsymbol{\Phi}_\kappa(\boldsymbol{y})$ is a third order polynomial. In particular, if $\kappa_3^i = 0$ and $\kappa_4^i = -1$, the non-linearity $\boldsymbol{\Phi}_\kappa(\boldsymbol{y})$ becomes $\frac{11}{12}\boldsymbol{y}^3$.

The algorithm (A') is based on the entropy formula (21) derived from the Gram-Charlier expansion. Using the entropy formula (24) based on the Edgeworth expansion rather than the formula (21), we obtain an algorithm to be referred as (A''). This algorithm has the same form as the algorithm (A') except for the definition of the functions f and g. For the Edgeworth expansion based algorithm (A''), instead of (26) we use the following definition for f and g:

$$f(y, z) = -\frac{1}{2}y + \frac{7}{4}y^3 + \frac{3}{4}yz, \qquad g(y, z) = -\frac{1}{6}z + \frac{1}{2}y^2. \tag{29}$$

So the algorithm of type (A) is the common form for both the ME algorithm and the MMI algorithm. In the ME approach, the function $\boldsymbol{\Phi}(\boldsymbol{y})$ in (A) is determined by the sigmoid transforms $\{g_a(y_a)\}$. If we use the cdfs of the source distributions, we need to evaluate the unknown $r_a(s_a)$. In the MMI approach, the function $\boldsymbol{\Phi}_\kappa(\boldsymbol{y}) = \boldsymbol{h}(\boldsymbol{y};\kappa_3,\kappa_4)$ depends on the cumulants κ_3^a and κ_4^a and the approximation procedures should be taken for computing the entropy. It is better to choose g_a' equal to unknown r_a or to choose κ_3^a and κ_4^a equal to the true values. However, it is verified by numerous simulations that even if the g_a' or κ_3^a and κ_4^a are misspecified, the algorithm (A) and (A') may still converge to the true separation matrix in many cases.

5 Adaptive On-Line Learning Algorithms

In order to implement an efficient learning algorithm, we propose the following adaptive algorithm to trace κ_3 and κ_4 together with (A'):

$$\begin{aligned} \frac{d\kappa_3^a}{dt} &= -\mu(t)(\kappa_3^a - (y^a)^3), \\ \frac{d\kappa_4^a}{dt} &= -\mu(t)(\kappa_4^a - (y^a)^4 + 3), \ a = 1, \cdots, n, \end{aligned} \qquad (30)$$

where $\mu(t)$ is a learning rate function.

It is possible to use the same adaptive scheme for implementing the efficient ME algorithm. In this case, we use the Gram-Charlier expansion (17) to evaluate $r_a(y_a)$ or $\Psi(\boldsymbol{y})$. The performances of the algorithms (A') with (30) will be compared with that of the algorithms (A) without adaptation in the next section by simulation. The simulation results demonstrate that the adaptive algorithm (A') needs less samples to reach the separation than the fixed algorithm (A).

6 Simulation

In order to show the effect of the on-line learning algorithms (A') with (30) and compare its performance with that of the fixed algorithm (A), we apply these algorithms to separate sources from the mixture of modulating signals or the mixture of visual signals. For the algorithm (A), we can use various fixed $\boldsymbol{\Phi}(\boldsymbol{y})$. For example, we can use one of the following functions

(a) $f(y) = y^3$

(b) $f(y) = 2\tanh(y)$

(c) $f(y) = \frac{3}{4}y^{11} + \frac{15}{4}y^9 - \frac{14}{3}y^7 - \frac{29}{4}y^5 + \frac{29}{4}y^3$

to define $\boldsymbol{\Phi}(\boldsymbol{y}) = (f(y_1), \cdots, f(y_n))^T$. The function **(c)** was used in (Amari et al. 1996) by replacing κ_3^a and κ_4^a by their instantaneous values:

$$\kappa_3^a \sim (y_a)^3, \qquad \kappa_4^a \sim (y_a)^4 - 3.$$

We also compare the adaptive algorithm (A') obtained from the Gram-Charlier expansion with the adaptive algorithm (A'') obtained from the Edgeworth expansion.

6.1 Modulating source signals

Assume that the following five unknown sources are mixed by a mixing matrix $\boldsymbol{A}$ randomly chosen:

$$\boldsymbol{s}(t) = [\mathrm{sign}(\cos(2\pi 155t)), \sin(2\pi 800t), \sin(2\pi 300t + 6\cos(2\pi 60t)), \sin(2\pi 90t), n(t)]^T. \qquad (31)$$

where four components of $\boldsymbol{s}(t)$ are modulating data signals, and one component $n(t)$ is a noise source uniformly distributed in $[-1,+1]$. The elements of the mixing matrix $\boldsymbol{A}$ are randomly chosen subject to the uniform distribution in $[-1,+1]$ such that $\boldsymbol{A}$ is non-singular.

We use the cross-talking error defined below to measure the performance of the algorithms:

$$E=\sum_{i=1}^{n}(\sum_{j=1}^{n}\frac{|p_{ij}|}{\max_k|p_{ik}|}-1)+\sum_{j=1}^{n}(\sum_{i=1}^{n}\frac{|p_{ij}|}{\max_k|p_{kj}|}-1)$$

where $\boldsymbol{P}=(p_{ij})=\boldsymbol{WA}$.

The mixed signals are sampled at the sampling rate of 10K Hz. Taking 2000 samples, we simulate the algorithm (A) with fixed $\boldsymbol{\Phi}$ defined by the functions (**a**) and (**b**), and the algorithms (A') and (A'') with the adaptive $\boldsymbol{\Phi}_\kappa$ defined by (28) for which the functions f and g are defined by (26) and (29) respectively. The learning rate is a constant. For each algorithm, we select a nearly optimal learning rate. The initial matrix is chosen as $\boldsymbol{W}(0)=0.5\boldsymbol{I}$ in all simulations. The sources, mixtures and outputs obtained by using the different algorithms are displayed in Figure 1. In each sub-figure, four out of five components are shifted upwards from the zero level for a better illustration. The sources, mixtures and all the outputs shown there are within the same time window $[0.1,0.2]$. It is shown in Figure 1 that the algorithms with adaptive $\boldsymbol{\Phi}_\kappa$ need less samples than those with fixed $\boldsymbol{\Phi}$ to achieve separation.

The cross-talking errors are plotted in Figure 2. It is shown in Figure 2 that the adaptive MMI algorithms (A') and (A'') outperform the ME algorithm (A) either with fixed function (**a**) or (**b**). Note the cross-talking error for each algorithm can be further decreased if more observations are taken and an exponentially decreasing learning rate is chosen.

6.2 Image data

In this paper, the two basic algorithms (A) and (A') are derived by ME and MMI. The assumption that sources are independent is needed as a sufficient condition for identifiability. To check whether the performance of these algorithms is sensitive to the independence assumption, we test these algorithms on correlated data such as images. Images are usually correlated and non-stationary in the spatial coordinate. In the first row in Figure 3, we have six image sources $\{s_i(x,y), i=1,\cdots,6\}$ consisting of five natural images and one uniformly distributed noise. All images have the same size with N pixels in each of them. From each image, a pixel sequence is taken by scanning the image in a certain order, for example, row by row or column by column. These sources have the following correlation coefficient matrix:

$$\boldsymbol{R}_s=(c_{ij})=\begin{bmatrix} 1.0000 & 0.0819 & -0.1027 & -0.0616 & 0.2154 & 0.0106 \\ 0.0819 & 1.0000 & 0.3158 & 0.0899 & -0.0536 & -0.0041 \\ -0.1027 & 0.3158 & 1.0000 & 0.3194 & -0.4410 & -0.0102 \\ -0.0616 & 0.0899 & 0.3194 & 1.0000 & -0.0692 & -0.0058 \\ 0.2154 & -0.0536 & -0.4410 & -0.0692 & 1.0000 & 0.0215 \\ 0.0106 & -0.0041 & -0.0102 & -0.0058 & 0.0215 & 1.0000 \end{bmatrix},$$

where the correlation coefficients c_{ij} are defined by

$$c_{ij}=\frac{\sum_{x,y}(s_i(x,y)-\overline{s}_i)(s_j(x,y)-\overline{s}_j)}{N\sigma_i\sigma_j}$$
$$\overline{s}_i=\frac{1}{N}\sum_{x,y}s_i(x,y)$$
$$\sigma_i^2=\frac{1}{N}\sum_{x,y}(s_i(x,y)-\overline{s}_i)^2.$$

It is shown by the above source correlation matrix that each natural image is correlated with one or more other natural images.

Mixing the image sources by the following matrix:

$$\boldsymbol{A} = \begin{bmatrix} 1.0439 & 1.0943 & 0.9927 & 1.0955 & 1.0373 & 0.9722 \\ 1.0656 & 0.9655 & 0.9406 & 1.0478 & 1.0349 & 1.0965 \\ 1.0913 & 0.9555 & 0.9498 & 0.9268 & 0.9377 & 0.9059 \\ 1.0232 & 1.0143 & 0.9684 & 1.0440 & 1.0926 & 1.0569 \\ 0.9745 & 0.9533 & 1.0165 & 1.0891 & 1.0751 & 1.0321 \\ 0.9114 & 0.9026 & 1.0283 & 0.9928 & 1.0711 & 1.0380 \end{bmatrix},$$

we obtain six mixed pixel sequences and their images are shown in the second row in Figure 3. The mixed images are highly correlated with the correlation coefficient matrix

$$\boldsymbol{R}_m = \begin{bmatrix} 1.0000 & 0.9966 & 0.9988 & 0.9982 & 0.9985 & 0.9968 \\ 0.9966 & 1.0000 & 0.9963 & 0.9995 & 0.9989 & 0.9989 \\ 0.9988 & 0.9963 & 1.0000 & 0.9972 & 0.9968 & 0.9950 \\ 0.9982 & 0.9995 & 0.9972 & 1.0000 & 0.9996 & 0.9993 \\ 0.9985 & 0.9989 & 0.9968 & 0.9996 & 1.0000 & 0.9996 \\ 0.9968 & 0.9989 & 0.9950 & 0.9993 & 0.9996 & 1.0000 \end{bmatrix}.$$

To compute the de-mixing matrix, we apply the algorithm (A') and use only 20% of the mixed data. The separation result is shown in the third row in Figure 3. If the outputs are arranged in the same order as the sources, the correlation coefficient matrix of the output is

$$\boldsymbol{R}_o = \begin{bmatrix} 1.0000 & -0.0041 & -0.1124 & -0.1774 & 0.0663 & 0.1785 \\ -0.0041 & 1.0000 & 0.1469 & 0.0344 & -0.1446 & -0.0271 \\ -0.1124 & 0.1469 & 1.0000 & 0.1160 & -0.1683 & -0.0523 \\ -0.1774 & 0.0344 & 0.1160 & 1.0000 & -0.1258 & 0.0946 \\ 0.0663 & -0.1446 & -0.1683 & -0.1258 & 1.0000 & -0.1274 \\ 0.1785 & -0.0271 & -0.0523 & 0.0946 & -0.1274 & 1.0000 \end{bmatrix}.$$

Applying the algorithm (A''), we obtain a separation result similar to that in Figure 3. It is not strange that dependent sources can sometimes be separated by the algorithms (A') and (A'') since the MI of the mixtures is usually much higher than the MI of the sources. Applying the algorithms (A') or (A''), the MI of the transformed mixture is decreased. When it reaches a level similar to that of the sources, the dependent sources can be extracted. In this case, some prior knowledge about sources (e.g. human face and speech) are needed to select the separation results.

7 Conclusion

The blind separation algorithm based on the ME approach is concise and often very effective in practice. But it is not yet fully justified. The MMI approach, on the other hand, is based on the contrast function MI which is well justified. We have studied the relation between the ME and MMI and prove that the ME will not update the de-mixing matrix in the directions of increasing the cross-talking at the solution points when all sources are zero mean signals. When one of sources has a non-zero mean, the entropy is not maximized in general at the solution points. It is suggested by this result that the mixture model should be reformulated such that all sources have a zero mean in order to use the ME approach.

The two basic MMI algorithms (A') and (A'') have been derived based on the minimization of the MI of the outputs. The contrast function MI is impractical unless we approximate it.

The difficulty is to evaluate the marginal entropy of the outputs. The Gram-Charlier expansion and the Edgeworth expansion are applied to estimate the MI. The natural gradient method is used to minimize the estimated MI to obtain the basic algorithm. These algorithms have been tested for separating unknown source signals mixed by a mixing matrix randomly chosen, and the validity of the algorithms has been verified.

Because of the approximation error, the MMI algorithms have limitations and they cannot replace the ME. There are cases such as the experiment in (Bell and Sejnowski 1995a) using speech signals in which the ME algorithm performs better.

Although the ME and the MMI result in very similar algorithms, it is unknown whether the two approaches are equivalent globally. The activation functions in (A) and the cumulants in (A') should be determined adequately. We proposed an adaptive algorithms to estimate these cumulants. It is observed from many simulations that the adaptive on-line algorithms (A') and (A'') need less samples than the on-line algorithm (A) (with non-linear functions such as $2\tanh(x)$ and x^3) to reach the separation.

The test of the algorithms (A') and (A'') using the image data suggests that even dependent sources can sometimes be separated if some prior knowledge is used to select the separation results. The reason behind this is that the MI of the mixtures is usually higher than that of sources and the algorithms (A') and (A'') can decrease the MI from a high level to a lower level.

8 Appendices

8.1 Appendix A: proof of Lemma 1

Let $\|\boldsymbol{B}\| \leq \varepsilon < 1$, then $|\boldsymbol{I}+\boldsymbol{B}|^{-1} = 1 - \mathrm{tr}(\boldsymbol{B}) + O(\varepsilon^2)$ and $|\boldsymbol{I}+\boldsymbol{B}|^{-1} = \boldsymbol{I} - \boldsymbol{B} + O(\varepsilon^2)$. So

$$\begin{aligned}
p(\boldsymbol{y};\boldsymbol{W}) &= |\boldsymbol{I}+\boldsymbol{B}|^{-1} r((\boldsymbol{I}+\boldsymbol{B})^{-1}\boldsymbol{y}) \\
&= (1-\mathrm{tr}(\boldsymbol{B}))\prod_{a=1}^{n} r_a(y_a - \sum_c B_{ac}y_c + O(\varepsilon^2)) \\
&= (1-\mathrm{tr}(\boldsymbol{B}))[\prod_{a=1}^{n} r_a(y_a) - \sum_a r'_a(y_a)\prod_{b\neq a}^{n} r_b(y_b)\sum_c B_{ac}y_c] + O(\varepsilon^2) \\
&= r(\boldsymbol{y}) - [\mathrm{tr}(\boldsymbol{B}) + \sum_{a,c} l'_a(y_a)B_{ac}y_c]r(\boldsymbol{y}) + O(\varepsilon^2) \qquad (32)
\end{aligned}$$

where $l'_a(y_a) = \frac{r'_a(y_a)}{r_a(y_a)}$.

From the expression (32), we have the linear expansion of $C(\boldsymbol{W}) = C((\boldsymbol{I}+\boldsymbol{B})\boldsymbol{A}^{-1})$ at $\boldsymbol{B}=0$

$$C((\boldsymbol{I}+\boldsymbol{B})\boldsymbol{A}^{-1}) \approx -\sum_a \int d\boldsymbol{y}[\mathrm{tr}(\boldsymbol{B}) + \sum_{b,c} l'_b(y_b)B_{bc}y_c]r(\boldsymbol{y})\log k_a(y_a)$$

from which we calculate $\frac{\partial C}{\partial \boldsymbol{B}}$ at $\boldsymbol{B}=0$. When $b \neq c$,

$$\begin{aligned}
\frac{\partial C}{\partial B_{bc}} &= -\sum_a \int d\boldsymbol{y} y_c r(\boldsymbol{y}) l'_b(y_b)\log k_a(y_a) \\
&= -\sum_{a\neq c, a\neq b}(\int dy_c\ y_c r(y_c))(\int dy_b\ r'_b(y_b))D[r_a(y_a)\|g'_a(y_a)] \\
&\quad -\int d\boldsymbol{y} y_c r_c(y_c)(\prod_{i\neq c} r_i(y_i))l'_b(y_b)\log k_c(y_c)
\end{aligned}$$

$$
\begin{aligned}
&-(\int dy_c\ y_c r(y_c))(\int dy_b\ r_b'(y_b) r_b(y_b) \log k_b(y_b)) \\
&= -(\int dy_c\ y_c r(y_c))(\int dy_b\ r_b'(y_b) r_b(y_b) \log k_b(y_b))
\end{aligned} \tag{33}
$$

since $\int dy r_b'(y) = 0$.

8.2 Appendix B: Natural Gradient

Denote $\mathrm{Gl}(n) = \{\boldsymbol{X} \in R^{n\times n} : \ \det(\boldsymbol{X}) \neq 0\}$. Let $\boldsymbol{W} \in Gl(n)$ and $\phi(\boldsymbol{W})$ be a scalar function. Before we define the natural gradient of $\phi(\boldsymbol{W})$, we recapitulate the gradient in a Riemannian manifold S in the tensorial form. Let $\boldsymbol{x} = (x^i)$, $i = 1, \cdots, m$, be its (local) coordinates. The gradient of $\phi(\boldsymbol{x})$ is written as

$$\nabla\phi = \left(\frac{\partial\phi}{\partial x^i}\right) = (a_i),$$

which is a linear operator (or a covariant vector) mapping a vector (contravariant vector) $\boldsymbol{X} = (X^i)$ to real values in $\boldsymbol{R}$,

$$\nabla\phi \circ \boldsymbol{X} = \sum a_i X^i.$$

The manifold S has the Riemannian metric $g_{ij}(\boldsymbol{x})$, by which the inner product of $\boldsymbol{X}$ and $\boldsymbol{Y}$ is written as

$$\langle \boldsymbol{X}, \boldsymbol{Y} \rangle = \sum g_{ij} X^i Y^j.$$

Let $\tilde{\nabla}\phi = \tilde{\boldsymbol{a}}$ be the contravariant version of $\boldsymbol{a} = \nabla\phi$, such that

$$\nabla\phi \circ \boldsymbol{X} = \langle \tilde{\nabla}\phi, \boldsymbol{X} \rangle$$

or

$$\sum a_i X^i = \sum g_{ij} \tilde{a}^i X^j.$$

So we have

$$\tilde{a}^i = \sum g^{ij} a_j,$$

where (g^{ij}) is the inverse matrix of (g_{ij}).

It is possible to give a more direct meaning to $\tilde{a}^i$. We search for the steepest direction of $\phi(\boldsymbol{x})$ at point $\boldsymbol{x}$. Let $\varepsilon > 0$ be a small constant, and we study the direction $\boldsymbol{d}$ such that

$$\tilde{\boldsymbol{d}} = \mathrm{argmax}\ \phi(\boldsymbol{x} + \varepsilon\boldsymbol{d}),$$

under the constraint that $\boldsymbol{d}$ is a unit vector,

$$\sum g_{ij} d^i d^j = 1.$$

Then it is easy to calculate that

$$\tilde{d}^i = \tilde{a}^i.$$

It is natural to define the gradient flow in S by

$$\dot{\boldsymbol{x}} = -\eta\tilde{\nabla}\phi = -\eta \sum g^{ij} \frac{\partial}{\partial x^j} \phi(\boldsymbol{x}).$$

Now we return to our manifold $Gl(n)$ of matrices. We define the natural gradient of $\phi(\boldsymbol{W})$ by imposing a Riemannian structure in $Gl(n)$. The manifold $Gl(n)$ of matrices has the Lie group structure : any $\boldsymbol{A} \in Gl(n)$ maps $Gl(n)$ to $Gl(n)$ by $\boldsymbol{W} \rightarrow \boldsymbol{W}\boldsymbol{A}$, where $\boldsymbol{A} = \boldsymbol{E}$ (unit matrix) is the unit. We impose that the Riemannian structure should be invariant by this operation $\boldsymbol{A}$.

More definitely, at $\boldsymbol{W}$, we consider a small deviation of $\boldsymbol{W}$, $\boldsymbol{W}+\varepsilon\boldsymbol{Z}$, where ε is infinitesimally small. Here, $\boldsymbol{Z}$ is the tangent vector at $\boldsymbol{W}$ (or an element of the Lie algebra). We introduce an inner product

$$\langle \boldsymbol{Z}, \boldsymbol{Z}\rangle_{\boldsymbol{W}}$$

at $\boldsymbol{W}$. Now, $\boldsymbol{W}$ is mapped to the unit matrix $\boldsymbol{E}$ by the operation of $\boldsymbol{A}=\boldsymbol{W}^{-1}$. Then, $\boldsymbol{W}$ and $\boldsymbol{W}+\varepsilon\boldsymbol{Z}$ are mapped to $\boldsymbol{W}\boldsymbol{A}=\boldsymbol{W}\boldsymbol{W}^{-1}=\boldsymbol{E}$ and $(\boldsymbol{W}+\varepsilon\boldsymbol{Z})\boldsymbol{W}^{-1}=\boldsymbol{E}+\varepsilon\boldsymbol{Z}\boldsymbol{W}^{-1}$, respectively. So the tangent vector $\boldsymbol{Z}$ at $\boldsymbol{W}$ corresponds to the tangent vector $\boldsymbol{Z}\boldsymbol{W}^{-1}$ at $\boldsymbol{E}$. They should have the same length (the Lie-group invariance : the same as the invariance under the basis change of vectors on which a matrix acts). So

$$\langle \boldsymbol{Z}, \boldsymbol{Z}\rangle_{\boldsymbol{W}} = \langle \boldsymbol{Z}\boldsymbol{W}^{-1}, \boldsymbol{Z}\boldsymbol{W}^{-1}\rangle_{\boldsymbol{E}}.$$

It is very natural to define the inner product at $\boldsymbol{E}$ by

$$\langle \boldsymbol{Y}, \boldsymbol{Y}\rangle_{\boldsymbol{E}} = \mathrm{tr}(\boldsymbol{Y}^T\boldsymbol{Y}) = \sum (Y_{ij})^2,$$

because we have no specific preference in the components at $\boldsymbol{E}$. Then, we have

$$\langle \boldsymbol{Z}, \boldsymbol{Z}\rangle_{\boldsymbol{W}} = \langle \boldsymbol{Z}\boldsymbol{W}^{-1}, \boldsymbol{Z}\boldsymbol{W}^{-1}\rangle_{\boldsymbol{E}} = \mathrm{tr}(\boldsymbol{W}^{-T}\boldsymbol{Z}^T\boldsymbol{Z}\boldsymbol{W}^{-1}).$$

On the other hand, the gradient operator $\partial\phi$ is defined by

$$\begin{aligned}\phi(\boldsymbol{W}+\varepsilon\boldsymbol{Z}) &= \phi(\boldsymbol{W}) + \varepsilon\partial\phi\circ\boldsymbol{Z},\\ \partial\phi\circ\boldsymbol{Z} &= \mathrm{tr}(\partial\phi^T\boldsymbol{Z}) = \sum \frac{\partial\phi}{\partial W_{ij}} Z_{ij}.\end{aligned}$$

Hence, the contravariant version $\tilde{\partial}\phi$ is

$$\begin{aligned}\partial\phi\circ\boldsymbol{Z} &= \langle\tilde{\partial}\phi, \boldsymbol{Z}\rangle_{\boldsymbol{W}}\\ &= \mathrm{tr}(\boldsymbol{W}^{-T}\tilde{\partial}\phi^T\boldsymbol{Z}\boldsymbol{W}^{-1})\\ &= \mathrm{tr}(\boldsymbol{W}^{-1}\boldsymbol{W}^{-T}\tilde{\partial}\phi^T\boldsymbol{Z}),\end{aligned}$$

giving

$$\partial\phi^T = \boldsymbol{W}^{-1}\boldsymbol{W}^{-T}\tilde{\partial}\phi^T$$

or

$$\partial\phi = \tilde{\partial}\phi\boldsymbol{W}^{-1}\boldsymbol{W}^{-T}$$

i.e.

$$\tilde{\partial}\phi = \partial\phi\boldsymbol{W}^T\boldsymbol{W}.$$

Hence, the natural gradient should be

$$\frac{d\boldsymbol{W}}{dt} = -\eta(\nabla\phi)\boldsymbol{W}^T\boldsymbol{W}.$$

Note the natural gradient $(\nabla\phi)\boldsymbol{W}^T\boldsymbol{W}$ is exactly the same as the relative gradient introduced in (Cardoso and Laheld 1996). Using the natural gradient or the relative gradient leads to the blind separation algorithms with the equivariant property, i.e., the performance of the algorithms is independent from the scaling of the sources.

Acknowledgment

The authors wish to thank the reviewers for their constructive comments on the manuscript and to acknowledge the useful discussions with J.-F. Cardoso and A. Cichocki.

References

[1] Amari, S. 1967. A theory of adaptive pattern classifiers. *IEEE Trans. on Electronic Computers*, EC.16(3):299–307, June.

[2] Amari, S. 1993. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5:185–196.

[3] Amari, S. 1997. Neural learning in structured parameter spaces – natural Riemannian gradient. In *Advances in Neural Information Processing Systems, 9, MIT Press: Cambridge, MA.*

[4] Amari, S., Cichocki, A., and Yang, H. H. 1996. A new learning algorithm for blind signal separation. In *Advances in Neural Information Processing Systems, 8, eds. David S. Touretzky, Michael C. Mozer and Michael E. Hasselmo, MIT Press: Cambridge, MA.*, pages 757–763.

[5] Bell, A. J., and Sejnowski, T. J. 1995. An information-maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159.

[6] Bell, A. J., and Sejnowski, T. J. 1995. Fast blind separation based on information theory. In *Proceedings 1995 International Symposium on Nonlinear Theory and Applications*, volume I, pages 43–47, December.

[7] Cardoso, J.-F., and Laheld, B. 1996. Equivariant adaptive source separation. *IEEE Trans. on Signal Processing*, 44(12):3017–3030, December.

[8] Cichocki, A., Unbehauen, R., Moszczyński, L., and Rummert, E. 1994. A new on-line adaptive learning algorithm for blind separation of source signals. In *ISANN94*, pages 406–411, Taiwan, December.

[9] Comon, P. 1994. Independent component analysis, a new concept? *Signal Processing*, 36:287–314.

[10] Deco, G., and Brauer, W. 1996. Nonlinear higher-order statistical decorrelation by volumme-conserving neural architectures. *Neural Networks*, 8(4):525–535.

[11] Barlow, H. B., and Földiák, P. 1989. Adaptation and decorrelation in the cortex. In C. Miall, R. M. Durbin, and G. J. Mitchison, editors, *The computing neuron*, pages 54–72. Addison-Wesley, New York.

[12] Nadal, J. P., and Parga, N. 1994. Nonlinear neurons in the low noise limit: a factorial code maximizes information transfer. *Network*, 5:561–581.

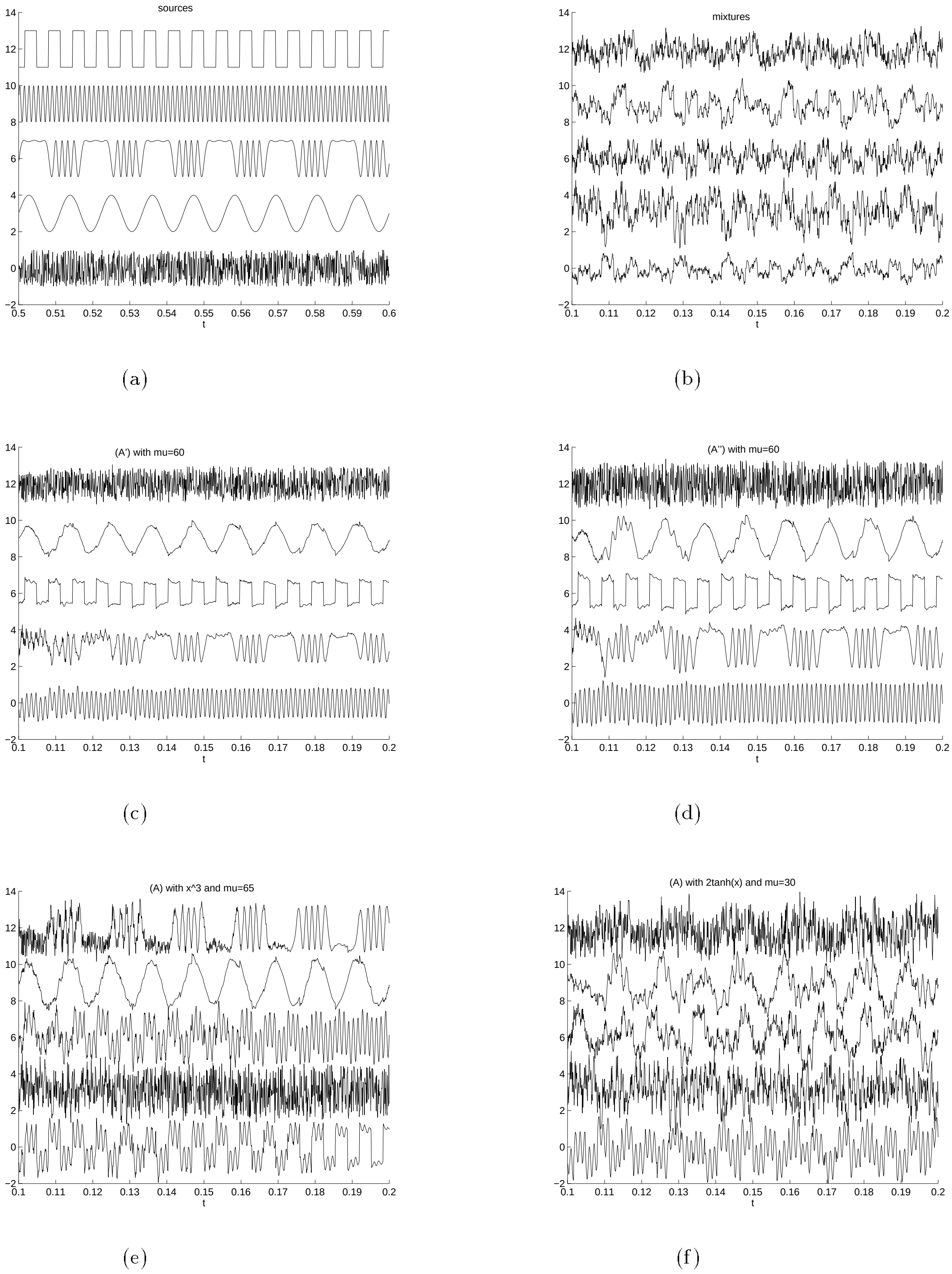
[13] Stuart, A., and Ord, J. K. 1994. *Kendall's Advanced Theory of Statistics.* Edward Arnold.

Figures

Figure 1: The comparison of the separation by the algorithms (A), (A') and (A''):
(a) the sources;
(b) the mixtures;
(c) the separation by (A') using learning rate mu=60;
(d) the separation by (A'') using learning rate mu=60;
(e) the separation by (A) using x^3 and mu=65;
(f) the separation by (A) using 2tanh(x) and mu=30.

Figure 2: Comparison of the performances of (A) (using x^3 or 2tanh(x)), and (A') and (A'').

Figure 3: Separation of mixed images: the images in the first row are sources; those in the second row are the mixed images; and those in the third row are separated images.



(a) (b)

(c) (d)

(e) (f)

Figure 1:

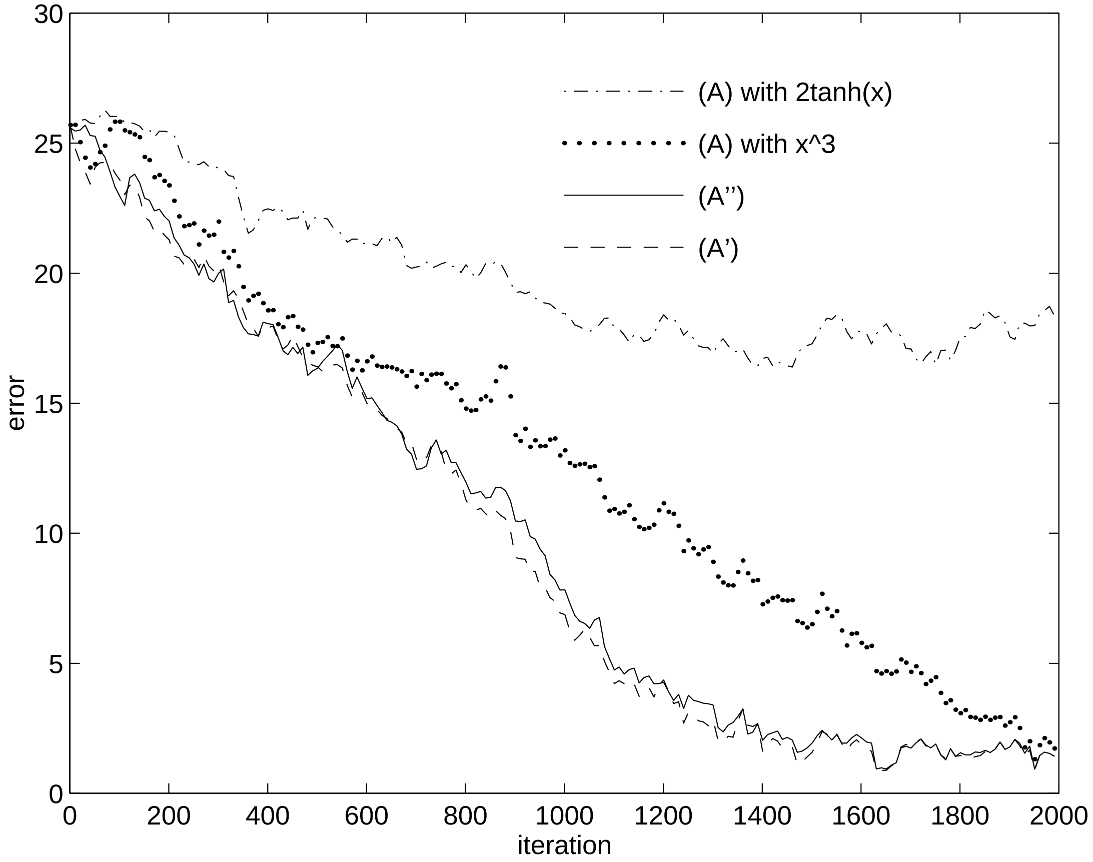

30
25
20
15
10
5
0
error
0
200
400
600
800
1000
1200
1400
1600
1800
2000
iteration
(A) with 2tanh(x)
(A) with x^3
(A'')
(A')

Figure 2:

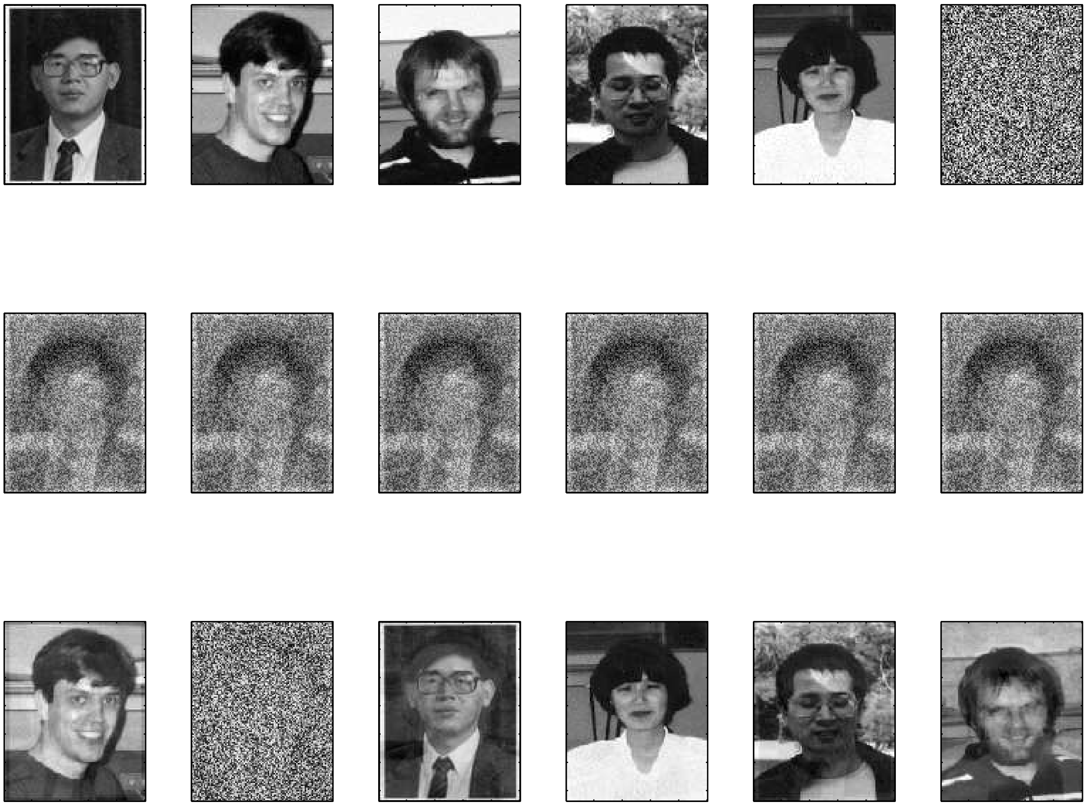


Figure 3: