BLIND DECONVOLUTION OF TIMELY CORRELATED SOURCES BY GRADIENT DESCENT SEARCH

Włodzimierz Kasprzak

Warsaw University of Technology
Institute of Control and Computation Engineering
ul. Nowowiejska 15-19, PL - 00-665 Warsaw
POLAND
E-mail: *W.Kasprzak@ia.pw.edu.pl*

Abstract. In multichannel blind deconvolution (MBD) the goal is to calculate possibly scaled and delayed estimates of source signals from their convoluted mixtures, using approximate knowledge of the source characteristics only. Nearly all of the solutions to MBD proposed so far require from source signals to be pair-wise statistically independent and to be timely not correlated. In practice, this can only be satisfied by specific synthetic signals. In this paper we describe how to modify gradient-based iterative algorithms in order to perform the MBD task on timely correlated sources. Implementation issues are discussed and specific tests on synthetic and real 2-D images are documented.

Key words. Blind source deconvolution, gradient descent search, speech and image processing, unsupervised learning.

1 Introduction

In speech and image recognition systems (which are based on the pattern recognition theory) [DUD73] the first processing step is to preprocess the sensor data in such a way, that the useful original source patterns are extracted without noise and disruption [GON87]. The applied methods give satisfactory results if the characteristics of disruption or source characteristics are properly predicted. But methods are needed, that can automatically adjust to the sensor signal [NIE90]. One possible way of seeking general solutions to source extraction lies in the increase of sensors (e.g. microphones or camera views) and the processing of a combined sensor signals (containing given pattern) at the same time.

Instead of a single scalar sample we analyse vector samples in a multi-dimensional space. There exist well-known methods of low-level vector signal processing, which perform transformation of the representation space [WID85, CIA02]:

- **decorrelation** in Cartesian space,
- **PCA, PSA** - principal component (or subspace) analysis;
- **ICA** - independent component analysis.

The decorrelation process is usually a prelimi-

nary step for some ICA algorithms, whereas the PCA and PSA are inherently linked with signal compression. In the context of source extraction they are applied for noise cancellation (noise corresponds to minor components) or ordered sequential source extraction - after a previous whitening of the sensor signals.

One of approaches to the ICA constitutes adaptive methods of *blind source separation* (**BSS**) [HYV96, KAR97, KAS97, CIA02]. They are applied to instantaneous mixtures of sources. An extension of BSS for convolved mixtures (mixtures in space and time) is called **BSD** (*blind source deconvolution*) [AMA97, CIA02, HUA96]. In multichannel blind deconvolution (MBD) the goal is to calculate possibly scaled and delayed estimates of source signals from their convoluted mixtures, using approximate knowledge of the source characteristics only. Nearly all of the solutions to MBD proposed so far require from source signals to be pair-wise statistically independent and timely not correlated. In practice, this can only be satisfied by specific synthetic signals.

In this paper we first describe different iterative update rules that can perform the MBD task. Then we discuss how to modify the update rules in order to adapt to the self-correlation of sources. Implementation issues are discussed and specific tests on synthetic and real 2-D images are documented.

The paper is organised into following five main sections. In section 2 the role of blind signal processing techniques is discussed. In section 3 the theoretical MBD problem is introduced. Then, in section 4, the class of search-based methods is described, and three specific search algorithms are reviewed, that can be implemented as learning algorithms of artificial neural networks. The tests on different signal and image data are documented in section 5. The conclusions are drawn in final section.

2 The role of MBD techniques

The main application fields of the blind separation/deconvolution techniques can be grouped according to the nature of the signal, i.e. a 1-D signal or a 2-D image. In the processing of 1-D signals we distinguish:

- Bio-medical applications: the extraction of nervous signals in muscles and inner organs [KAR97].
- Speech processing the extraction of selected speaker in the "cocktail party" problem [CIA02].
- Seismology the extraction of signals originating in different earth layers.
- General data mining in different fields (e.g. a prospective application might be the separation of economic processes in overall economic data).

The pre-processing of 2-D images may include:

- The extraction of sparse binary images [PAJ96] (e.g. images of documents) (Fig. 1).
- Contrast strengthening of smoothed images in selected regions [CIA02].
- Encryption of transmitted images [KAS96] (Fig. 2).

We explain the source deconvolution approach using the exemplary "cocktail party" problem. Let sounds are recorded in a typical room using an array of microphones. Each microphone will receive a mixture of (*convolved and delayed*) copies of the sound sources (the propagation delay is based on the location of both the sources and the microphone). Since we have no prior knowledge of the source mixing and distortion process, we call the solution: the **multichannel blind deconvolution** approach to the *cocktail party* problem. In a simplified solution to the

Figure 1: Blind extraction of three sparse sources (binary images) from their two mixtures.

cocktail party problem - in **blind source separation (BSS)** we make the fundamental assumption that the signals are mixed together **instantaneously**, meaning that all of the signals are time-aligned so that they enter the sensors simultaneously without any delays.

3 The MBD problem

3.1 BSS and MBD

The multichannel **blind source deconvolution** problem **(MBD)** [AMA98, CIA02, HUA96, SAB98] can be considered as a natural extension of the **instantaneous blind source separation** problem **(BSS)**. An m-dimensional vector of sensor signals (in discrete time): $\mathbf{x}(k) = [x_1(k), \ldots, x_m(k)]^T$ at time k, is assumed to be produced from an n-dimensional vector of source signals $\mathbf{s}(k) = [s_1(k), \ldots, s_n(k)]^T$ $(m > n)$, by using the mixture model:

$$\mathbf{x}(k) = \sum_{p=-\infty}^{\infty} \mathbf{H}_p \mathbf{s}(k-p) = \mathbf{H}_p * \mathbf{s}(k) = \mathbf{H}(z)\mathbf{s}(k) \quad (1)$$

where $\mathbf{H}_p$ is an $(m \times n)$ matrix of mixing coefficients at lag p :

$$\mathbf{H}(z) = \sum_{p=-\infty}^{\infty} \mathbf{H}_p z^{-p} \quad (2)$$

$\mathbf{H}(z)$ is a matrix transfer function. z^{-1} is the delay operator defined by: $z^{-p}[s_i(k)] = s_i(k-p)$.

The goal of MBD is to calculate possibly scaled and delayed estimates of the source signals from the received signals by using an approximate knowledge of the source signal distributions and statistics.

Two general approaches to multichannel blind deconvolution can be considered. The first approach focuses on the estimation of the multichannel impulse response sequence $\{\mathbf{H}_p\}$ from the sensor signals $\mathbf{x}(k)$ [HUA96] and it can be called blind identification or equalisation. In the second approach, on tries to estimate the direct sources by a generalisation of the blind source separation approach [AMA97, CIA02].

3.2 A feed-forward model for MBD

Consider a feed-forward model that estimates the source signals directly by using a *truncated version* of a doubly infinite multichannel equalizer:

$$\mathbf{y}(k) = \sum_{p=-\infty}^{\infty} \mathbf{W}_p(k)\mathbf{x}(k-p) = \mathbf{W}_p(k) * \mathbf{x}(k)$$
$$= \mathbf{W}(z,k)[\mathbf{x}(k)] \quad (3)$$

where $\mathbf{y}(k) = [y_1(k), \ldots, y_n(k)]^T$ is an n-dimensional vector of outputs,
$\mathbf{W}(k) = \{\mathbf{W}_p(k) \mid -\infty \leq p \leq \infty\}$ is a sequence of matrices (of size $n \times m$) at time k, and the

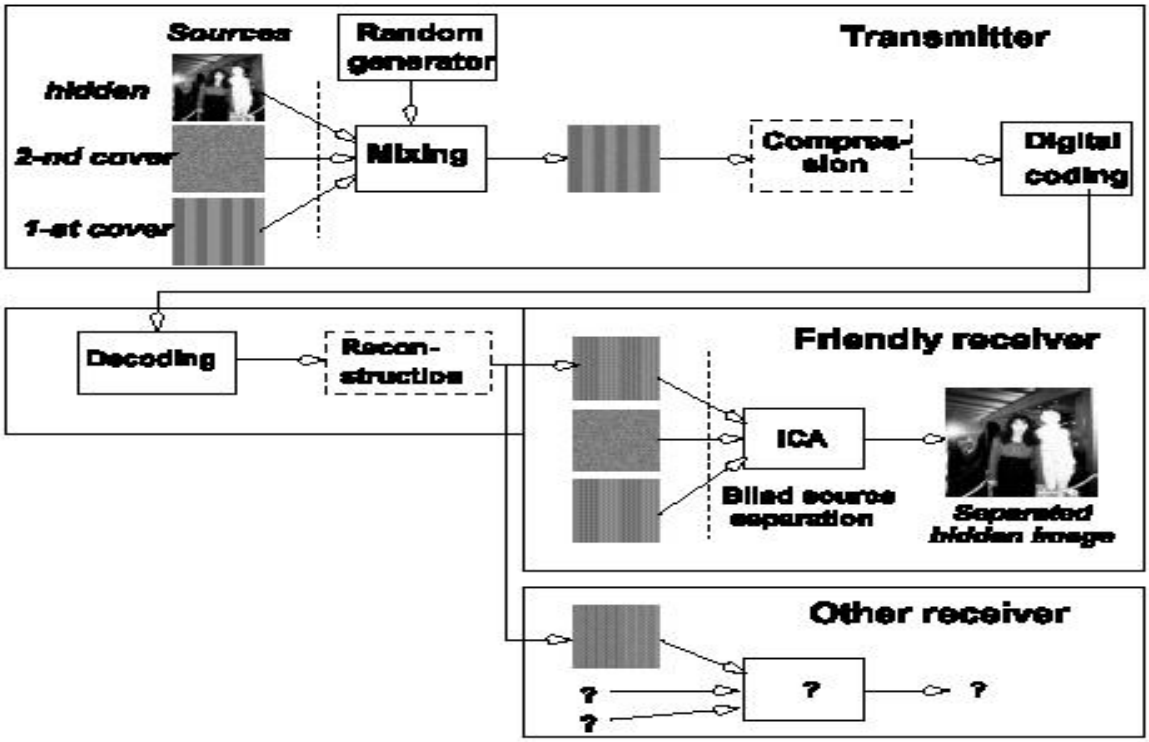

Figure 2: Encryption of transmitted images: only the friendly receiver knows the two cover images [KAS96].

matrix transfer function is

$$\mathbf{W}(z,k) = \sum_{p=-\infty}^{\infty} \mathbf{W}_p(k) z^{-p} \tag{4}$$

In the standard BSD approach we use the neural network models for BSS but the synaptic weights are generalised to **finite-size dynamic filters** (e.g. FIR), i.e. each synaptic weight:

$$\mathbf{W}_{ij}(z,k) = \sum_{p=0}^{L} w_{ijp}(k) z^{-p} \tag{5}$$

is described by a finite-duration impulse response (FIR) filter at time k.

Then the goal of MBD-search is to adjust $\mathbf{W}(z;k)$ such that the global system is:

$$\lim_{k\to\infty} \mathbf{G}(z,k) = \mathbf{W}(z,k)\mathbf{H}(z) = \mathbf{P}_0\mathbf{D}(z) \tag{6}$$

Where $\mathbf{P}_0$ is an $n \times n$ permutation matrix, $\mathbf{D}(z)$ is an $n \times n$ diagonal matrix whose $(i;i)$-th entry is $c_i z^{-\triangle i}$, c_i is a scalar factor, and $\triangle i$ is an integer delay.

General requirements of the MBD theory are:

1. Each source has to be temporally independent, identically distributed.
2. Sources should be drawn from non-Gaussian distributions.
3. The number of sensors is equal or greater than the number of independent sources.
4. The convoluting filters have no common zeroes.
5. The convoluting filters have no zeroes on the unit circle.

The definitions of decorrelation and independence are given next. Let us consider two random variables $x(t)$ and $y(t)$. The following condition holds for decorrelated signals:

$$E\{x \cdot y\} = E\{x\} \cdot E\{y\} \tag{7}$$

If $E\{x \cdot y\} = 0$ then the signals are also orthogonal. The independence condition is defined as:

$$p_{xy}(x, y) = p_x(x) \cdot p_y(y) \tag{8}$$

where p_{xy}, p_x, p_y are pdf's of appropriate distributions xy, x and y.
The decorrelation condition is weaker than independence, i.e. if x and y are independent then they are also decorrelated.

The generalized decorrelation can be defined as:

$$E\{f(x) \cdot g(y)\} = E\{f(x)\} \cdot E\{g(y)\} \tag{9}$$

for arbitrary non-linearitys $f(.)$ and $g(.)$.

Yet another, general definition of independence is:

$$E\{x^p \cdot y^q\} = E\{x^p\} \cdot E\{y^q\} \tag{10}$$

The ICA algorithms force all the statistical cross-moments to zero, making the output signals statistically independent.

4 The gradient search methods for MBD

4.1 Gradient based optimisation

Two well-known iterative optimisation methods are the **stochastic gradient (or gradient descent)** method and the **quasi-Newton** convergence method [DUD73].

In gradient descent search the basic task is to define a criterion $J(\mathbf{W}(z,k))$, which obtains its minimum for some $\mathbf{W}_{opt}$ if this $\mathbf{W}_{opt}$ is the seeked optimum solution.

The gradient descent search

1. Start with some initial point $\mathbf{W}(z, k=0)$ in the multi-dimensional parameter space.
2. Obtain the gradient value $\nabla J(\mathbf{W}(z,k))$.
3. Compute the value $\mathbf{W}(z, k+1)$ by moving from $\mathbf{W}(z,k)$ along the gradient descent, i.e. along - $\nabla J(\mathbf{W}(z,k))$:

   $$\mathbf{W}(z,k{+}1) = \mathbf{W}(z,k)) - \eta(k) \nabla J(\mathbf{W}(z,k)) \tag{11}$$

   where $\eta(k)$ is a positive-valued step scaling coefficient.
4. Test the stability of parameters, i.e. if $|\mathbf{W}(z,k+1) - \mathbf{W}(z,k)| < \theta$ (threshold).

Quasi-Newton search
An alternative solution to gradient descent search is a direct minimisation of the series expansion of $J(\mathbf{W})$ by its first- and second-order derivatives:

$$J(\mathbf{W}) \approx J(\mathbf{W}(z,k)) + \nabla J^T (\mathbf{W} - \mathbf{W}(z,k)) + \\ + \frac{1}{2} (\mathbf{W} - \mathbf{W}(z,k))^T \mathbf{D}(k) (\mathbf{W} - \mathbf{W}(z,k)) \tag{12}$$

where $\mathbf{D}(k)$ is the Laplasjan computed at point $\mathbf{W}(z,k)$.

The minimisation of $J(\mathbf{W})$ appears if:

$$\mathbf{W}(z,k{+}1) = \mathbf{W}(z,k) - \mathbf{D}(k)^{-1} \nabla J(\mathbf{W}(z,k)) \tag{13}$$

Obviously the matrix $\mathbf{D}(k)$ need not to be singular.

Both standard approaches suffer from practical implications. The stochastic gradient shows slow convergence if statistical correlation between signals appear in the parameter update

process. The quasi- Newton methods are often of heavy computational complexity and suffer from numerical problems.

The natural gradient search

The natural gradient search (as proposed by Amari [AMA96]) is a modified gradient descent search, where the standard search direction is modified according to some local Riemannian metric of the (weight) parameter space. This new direction is invariant to the statistical relationships between the model parameters, and it has a convenient form for the MBD task:

$$\nabla J^N(\mathbf{W}(z,k)) = \\ = \nabla J(\mathbf{W}(z,k))\mathbf{W}^T(z^{-l},k)\mathbf{W}(z,k) \quad (14)$$

Gradient methods for detection of function extremes in BSS / MBD

Let us assume a cost function $E\{J(\mathbf{W})\}$, dependent on parameters $\mathbf{W}$. The most popular cost minimisation approach (in many applications) uses the steepest descent form. Additionally for solving the BSS and MBD problems another two gradient approaches were developed recently the natural gradient descent and the dual natural gradient descent.

1. Steepest descent approach:

$$\mathbf{W}(l+1) = \mathbf{W}(l) - \eta(l)\frac{\partial E\{J(\mathbf{W})\}}{\partial \mathbf{W}} \quad (15)$$

or its stochastic version (in which the expected value is replaced by a single observation):

$$\mathbf{W}(k+1) = \mathbf{W}(k) - \eta(k)\frac{\partial J(\mathbf{W}(k))}{\partial \mathbf{W}(k)} \quad (16)$$

2. Natural gradient descent approach

The natural gradient was developed independently by: Amari et al. (which gave a theoretical justification by using the Riemannian space notation) [AMA97], Cichocki et al. (which proposed an algorithm justified by computer simulations) [CIC94] and Cardoso and Laheld (that introduced the relative gradient) [CAR96]:

$$\mathbf{W}(l+1) = \mathbf{W}(l) - \\ -\eta(l)\frac{\partial E\{J(\mathbf{W})\}}{\partial \mathbf{W}}\mathbf{W}^T(l)\mathbf{W}(l) \quad (17)$$

or its stochastic version:

$$\mathbf{W}(k+1) = \mathbf{W}(k) - \\ -\eta(k)\frac{\partial J(\mathbf{W}(k))}{\partial \mathbf{W}(k)}\mathbf{W}^T(k)\mathbf{W}(k) \quad (18)$$

3. Dual natural gradient descent approach.

This approach was introduced by Attick and Redlich:

$$\mathbf{W}(l+1) = \mathbf{W}(l) - \\ -\eta(l)\mathbf{W}(l)[\frac{\partial E\{J(\mathbf{W})\}}{\partial \mathbf{W}}]^T\mathbf{W}(l) \quad (19)$$

or its stochastic version:

$$\mathbf{W}(k+1) = \mathbf{W}(k) - \\ -\eta(k)\mathbf{W}(k)[\frac{\partial J(\mathbf{W}(k))}{\partial \mathbf{W}(k)}]^T\mathbf{W}(k) \quad (20)$$

4.2 The optimisation criterion for standard BSS and MBD

Cost functions for BSS

There exist different theoretical justifications of the BSS, like the Kullback-Leibler divergence minimisation, the information maximisation and the mutual information minimisation. All of them lead to the same cost function:

$$J(\mathbf{W}, y) = -\log\det(\mathbf{W}) - \sum_{i=1}^{n} \log p_i(y_i) \qquad (21)$$

where the $p_i(y_i)$-s are pdf's of signals y_i respectively, $\det(\mathbf{W})$ is the determinant of the matrix $\mathbf{W}$.

Cost functions for MBD

We observe m inputs $x(k) = \{x_i(k)|i = 1, \ldots, m\}$ and m outputs $y(k) = \{y_i(k)|i = 1, \ldots, m\}$ over N time points, i.e. $\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(N)$ and $\mathbf{y}(1), \mathbf{y}(2), \ldots, \mathbf{y}(N)$.

The m-dimensional output of the network with finite order FIR weights is

$$\mathbf{y}(k) = \sum_{p=0}^{L} \mathbf{W}_p(l)\mathbf{x}(k-p) \qquad (22)$$

$\mathbf{W}_p(l)$ is the synaptic weight matrix at the l-th iteration (time of learning) and it represents the connection between vectors $\mathbf{y}(k)$ and $\mathbf{x}(k-p)$. Note that for on-line learning, the index l can be replaced by the time index k. The length of time delay L is much smaller than N.

The FIR filter should be trained such that the joint probability density of $\mathbf{y}$ is:

$$p(\mathbf{y}) = \prod_{i=1}^{m}\prod_{k=1}^{N} \mathbf{r}_i(y_i(k)) \qquad (23)$$

where $\{r_i(.)\}$ are the probability densities of the source signals. In practice we shall replace $r_i(.)$ by a hypothesized density model for sources $q_i(.)$, since the true probability distributions are not known.

In MBD the mostly used criterion is the *Kullback-Leibler* divergence (the loss function), which is a measure of distance between two different probability distributions. This loss function is:

$$J(\mathbf{W}(z,l)) = -\log|\det\mathbf{W}_0| - \\ -\sum_{i=1}^{m}\langle\log q_i(y_i(k))\rangle \qquad (24)$$

In above equation $\langle.\rangle$ represents the time-average, i.e.

$$\langle\log q_i(y_i(k))\rangle = \frac{1}{N}\sum_{k=1}^{N}\log q_i(y_i(k)) \qquad (25)$$

$f(\mathbf{y}(k))$ is a column vector whose components are defined as:

$$f_i(y_i(k)) = -\frac{d[\log q_i(y_i(k))]}{dy_i(k)} \qquad (26)$$

4.3 The derivation of BSS algorithms

The gradient of $E\{J(\mathbf{y}, W)\}$ may be expressed as:

$$\nabla_{\mathbf{W}} J(\mathbf{W}) = \frac{\partial E(J(\mathbf{W}))}{\partial \mathbf{W}} = \\ = -\mathbf{W}^{-T} + \langle f(\mathbf{y})\cdot\mathbf{x}^T\rangle \qquad (27)$$

with the non-linearity defined as:

$$f_i(y_i) = -\frac{\partial E\log p_i(y_i)}{\partial y_i} = -\frac{p_i'(y_i)}{p_i(y_i)} \qquad (28)$$

The standard gradient descent approach leads to the learning rule:

$$\triangle \mathbf{W}(k) = -\eta(k)\frac{\partial J(\mathbf{W}(k))}{\partial \mathbf{W}(k)} =$$

$$= \eta(k)[\mathbf{W}^{-T}(k) - f(\mathbf{y}(k)) \cdot \mathbf{x}^T(k)] \quad (29)$$

The observations of low convergence speed and the inversion of matrix W in each iteration cycle are serious drawbacks of this standard learning rule. An initial whitening of the sensor signals could be computed in order to speed up the convergence of the separation rule.

Applying the natural gradient approach we may derive the learning rule for BSS:

$$\triangle \mathbf{W}(t) = -\eta(t)\frac{\partial E\{(\mathbf{W}(t))\}}{\partial \mathbf{W}(t)} \cdot \mathbf{W}^T(t) \cdot \mathbf{W}(t) =$$

$$= \eta(t)[\mathbf{I} - \langle f(\mathbf{y}(t)) \cdot \mathbf{y}^T(t)\rangle]\mathbf{W}(t) \quad (30)$$

By using a stochastic approximation of the averaging process we obtain the on-line adaptive algorithm:

$$\triangle \mathbf{W}(k) = -\eta(k)\lfloor \mathbf{I} - f(\mathbf{y}(k)) \cdot \mathbf{y}^T(k)\rfloor \mathbf{W}(k) \quad (31)$$

Applying the dual natural gradient descent rule we obtain:

$$\triangle \mathbf{W}(t) = -\eta(t)\mathbf{W}(t)[\frac{\partial E\{J(\mathbf{W}(t))\}}{\partial \mathbf{W}(t)}]^T \cdot \mathbf{W}(t) =$$

$$= \eta(t)[\mathbf{I} - \langle \mathbf{y}(t) \cdot g(\mathbf{y}^T(t))\rangle]\mathbf{W}(t) \quad (32)$$

or its stochastic approximation:

$$\triangle \mathbf{W}(k) = \eta(k)\lfloor \mathbf{I} - \mathbf{y}(k) \cdot g(\mathbf{y}^T(k))\rfloor \mathbf{W}(k) \quad (33)$$

These two natural gradient-based learning rules can be combined together to one general, flexible learning rule [CIA02]:

$$\triangle \mathbf{W}(k) = \eta(k)\lfloor \mathbf{I} - f(\mathbf{y}(k)) \cdot g(\mathbf{y}^T(k))\rfloor \mathbf{W}(k) \quad (34)$$

4.4 Derivation of learning rules for MBD

The standard *Bussgang method* applies the *gradient descent search*, which gives the following weight update (learning) rule:

$$\mathbf{W}_p(k+1) = \mathbf{W}_p(k) + \eta(k) f(\mathbf{y}(k))\mathbf{x}^T(k-p)$$
$$\text{for} \quad \text{lag} \quad p = 0, \ldots, L \quad (35)$$

In their first attempt to MBD Amari et al. [AMA97] derived the following update rule for MBD:

$$\mathbf{W}_p(k+1) = \mathbf{W}_p(k) +$$
$$+\eta(k)[\mathbf{W}_p(k) - f(\mathbf{y}(k)) \cdot \mathbf{u}_p^T(k-p)] \quad (36)$$

in which delayed cross filters are used:

$$\mathbf{u}_p(k) = \sum_{p=0}^{L} \mathbf{W}_p^T(k) \cdot \mathbf{y}(k+p) \quad (37)$$

The number of weight matrices is in practice limited to some finite range: $p = 0, \ldots, L$.

In a more elaborated attempt, while using the natural gradient search, Cichocki and Amari have obtained the following weight update (learning) rule for MBD [CIA02]:

$$\triangle \mathbf{W}_p = -\eta \sum_{q=0}^{p} \frac{\partial J(\mathbf{W}(z))}{\partial \mathbf{X}_q}\mathbf{W}_{p-q} =$$

$$= \eta \sum_{q=0}^{p} (\partial_{0q}\mathbf{I} - \langle \mathbf{f}(\mathbf{y}(k))\mathbf{y}^T(k-q)\rangle)\mathbf{W}_{p-q} \quad (38)$$

for $p = 0, 1, \ldots, L$, where η is the learning rate. The term $(\partial_{0q}\mathbf{I})$ denotes, that:

$$\partial_{0q}\mathbf{I} = \begin{cases} \mathbf{I} & for \quad q = p \\ \mathbf{0} & for \quad q \neq p \end{cases} \quad (39)$$

In particular the learning rule for $\mathbf{W}_0$ is :

$$\triangle \mathbf{W}_0 = \eta(\mathbf{I} - \langle \mathbf{f}(\mathbf{y}(k) \cdot \mathbf{y}^T(k) \rangle) \cdot \mathbf{W}_0 \tag{40}$$

The selection of appropriate function $f(y)$ depends on the sign of the *kurtosis*, which is a 4-th order statistical moment:

$$\kappa_4(x) = m_4 - 3m_2^2 \tag{41}$$

One can choose, for example, $f(y) = y^3$, for *sub-Gaussian* sources (with negative kurtosis) and $f(y) = \tanh(\gamma y) \quad (\gamma > 2)$, for *super-Gaussian* signals (with positive kurtosis).

4.5 The modified natural-gradient update rule

The update rule (38) converges to the equilibrium point, described by the two following equations:

$$E\{\langle f(y_i(k)) \cdot y_i(k) \rangle\} = 1, \quad \forall i \tag{42}$$

and

$$E\{\langle f(y_i(k)) \cdot y_i(k-p) \rangle\} = 0, \quad \forall i$$
$$(p = 1, 2, \ldots, L-1) \tag{43}$$

In other words, the deconvolution system, if succeeded, then it produces statistically independent signals of timely non-correlated structure.

In practice, for most natural signals or images, both goals can only approximately be achieved, as the natural signals are pair-wise not fully independent and they also have not a correlation-free time structure. To deal with natural signals, we need to know their temporal structure. Probably for most signal categories this can only approximately be known. Hence, a vector of "generalized correlation" coefficients is assumed to be available to the weight update rule:

$$\Lambda_p = \frac{E\{\langle f(x(k)) \cdot x(k-p) \rangle\}}{E\{\langle f(x(k)) \cdot x(k) \rangle\}} \tag{44}$$
$$(p = 0, 1, 2, \ldots, L-1)$$

From above coefficients we form an appropriate set of diagonal matrices:

$$\Gamma_p = \begin{bmatrix} \Lambda_p & 0 & \ldots & 0 \\ 0 & \Lambda_p & \ldots & 0 \\ 0 & \ldots & \Lambda_p & 0 \\ 0 & 0 & \ldots & \Lambda_p \end{bmatrix} \tag{45}$$
$$(p = 0, 1, 2, \ldots, L-1)$$

The modification of update rule (4.28) takes the form:

$$\triangle \mathbf{W}_p(k) =$$
$$= \eta(k) \sum_{q=0}^{p} [\mathbf{\Gamma}_q - f(\mathbf{y}(k)) \cdot \mathbf{y}^T(k-q))] \mathbf{W}_{p-q}(k) \tag{46}$$

5 Experimental results

5.1 Practical issues

Usually the initial step value should be selected appropriately to the variance of input signals. We observed that the best learning results were achieved for a descending step size. Usually, the choice of the initial weight values affects the initial convergence of the search process. Obviously the weight matrices need not to be zero-valued. We selected initial values of the weights from the range $\langle -0.1, 0.1 \rangle$.

5.2 Measuring the quality of outputs

For test purpose we assume that the source signals *are known* to the observer. Hence, for each pair (output Y_i ; source S_j) with amplitudes

scaled to $< -1, 1 >$ one computes the $SNR[i,j]$ (signal to noise ratio):

$$SNR[i,j] = 10 \cdot \log \frac{\langle S_i^2 \rangle}{MSE[i,j]} \tag{47}$$

where $\langle S_i^2 \rangle$ is the time-average of 2-nd power of source i (i.e. the variance) and $MSE[i,j]$ is the mean square error of approximating S_j by Y_i , i.e.

$$MSE[i,j] = \frac{1}{N} \sum_{k=0}^{N-1} [S_i(k) - Y_j(k)]^2 = \\ = \langle (S_i - Y_j)^2 \rangle \tag{48}$$

with N - number of samples in one signal.

All the individual SNR-s are combined to a matrix $\mathbf{P} = [p_{i,j}]$ with $p_{i,j} = (\sqrt{MSE[i,j]})^{-1}$.

Now a combined error index $EI(\mathbf{P})$ can be computed for the whole separated output set relative to the source set:

$$EI(\mathbf{P}) = \frac{1}{m}[(\sum_{i=1}^{n} \sum_{j=1}^{m} \frac{p_{ij}^2}{\max_i(p_{i,j}^2)}) - n] + \\ + \frac{1}{n}[(\sum_{j=1}^{m} \sum_{i=1}^{n} \frac{p_{ij}^2}{\max_j(p_{i,j}^2)}) - m] \tag{49}$$

The $p_{i,j}$-s are entries of two normalised versions of matrix $\mathbf{P}$. At first, each nonzero row ($i = 1, \ldots, n$) of $\mathbf{P}$ is normalised, such that $max_i(p_{i,j}) = 1$. This allows the computation of first part of above equation (5.3). At second, the columns of matrix $\mathbf{P}$ are normalized in a similar way and the second part of (5.3) is computed.
In the ideal case of perfect separation the following appears:

- $\mathbf{P}$ becomes a permutation matrix.
- Only one element in each row and column equals to unity, and all the other elements are zero.
- EI achieves its minimum possible value equal to 0.

Without the knowledge of original sources we still are able to judge about the separation quality by computing the correlation factors between outputs. The normalized *cross-correlation* between each pair of output signals s_i and s_j is:

$$\frac{E\{s_i \cdot s_j\}}{\sigma_i \cdot \sigma_j} \tag{50}$$

During the learning process the outputs should get less correlated. Initially the mixture signals are highly correlated (they are to some amount stochastically dependent).

5.3 The sources

We have applied two sets of sources: artificial binary images and natural face images (Fig. 3). In Table 1 the results of the normalisation of sources is documented: they all are of zero-mean and of negative kurtosis.
From Table 2 it is evident that the sources are highly self-correlated in time. Hence, we expect that the standard MBD rule (38) will not work properly on convolution mixtures of such sources.

5.4 Some test results

In our experiments three sources (assumed to be unknown to the deconvolution system) ((a) the synthetic binary images or (b) the natural face images) were convolved by using a set of mixing matrices $\{\mathbf{A}_q | q = 0, \ldots, M-1\}$, where each matrix $\mathbf{A}_q$ was of size: $m = 3, n = 3$. In practice we set $M = 4$ (see Fig. 4).

The deconvolution system consisted of three FIR filters of order (up to) $L = 8$, i.e. the number L of delay units were in the range from 1 to 8.

The two natural-gradient based update rules the original one (38) and the modified one (38)

Table 1: Cross-correlation between source pairs.

	mean	var	κ_4
Binary 1	0.0	0.9256	-1.99913
Binary 2	0.0	0.8005	-1.9579
Binary 3	0.0	0.8003	-1.9579

	mean	var	κ_4
Face 1	0.0	0.2790	-0.9108
Face 2	0.0	0.2271	-1.4727
Face 3	0.0	0.1077	-0.6037

Source I - Source J $\frac{E\{s_I s_J\}}{\sigma_I \sigma_J}$	1	2	3
Binary 1	1.0	0.063	0.012
Binary 2		1.0	0.031
Binary 3			1.0

Source I - Source J $\frac{E\{s_I s_J\}}{\sigma_I \sigma_J}$	1	2	3
Face 1	1.0	-0.3045	0.1719
Face 2		1.0	0.1727
Face 3			1.0

Table 2: Auto-correlation of sources in time (p is time delay).

$\frac{E\{f[s_i(k)]s_j(k-p)\}}{\sigma_i \sigma_j}$	p=0	1	2	3	4	5	6	7	8	9
Binary 1	1.0	0.924	0.853	0.783	0.712	0.641	0.568	0.494	0.420	0.345
Binary 2	1.0	0.939	0.881	0.824	0.766	0.710	0.653	0.596	0.539	0.483
Binary 3	1.0	0.939	0.881	0.825	0.768	0.711	0.655	0.598	0.540	0.482

$\frac{E\{f[s_i(k)]s_j(k-p)\}}{\sigma_i \sigma_j}$	p=0	1	2	3	4	5	6	7	8	9
Face 1	1.0	0.986	0.971	0.956	0.942	0.927	0.913	0.898	0.883	0.869
Face 2	1.0	0.936	0.884	0.841	0.806	0.780	0.772	0.771	0.769	0.767
Face 3	1.0	0.894	0.764	0.743	0.813	0.866	0.823	0.733	0.692	0.706

Table 3: Approximation of the auto-correlation of sources in time.

Norm. time-correlation	p=0	1	2	3	4	5	6	7	8	9
Binary images	1.0	0.93	0.88	0.80	0.75	0.70	0.65	0.59	0.50	0.40
Face images	1.0	0.93	0.88	0.85	0.85	0.85	0.83	0.78	0.76	0.76

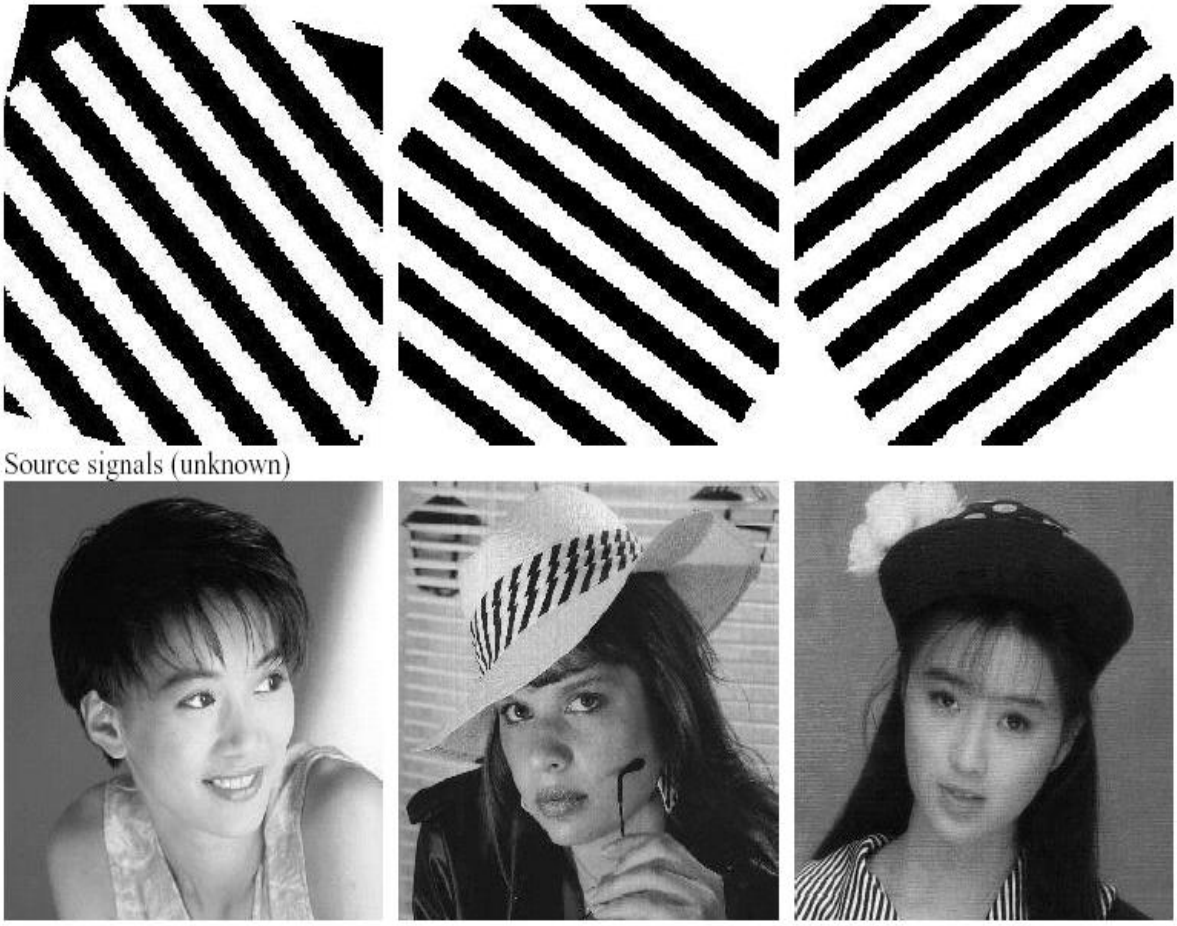

Figure 3: Sources - three synthetic binary images or three natural images.

for timely-correlated sources were tested on these two source sets (see Fig. 5).

The modified rule (38) requires an estimate for timely self-correlation of sources. We have approximated these estimates in the same way for all sources in given set. The applied parameter values are given in Table 3.

The quality index EI was computed for the whole range of deconvolving weight matrices $\mathbf{W}_0 - \mathbf{W}_7$ (see Table 4). The quality results behave like expected the original rule can not cope with clearly timecorrelated sources, whereas the modified rule is able to cope with these problems.

In the first case the best results were achieved after using only the weights $\mathbf{W}_0$, with additional weights corresponding to decorrelation in time, the results are steadily degrading.

In the modified rule case an improvement of the total output results is observed if additional matrices $\mathbf{W}_0 - \mathbf{W}_4$ are learned. In the case of face images after we add even more matrices, the quality is slightly decreasing again (i.e. EI is increasing).

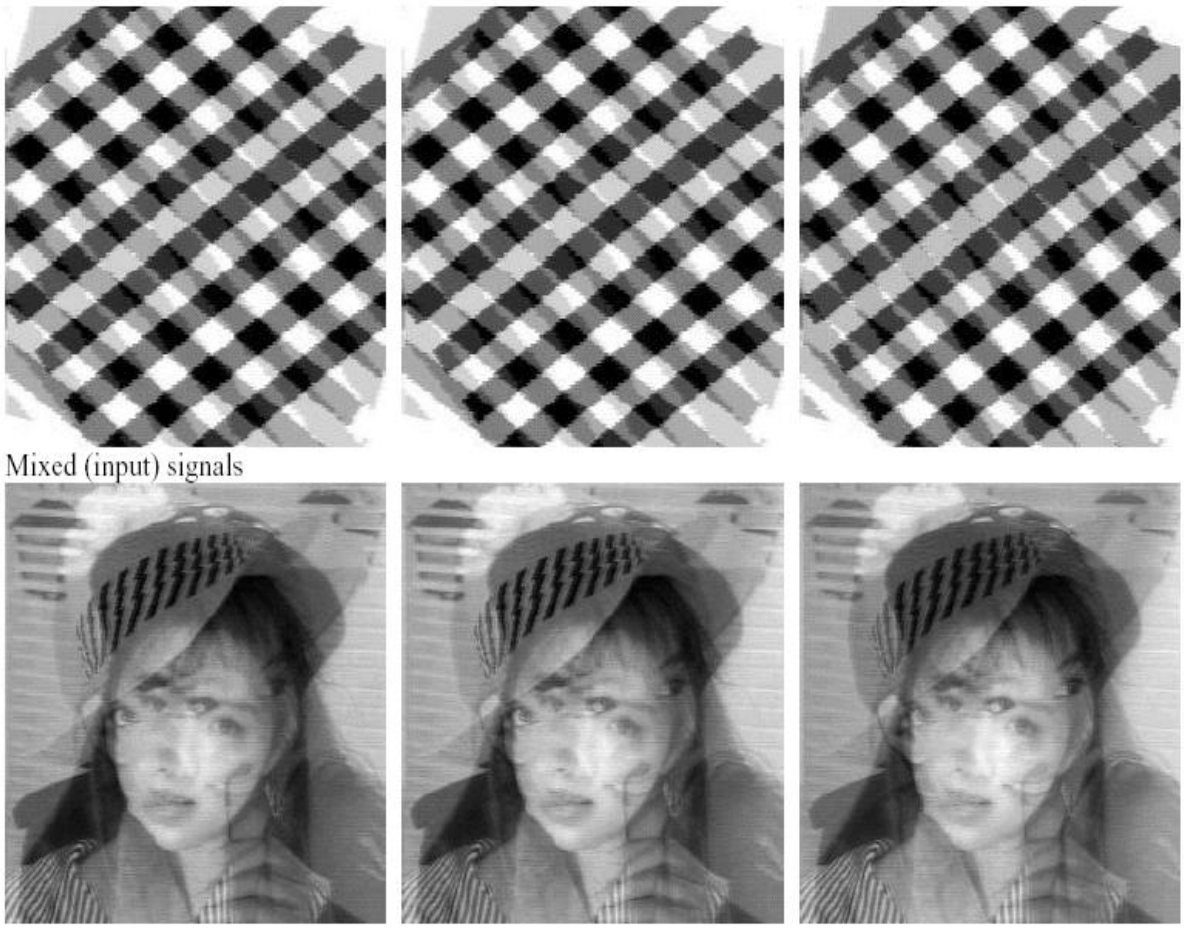

Figure 4: Examples of sensor signals.

6 Conclusions

Most work in the area of blind signal processing deals with simple, synthetic sources, which fully satisfy the independency constraints of the theory. Unfortunately, natural signals are quite different they have complex distributions and are not fully statistically independent, neither in space nor in time. This paper deals with the blind pre-processing of natural source mixtures. In order better to illustrate the test results image sources have been considered. But the results are also true for 1-D signals, e.g. speech or sound sources. We have shown how to modify the update rules in blind source deconvolution in order to adapt to the non-zero auto-correlation values of sources in time.

Acknowledgments

This work was supported by the *Research Centre for Control and Information-Decision Technology (CATID)* at Warsaw University of Technology, Poland.

7 Bibliography

[**AMA97**] Amari S., Douglas S.C., Cichocki A., Yang H.Y.: Novel on-line adaptive learning algorithms for blind deconvolution us-

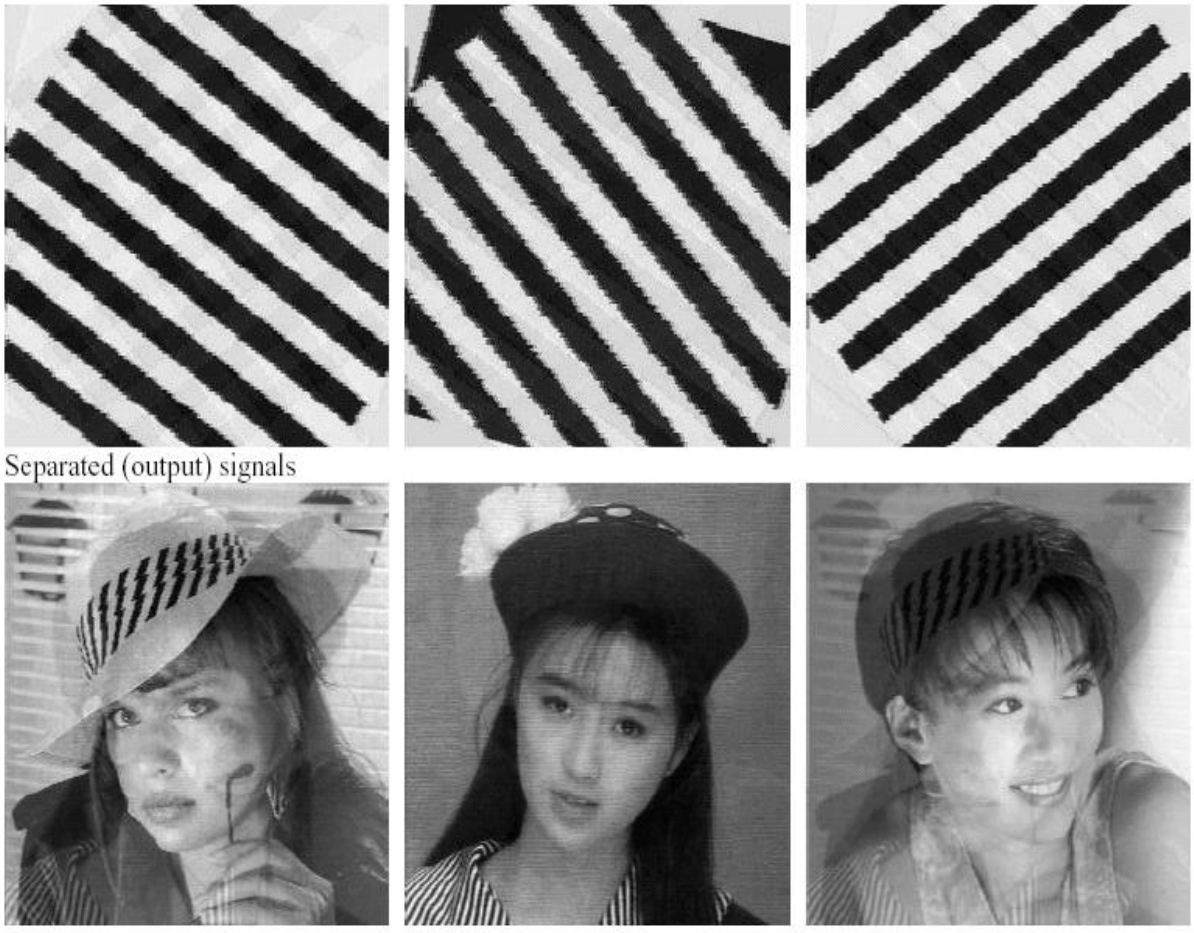

Figure 5: Examples of output signals.

ing the natural gradient approach. *IEEE Signal Proc. Workshop on Signal Processing Advances in Wireless Communications*, April 1997, Paris, 107 112.

[**CAR96**] Cardoso J.F., Laheld B.: Equivariant adaptive source separation, *IEEE Trans. on Signal Processing*, vol. 44(1996), December 1996, 3017-3030.

[**CIA02**] Cichocki A., Amari S., *Adaptive Blind Signal and Image Processing* , John Wiley, Chichester, UK, 2002.

[**CIC94**] Cichocki A., Unbehauen R., Rummert E.: Robust learning algorithm for blind separation of signals, *Electronics Letters*, vol. 30(1994), August 1994, 1386-1387.

[**DUD73**] Duda R.O., Hart P.E. : *Pattern classification and scene analysis*, John Wiley & Sons, New York, 1973.

[**HUA96**] Hua Y.: Fast maximum-likelihood for blind identification of multiple FIR channels, *IEEE Trans. Signal Processing*, vol. 44 (1996), 661 - 672.

[**HYV96**] Hyvarinen A., Oja E.: Simple neuron models for independent component analysis, *Int. J. Neural Systems*, vol. 7, 1996, 671-687.

[**KAR97**] Karhunen J., Oja E., Wang L., Vigario R., Joutsensalo J.: A class of neural networks for independent component analysis, *IEEE Trans. on Neural Networks*, vol. 8(1997), May 1997, 486-504.

[**KAS00**] Kasprzak W.: ICA algorithms for image restoration. Chapter 3 of the monograph work: Kasprzak W.: Adaptive computation methods in digital image sequence analysis. Prace Naukowe - Elektronika, Nr. 127 (2000), Warsaw University of Techology Press, Warszawa, 170 pages.

[**KAS96**] Kasprzak W., Cichocki A.: Hidden Image Separation From Incomplete Image Mixtures by Independent Component Analysis, *13th Int. Conf. on Pattern Recognition, ICPR'96*, Proceedings. IEEE Computer Society Press, Los Alamitos CA, 1996, vol.II, 394-398.

[**KAS97**] Kasprzak W., Cichocki A., Amari S.: *Blind Source Separation with Convolutive Noise Cancellation*, Neural Computing and Applications, Springer Ltd., London, vol. 6 (1997), 127-141.

[**NIE90**] Niemann H.: *Pattern Analysis and Understanding*, Springer, Berlin, 1990.

[**PAJ96**] Pajunen P.: An Algorithm for Binary Blind Source Separation, Helsinki University of Technology, *Lab. of Computer and Information Science, Report A36*, July 1996.

[**SAB98**] Sabaa I.: Multichannel deconvolution and separation of statistically independent signals for unknown dynamic systems. *Ph.D. dissertation*, Warsaw University of Technology, Dep. of Electrical Engineering, Warsaw, 1998.

[**WID85**] Widrow B., Stearns S.D.: *Adaptive signal processing*, Prentice Hall, Englewood Cliffs, NJ, 1985.

Appendix

Matlab source code of the MBD learning process

```
function weights =  MBDLearningProc(Signal, weightsIN, EtaIN,
sensnum, ...
                    outnum, delay, time, maxsample, Function, LAMBDAmat)
% The learning procedure for multichannel blind source deconvolution,
% based on the modified (by W.Kasprzak) NG- (defined by S.Amari et al.)
% -based learning (update) rule
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Parameters:
%Signal; /* the set of mixed input signals */
%weightsIN; /* the weights to be learned */
%EtaIN; /* the learning rate matrix */
%sensnum; /* number of sensors */
%outnum; /* number of outputs */
%delay; /* number of time delays - shift_signum */
%time;  /* maximum number of learning epochs */
%maxsample; /* length of one epoch */
%Function; /* the index of nonlinear function */
%LAMBDAmat; /* typical (expected) high-order correlation of sources in time */
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author: Wodzimierz Kasprzak (ICCE WUT)
% Place and time: Warszawa, 15. June 2002
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global BLOCKLEN; global WEIGHT_DIFF; global MINIMUM_ETAINIT;
%%%%%%%%%%%%%%%%%%%%%%
%I. Initialisation
[m,SIGLxBLL] = size(Signal); % The size of input data

weights = weightsIN; Eta = EtaIN; ffun = zeros(1,outnum); gfun =
zeros(1,outnum);

% Expected high-order correlations in time
IMAT = zeros(sensnum, sensnum, delay); for (i=1: 1: delay)
   IMAT(:,:,i) = diag(LAMBDAmat(:,i)');
end

%Local variables
s=0; samples=0; ss=0.0; samplestep = 0.0;

%%%%%%%%
%Parameters
```

```
p = outnum; %/* output  number: the y_j output index */
m = sensnum; %/* sensor number: the x_i index */
mmsize = m * m ; %/* m rows and m columns, quadrat */
nmsize = p * m ; %/* p rows and m columns, quadrat */
N = delay; %/* number of deconvolution samples for signals and ref.noise */

% Set Etainit
Etainit = zeros(1,delay); % Initial learning rate in vector form
Etainit = EtaIN; % default value
% or (comment it out if not used)
% a subfunction  if Etainit should depend on variance
Etainit = SetEtainitMBD(p, m, N, Signal, Function);

%Initialize weight matrix and store it for eventual re-start
winitmem = zeros(m,m); weightmat = zeros(m,m,N); winitmem =
InitWeightsMBD(Method, p, m, N, weightsIN(:,:,1)); for (i=1: 1:
delay)
   weightmat(:,:,i) =  winitmem / i;

   Etainit(i) =  Etainit(i) / i;
end Eta = Etainit; weights = weightmat;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% II. Repetitive learning process
sstart = 1; tstart = 1; DDsamples = 0;

for (DD = 1: 1: delay) % Main loop according to time delay index
   ffunmat = zeros(m, DD); %f(y) in matrx form
   gfunmat = zeros(m, DD); %g(y) in matrix form
   xk = zeros(m, DD + DD); yk = zeros(p, DD); %  input and output samples

%/*  Main repetition loop for a single delay index */
   REPEAT = 1;
   while (REPEAT==1)
      REPEAT = 0;
      if (Etainit(DD) < MINIMUM_ETAINIT)
         fprintf('*Error: Etainit is to small. No learning possible !\n');
         break;
      end
      blockDDsamples = 0; % number of learned blocks of samples for current DD
      lambda = 1.0;
      OrigIndex = zeros(1,m);
      %/* Initialize the weight matrix with index DD */
      Eta(DD) = Etainit(DD);
      weights(:,:,DD) = weightmat(:,:,DD);
```

```
%/* Store current weight modules required for convergence test */
wprev = 0.0;
for (k=1: 1: m)
   wprev = wprev + weights(k,:,DD) * (weights(k,:,DD))';
end
%/* Initialize samplestep */
ss = 1.0; %/* the decay step of learning rates */
samplestep = 99 / SIGLxBLL;

REPEAT = 0;
samples = DDsamples;
DDsamples = 0;
%/* c1. Epoch loop */
for (s = 1 : 1 : time)
   %/* Initialize indices to the ffun, gfun memory */
   FBegind = 0; FEndind = 1;
   %/* c2. Time loop */
   for (t = 2 : 1 : maxsample) % delay must be <= BLOCKLEN
      % We need two times DD samples
      blockDDsamples = blockDDsamples + 1;
      baseskip = t;
      %/* c3. Block loop */
      for (l = 1 : 1 : BLOCKLEN)
         baseskip = baseskip + 1; samples = samples + 1;
         DDsamples = DDsamples + 1;
         %/* Get current and delayed input samples */
         xk = GetMultiSample(m, DD + DD, Signal, baseskip);
         %/* Current output samples y(t) of network 'weights':
         %   D output samples */
         yk = ConvFForwardSample(m, p, DD, weights, xk, yk);
         %/* Set current activation function values: ffun, gfun */
         for (i=1:1:DD)
            lamsum = 0.0; lam2sum = 0.0;
            y2 = yk(:,i) .* yk(:,i);
            lamsum = yk(:,i)' * yk(:,i);
            y3 = y2 .* yk(:,i);
            lam2sum = y2' * y2;
            switch(Function)
            case 1,
               ffun = y3'; gfun = yk(:,i)'; % f(y) = y^3, g(y) = y

            otherwise,
               [ffun, gfun] = GetCurrentFunctions(m, Function, ...
                              ffun', gfun', yk(:,i)', y2', y3');
            end
```

```
                %/* Add current function value to delay function memory */
                ffunmat(:, i) = ffun';
                gfunmat(:, i) = gfun';
             end
             FEndind = FEndind + 1; % DELAY_TIME
             %  /* Scalar Eta of steady default decay - nonadaptive */
             Eta(DD) = Etainit(DD) / ss;
             ss = ss + samplestep;

             %/* c4. Deconvolution learning rule */
             weights(:,:,DD) = MBDNGLearningRule(l, m, p, N, DD, Eta, ...
                                    weights, ffunmat, gfunmat, IMAT);
             FBegind = (FBegind + 1); % DELAY_TIME;
             %/* Testing the stability of weights */
             [REPEAT, wprev] = WeightStabilityTest(m, p, mmsize, ...
                                  Etainit, weights(:,:,DD), wprev);
             if (REPEAT == 1)
                Etainit(DD) = 0.4 * Etainit(DD);
                fprintf ('(RESTART) etainit(%d)= %.6f * (10)-3\n', ...
                                 DD, 1000.0 * Etainit(DD));
             end

             if (REPEAT == 1)
                break;
             end
          end %/* Closing the block loop */
          %/* c6. Check weight restart condition */
          if (REPEAT ==1)
             DDsamples = 0;
             break;
          end
       end %/* Closing the time loop */
       %/* c7. Check weight restart condition, again */
       if (REPEAT == 1)
          DDsamples = 0;
          break;
       end
       %/* c8. Modify samplestep for next epoch */
       samplestep = 0.4 * samplestep;
    end %/* Closing the epoch loop */
 end %/* Closing the repeat loop */
end % Closing the DD loop
return;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Subfunction: the modified NG-based update rule
function wpD = MBDNGLearningRule(ll, n, m, N, D, eta, weights, ff,
gf, lmat)
% A single weight matrix with for the delay index D is updated
% Author: Wodzimierz Kasprzak (ICCE WUT)
% Place and time: Warszawa, 15. June 2002
%%%%%%%%%%%%%%%%%%%%%%%
w1n = weights(:,:,1:D); wpD = zeros(m,n);
k = D; %
w1 = zeros(m,n); for (q = 1: 1: k)
   pq = k - q + 1;
   if (q == 1)
      w1 = w1 + (lmat(:,:,pq) - ff(:, 1) * gf(:, pq)') * weights(:,:,pq); %
   else
      w1 = w1 + (lmat(:,:,pq) - ff(:, 1) * gf(:, pq)') * weights(:,:,pq); %
   end
end wpD = weights(:,:,k) + eta(k) * w1; return;
%%%%%%%%%%%%%%%%%%%%%%%%%%
```