Simple Neuron Models for Independent Component Analysis

Aapo Hyvärinen and Erkki Oja
Helsinki University of Technology
Laboratory of Computer and Information Science
Rakentajanaukio 2C, FIN-02150 Espoo, Finland
aapo.hyvarinen@hut.fi, erkki.oja@hut.fi
http://nucleus.hut.fi/~aapo, ~oja

To appear in Int. J. of Neural Systems

February 19, 1997

Abstract

Recently, several neural algorithms have been introduced for Independent Component Analysis. Here we approach the problem from the point of view of a single neuron. First, simple Hebbian-like learning rules are introduced for estimating one of the independent components from sphered data. Some of the learning rules can be used to estimate an independent component which has a negative kurtosis, and the others estimate a component of positive kurtosis. Next, a two-unit system is introduced to estimate an independent component of any kurtosis. The results are then generalized to estimate independent components from non-sphered (raw) mixtures. To separate several independent components, a system of several neurons with linear negative feedback is used. The convergence of the learning rules is rigorously proven without any unnecessary hypotheses on the distributions of the independent components.

1 Introduction

Independent Component Analysis (ICA) (Comon, 1994; Jutten and Herault, 1991) is a signal processing technique whose goal is to express a set of random variables as linear combinations of statistically independent component variables. The main applications are in blind source separation, feature extraction, and in a slightly modified form, in blind deconvolution. In the simplest form of ICA (Comon, 1994), we observe m scalar random variables $v_1, v_2, ..., v_m$ which are linear combinations of n *unknown independent components* $s_1, s_2, ..., s_n$, assumed mutually statistically independent, and zero-mean. In addition, we must assume $n \leq m$. Let us arrange the observed variables v_i into a vector $\mathbf{v} = (v_1, v_2, ..., v_m)^T$ and the component variables s_i into a vector $\mathbf{s}$, respectively; then the linear relationship is given by

$$\mathbf{v} = \sum_{i=1}^{n} \mathbf{a}_i s_i = \mathbf{A}\mathbf{s} \tag{1}$$

Here, $\mathbf{A}$ is an unknown $m \times n$ matrix of full rank, called the *mixing matrix*, whose columns are denoted $\mathbf{a}_i$, $i = 1, ..., n$. The basic problem of ICA is then to estimate the original components s_i from the mixtures v_j or, equivalently, to estimate the mixing matrix $\mathbf{A}$. The columns $\mathbf{a}_i$ are then called the *ICA basis vectors*. The fundamental restriction of the model is that we can only estimate non-Gaussian independent components (except if just one of the independent components is Gaussian). Moreover, neither the energies nor the signs of the independent components can be estimated, because any constant multiplying an independent component in eq. (1) could be cancelled by dividing the corresponding column of the mixing matrix $\mathbf{A}$ by the same constant. For mathematical convenience, we define here that the independent components s_i have unit variance. This makes the (non-Gaussian) independent components unique, up to their signs. Note that no order is defined between the independent components.

In *blind source separation* (Jutten and Herault, 1991), the observed values of $\mathbf{v}$ correspond to a realization of an m-dimensional discrete-time signal $\mathbf{v}(t)$, $t = 1, 2, ...$. Then the components $s_i(t)$ are called source signals, which are usually original, uncorrupted signals or noise sources.

In *blind deconvolution*, a convolved version $v(t)$ of a scalar i.i.d. signal $s(t)$ is observed, again without knowing the signal $s(t)$ or the convolution kernel (Donoho, 1981; Shalvi and Weinstein, 1990). The problem is then to find a separating filter f so that $s(t) = f(t) * v(t)$. The equalizer $f(t)$ is assumed to be a tap-delay line of sufficient length, so that the truncation effects can be ignored. This problem can also be represented (though only approximately) by eq. (1); then the realizations of $\mathbf{v}$ and $\mathbf{s}$ are vectors containing $n = m$ subsequent observations of the signals $v(t)$ and $s(t)$, beginning at different points of time. In other words, a sequence of observations $\mathbf{v}(t)$ is such that $\mathbf{v}(t) = (v(t+n-1), v(t+n-2), ..., v(t))^T$ for $t = 1, 2, ...$. The square matrix $\mathbf{A}$ is determined by the convolving filter. Though this formulation is only approximative, the exact formulation using linear filters leads to essentially the same algorithms and convergence proofs.

Another possible application of ICA is *feature extraction* (Bell and Sejnowski, 1997; Hurri et al., 1996; Karhunen et al., 1997a). Then the columns of $\mathbf{A}$ represent features, and s_i signals the presence and the 'amplitude' of the i-th feature in the observed data $\mathbf{v}$.

The current neural algorithms for Independent Component Analysis (Bell and Sejnowski, 1995; Cardoso and Laheld, 1996; Karhunen et al., 1997b) try to estimate simultaneously all the components. This is often not necessary, or feasible, and it is often desired to estimate only a subset of the independent components. (This is especially true in blind deconvolution, in which it is enough to estimate just one of the independent components as the s_i are simply delayed versions of each other.) This is the starting point of our paper. We introduce *learning rules for a single neuron*, by which the neuron learns to estimate one of the independent components, or ICs. Both learning rules for whitened data and for the 'raw' data are introduced. Some of the learning rules work for ICs of positive kurtosis and others work for ICs of negative kurtosis. General two-unit learning rules that allow estimation of any IC regardless of its kurtosis are also introduced. Finally, we present different feedback mechanisms that make it possible to construct a network of several neurons which can estimate several (1 to n) independent components. Some preliminary results on these learning rules have already been published in (Hyvärinen and Oja, 1996b; Hyvärinen, 1996b; Hyvärinen and Oja, 1996c).

This paper is organized as follows: in Section 2 the background of the problem and preprocessing of the data are explained. The one-unit algorithms are presented in Section 3 (for sphered data) and Section 4 (for raw data). Multi-unit algorithms are discussed in Section 5. Section 6 contains simulation results, and in Section 7 conclusions are drawn, and some extensions of the results are discussed.

2 ICA Using Kurtosis

2.1 Data sphering

Some of the algorithms presented here require a preliminary *sphering* or prewhitening of the data $\mathbf{v}$, i.e. the observed signal $\mathbf{v}$ of eq. (1) is linearly transformed to a signal $\mathbf{x}$

$$\mathbf{x} = \mathbf{M}\mathbf{v} \tag{2}$$

such that the correlation matrix of $\mathbf{x}$ equals unity

$$E\{\mathbf{x}\mathbf{x}^T\} = \mathbf{I}$$

This transformation is always possible. Indeed, it can be accomplished by classical PCA. In addition to sphering, PCA allows us to determine the number of independent components: if the noise level is low, the energy of $\mathbf{x}$ is essentially concentrated on the subspace spanned by the n first principal components, with n the number of independent components in the model (1). Thus we presume in the remainder of this section that the dimension of $\mathbf{x}$ equals the dimension of $\mathbf{s}$.

After sphering we have from (1) and (2):

$$\mathbf{x} = \mathbf{B}\mathbf{s} \tag{3}$$

where $\mathbf{B} = \mathbf{M}\mathbf{A}$ is an *orthogonal* matrix, because

$$E\{\mathbf{x}\mathbf{x}^T\} = \mathbf{B}E\{\mathbf{s}\mathbf{s}^T\}\mathbf{B}^T = \mathbf{B}\mathbf{B}^T = \mathbf{I}$$

Recall that we assumed that the independent components s_i have unit variance. We have thus reduced the problem of finding an arbitrary matrix $\mathbf{A}$ to the simpler problem of finding an orthogonal matrix $\mathbf{B}$. Once $\mathbf{B}$ is found, eq. (3) is used to solve the independent components from the observed $\mathbf{x}$ by

$$\mathbf{s} = \mathbf{B}^T\mathbf{x} \tag{4}$$

The columns of $\mathbf{B}$, or $\mathbf{b}_i$, are called the transformed (ICA) basis vectors. Note that the knowledge of just one transformed ICA basis vector allows us to calculate one (original) ICA basis vector, and to separate one independent component:

$$s_i = \mathbf{b}_i^T\mathbf{x}$$

$$\mathbf{a}_i = \mathbf{M}^{-1}\mathbf{b}_i$$

Methods for sphering the data and for calculating the ICA basis vectors from the transformed ones by a neural network are presented in (Karhunen et al., 1997b).

It is also worthwhile to reflect why sphering alone does not solve the separation problem. This is because sphering is only defined up to an orthogonal transformation: if $\mathbf{M}_1$ is a sphering matrix, then $\mathbf{M}_2 = \mathbf{U}\mathbf{M}_1$ is also a sphering matrix if and only if $\mathbf{U}$ is an orthogonal matrix. Therefore, we have to find the *correct* sphering matrix that equally separates the independent components. This is done by first finding any sphering matrix $\mathbf{M}$, and later determining the appropriate orthogonal transformation $\mathbf{B}$ from a suitable criterion.

2.2 ICA and kurtosis

The goal is now to find, according to eq. (4), a linear transformation of the observed, sphered vector $\mathbf{x}$ such that the elements of the transformed vector are statistically independent. Because the densities of the original independent components s_i are not known, the independence criterion cannot be directly used. Many suggested solutions to the ICA problem use the fourth-order cumulant or *kurtosis* of the ICs, defined for a zero-mean random variable v as

$$\text{kurt}\,(v) = E\{v^4\} - 3(E\{v^2\})^2 \tag{5}$$

For a Gaussian random variable, kurtosis is zero; it is typically positive for distributions with heavy tails and a peak at zero, and negative for flatter densities with lighter tails. Note that for two independent random variables v_1 and v_2 and for a scalar α, it holds $\text{kurt}\,(v_1 + v_2) = \text{kurt}\,(v_1) + \text{kurt}\,(v_2)$ and $\text{kurt}\,(\alpha v_1) = \alpha^4\,\text{kurt}\,(v_1)$.

Let us search for a linear combination of the sphered observations x_i, say, $\mathbf{w}^T\mathbf{x}$, such that it has maximal or minimal kurtosis. Note that if $\mathbf{w} = \mathbf{b}_i$, with $\mathbf{b}_i$ a column of the matrix $\mathbf{B}$, then $\mathbf{w}^T\mathbf{x} = s_i$. Obviously, this minimum/maximum problem is meaningful only if the norm of $\mathbf{w}$ is somehow bounded; let us assume $\|\mathbf{w}\| = 1$. Using the (unknown) orthogonal mixing matrix $\mathbf{B}$, let us define

$\mathbf{z} = \mathbf{B}^T\mathbf{w}$. Then also $\|\mathbf{z}\| = 1$. Using eq. (3) and the properties of the kurtosis, we have

$$\text{kurt}\,(\mathbf{w}^T\mathbf{x}) = \text{kurt}\,(\mathbf{w}^T\mathbf{B}\mathbf{s}) = \text{kurt}\,(\mathbf{z}^T\mathbf{s}) = \sum_{i=1}^{n} z_i^4\,\text{kurt}\,(s_i) \tag{6}$$

Under the constraint $\|\mathbf{w}\| = \|\mathbf{z}\| = const$, the function (6) has a number of local minima and maxima. To make the argument clearer, let us assume for the moment that in the mixture (1) there is at least one independent component s_j whose kurtosis is negative, and at least one whose kurtosis is positive. Then, as was shown by Delfosse and Loubaton (1995) , the extremal points of (6) are the canonical base vectors $\mathbf{z} = \pm\mathbf{e}_j$, i.e. vectors whose all components are zero except one component which is ± 1. The corresponding weight vectors are $\mathbf{w} = \mathbf{B}\mathbf{z} = \pm\mathbf{B}\mathbf{e}_j = \pm\mathbf{b}_j$, i.e. the columns of the orthogonal mixing matrix $\mathbf{B}$. So, by minimizing or maximizing the kurtosis in eq. (6) under the given constraint, the columns of the mixing matrix are obtained as solutions for $\mathbf{w}$, and the linear combination itself will be one of the independent components: $\mathbf{w}^T\mathbf{x} = \pm(\mathbf{b}_j)^T\mathbf{x} = \pm s_j$. Equation (6) also shows that Gaussian components cannot be estimated by this way, because for them kurt (s_i) is zero.

An intuitive interpretation of these results is also possible. Sums of signals tend to be 'more Gaussian' than the original signals (cf. the Central Limit Theorem). Thus, to find the original signals, one has to find the 'least Gaussian' linear combinations of the signals. But, as kurtosis can be considered a measure of the non-normality of a random variable (Comon, 1994; Kendall and Stuart, 1958), this corresponds to finding a linear combination whose kurtosis is, in absolute value, maximal. Under the simplifying assumption made above, this means that we must find the extrema of kurtosis.

These results form the basis of our neural algorithms. Interpreting $\mathbf{w}$ as the weight vector of the neuron, and $\mathbf{x}$ as the input vector of the neuron, the linear output of the neuron $\mathbf{w}^T\mathbf{x}$ is the considered linear combination. Then we obtain Hebbian-like learning terms from the instantaneous gradient of the kurtosis of the linear neural output $\mathbf{w}^T\mathbf{x}$. The constraint $\|\mathbf{w}\| = 1$ will be taken into account by a simple forgetting term.

3 One-Unit Learning Rules for Sphered Data

3.1 Introduction

In this section, we introduce and analyze learning rules that work for sphered (i.e. whitened) data. Corresponding learning rules for non-sphered (raw) data are introduced in Section 4. The advantages of sphering are that, firstly, convergence of the ICA algorithm is speeded up considerably, secondly, noise may be decreased at the same time by the PCA sphering, and thirdly, the mathematical analysis will become somewhat easier. However, sphering also has some disadvantages; especially, the number of independent components should usually be known before separation. By observing the eigenvalues of the covariance matrix in PCA sphering, it may be possible to guess the number of the independent components, but a wrong guess can in fact lead to increased noise in some cases.

We present here some learning rules for a single neuron unit. In the first subsection, we present an algorithm to estimate one of the independent components of negative kurtosis. In the second subsection, a similar algorithm is presented for an independent component of positive kurtosis. In the third subsection, a unified algorithm covering both cases, as well as a rigorous convergence theorem for this algorithm, are presented.

3.2 Separating One Independent Component of Negative Kurtosis

Let us suppose, that we observe a sequence of samples $\mathbf{x}(t), t = 1, 2, ...$ of the sphered variable $\mathbf{x}$, which is a linear combination of independent components $\mathbf{s}$:

$$\mathbf{x}(t) = \mathbf{B}\mathbf{s}(t) \tag{7}$$

Here, the square matrix $\mathbf{B}$ is necessarily orthogonal because of the sphering, as was proven in the preceding section. Now, assume that at least one of the independent components has a negative kurtosis. Then, to estimate one of those independent components, we can use the following learning rule for the weight vector $\mathbf{w}$ of a neuron :

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t)) - \mathbf{w}(t)] \tag{8}$$

where the initial value $\mathbf{w}(0)$ is chosen randomly, and $\mu(t)$ is the learning rate, which must fulfill the conditions usual in stochastic approximation (see Appendix A). The function g is a polynomial that locally approximates a sigmoidal function:

$$g(y) = ay - by^3 \tag{9}$$

where $a > 1$ and $b > 0$. For example, because $\tanh(y) = y - 1/3y^3 + ...$, the polynomial can approximate well a multiple of the tanh function for small y. Indeed, for practical calculations, the tanh function itself may be preferred, but for the mathematical analysis a polynomial is better.

This algorithm can be interpreted as constrained Hebbian learning. The first term in brackets, $\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t))$ is a Hebbian term, when $\mathbf{x}(t)$ is the input of the neuron, $\mathbf{w}(t)$ is the weight vector, and g is the nonlinear learning function. The second term is a constraint term, which prevents $\mathbf{w}(t)$ from growing too large.

The convergence of algorithm (8) is based on the fact that it can be interpreted as a stochastic gradient descent to minimize the function

$$J(\mathbf{w}) = -E\{G(\mathbf{w}^T\mathbf{x}) - \frac{1}{2}\|\mathbf{w}\|^2\}$$

Where $G'(s) = g(s)$; the instantaneous gradient of $J(\mathbf{w})$ is $\nabla J(\mathbf{w}) = -g(\mathbf{w}^T\mathbf{x})\mathbf{x} + \mathbf{w}$. Substituting $G(y) = \frac{a}{2}y^2 - \frac{b}{4}y^4$ from eq. (9), and noting that

$$E\{(\mathbf{w}^T\mathbf{x})^2\} = \mathbf{w}^T E\{\mathbf{x}\mathbf{x}^T\}\mathbf{w} = \|\mathbf{w}\|^2 \tag{10}$$

due to the sphering of $\mathbf{x}$, we get

$$J(\mathbf{w}) = \frac{b}{4}E\{(\mathbf{w}^T\mathbf{x})^4\} + \frac{1}{2}(1-a)\|\mathbf{w}\|^2 \tag{11}$$

This can be further expressed in terms of the kurtosis of $\mathbf{w}^T\mathbf{x}$ by noting that now, due to (10), it holds kurt $(\mathbf{w}^T\mathbf{x}) = E\{(\mathbf{w}^T\mathbf{x})^4\} - 3\|\mathbf{w}\|^4$. This finally yields the following form for the objective function:

$$J(\mathbf{w}) = \frac{b}{4}\,\mathrm{kurt}\,(\mathbf{w}^T\mathbf{x}) + [\frac{3b}{4}\|\mathbf{w}\|^4 + \frac{1}{2}(1-a)\|\mathbf{w}\|^2] \tag{12}$$

As was explained in Section 2, minimization of the kurtosis of the neuron output $\mathbf{w}^T\mathbf{x}$ under the constraint $\|\mathbf{w}\| = const$ separates one of the independent components s_i of negative kurtosis in ICA in the sense that $\mathbf{w}$ will tend to the column $\mathbf{b}_i$ of the matrix $\mathbf{B}$ (up to a scalar multiplier). The normalization of $\|\mathbf{w}\|$ is taken care of by the penalty term $[\frac{3b}{4}\|\mathbf{w}\|^4 + \frac{1}{2}(1-a)\|\mathbf{w}\|^2]$ which only depends on the norm of $\mathbf{w}$, but not on its orientation. A rigorous mathematical analysis of the convergence of algorithm (8) is presented in Subsection 3.4 and Appendix A.

3.3 Separating One Independent Component of Positive Kurtosis

In this subsection, we consider the following algorithm:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[b\mathbf{x}(t)(\mathbf{w}(t)^T\mathbf{x}(t))^3 - \|\mathbf{w}(t)\|^4\mathbf{w}(t)] \tag{13}$$

where the initial value $\mathbf{w}(0)$ is chosen randomly, $\mu(t)$ is the learning rate, and $b > 0$.

Algorithm (13) is the counterpart of algorithm (8) for separating an independent component with positive kurtosis. As the previous algorithm, this algorithm can be interpreted as constrained Hebbian learning. The first term in brackets $\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t))$ is a Hebbian term. The second term is a constraint term, which prevents $\mathbf{w}(t)$ from growing infinite.

As above, the convergence of algorithm (8) is based on the fact that it can be interpreted as a stochastic gradient ascent to maximize the function

$$J(\mathbf{w}) = \frac{b}{4}E\{(\mathbf{w}^T\mathbf{x})^4\} - \frac{1}{3}\|\mathbf{w}\|^6 \tag{14}$$

Thus, algorithm (13) works because of the same principles as algorithm (8). A mathematical analysis of the convergence of algorithm (13) is presented in the next subsection as a special case of the more general learning rule.

3.4 The General Case

The algorithms of the preceding subsections can be unified into a general mathematical form

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[-\sigma\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t)) - f(\|\mathbf{w}(t)\|^2)\mathbf{w}(t)] \tag{15}$$

where f is a scalar function, and $\sigma = \pm 1$ is a sign that determines whether we are minimizing ($\sigma = -1$) or maximizing ($\sigma = +1$) kurtosis. As above, the initial value $\mathbf{w}(0)$ is chosen randomly, $\mu(t)$ is the learning rate sequence, and the function g is a polynomial:

$$g(y) = ay - by^3 \tag{16}$$

where $a \geq 0$ and $b > 0$. Clearly, algorithms (8) and (13) are special cases of algorithm (15).

This algorithm is still a stochastic gradient descent. The function to be minimized is

$$J(\mathbf{w}) = \sigma[\frac{a}{2}\|\mathbf{w}\|^2 - \frac{b}{4}E\{(\mathbf{w}^T\mathbf{x})^4\}] + F(\|\mathbf{w}\|^2) \tag{17}$$

with $F'(y) = f(y)$.

To show the convergence of $\mathbf{w}(t)$ to a column of the orthogonal mixing matrix $\mathbf{B}$, we have the following theorem (for a proof, see Appendix A).

Theorem 1 *In algorithm (15), assume that the signals $\mathbf{x}(t)$, $t = 1, 2, ...$ are samples from a sphered transformation $\mathbf{x}$ of a linear mixture of independent signals according to eqs. (1), (2). Choose the learning rate $\mu(t)$ as required in stochastic approximation, and make some technical assumptions (see Appendix A). Make the following hypotheses*
1) there exists an index i and a real number α such that

$$\sigma \; kurt \, s_i > 0 \tag{18}$$

$$-\sigma a + b\sigma E\{s_i^4\}\alpha^2 - f(\alpha^2) = 0 \tag{19}$$

$$2\sigma b E\{s_i^4\} - 2f'(\alpha^2) < 0 \tag{20}$$

2) if $\sigma = 1$,

$$\lim_{\|\mathbf{w}\| \to \infty} \|\mathbf{w}\|^2 / f(\|\mathbf{w}\|^2) = 0$$

3) either

$$-\sigma a - f(0) > 0 \tag{21}$$

or $a = f(0) = 0$, $\sigma = 1$, and

$$\lim_{\|\mathbf{w}\| \to 0} \|\mathbf{w}\|^2 / f(\|\mathbf{w}\|^2) = \infty$$

Then $\mathbf{w}$ in algorithm (15) will converge, up to a constant, to one of the columns $\mathbf{b}_i$ of the transformed mixing matrix $\mathbf{B}$, where i fullfills condition 1).

Note that no hypotheses on the distributions of s_i (in addition to the basic ones in Section 2) are needed, except that one of the signals must be of appropriate kurtosis (see (18)).

The conditions of the theorem can easily be interpreted heuristically. Condition 3) means that $\mathbf{w} = 0$ is not a minimum. Condition 2) prevents $\mathbf{w}$ from growing infinite. Note that this is not possible when we are minimizing the kurtosis. In condition 1), (18) means that there exists an independent component whose kurtosis is of the right sign for the algorithm, (19) ensures that the corresponding column $\mathbf{b}_i$ is a critical point of J, and (20) ensures that this critical point is a strict local minimum. Incidentally, no condition is needed to prevent the existence of irrelevant local minima.

Using Theorem 1, we can easily find other one-unit separation algorithms. We give here two examples.

Example 1. For independent components of negative kurtosis, it is easy to see that in the previously introduced algorithm, $f \equiv 1$ and the first order term of g are in a certain way redundant: in the averaged differential equation (see Appendix A), these terms are equivalent. Therefore, we might simplify the algorithm and get two new algorithms:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t))] \tag{22}$$

where g is a polynomial as in (16), with $a, b > 0$, and

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[-b\mathbf{x}(t)(\mathbf{w}(t)^T\mathbf{x}(t))^3 + \mathbf{w}(t)] \tag{23}$$

These algorithms fulfill all the conditions of the theorem. Algorithm (22) is appealing because of its simplicity; it is a pure gradient ascent of $G(\mathbf{w}^T\mathbf{x})$, without any normalization term. However, the advantage of algorithm (8) over (22) is that in practical computations, we can approximate g with the tanh function, and thus avoid problems of numerical instability. We cannot approximate g with the tanh function in (22), because $(\mathbf{w}(t)^T\mathbf{x}(t))$ does not stay small enough for the approximation to be correct. As will be seen in the next section, however, algorithm (22) is more general because it can also be used for non-sphered data.

Example 2. The following algorithm has some interesting properties:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[-\sigma\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t)) - (\|\mathbf{w}(t)\|^4 - c)\mathbf{w}(t)] \tag{24}$$

with g as in (16) and $c > a$. Note the explicit construction of the constraint term: The first part $-\|\mathbf{w}(t)\|^4\mathbf{w}(t)$ prevents $\mathbf{w}(t)$ from becoming infinite, and the second part $+c\mathbf{w}(t)$ prevents it from reaching 0. Because of such an all-purpose constraint term, this algorithm can be used, simply by changing the sign σ, for finding independent components of either positive or negative kurtosis. Moreover, if we choose $a >> b$, we can avoid possible numerical instability by using the tanh approximation for g, which was not possible in the preceding algorithm (13).

3.5 General Two-Unit Learning Rule for Sphered Data

Basically, the above learning rules assume that we decide in advance whether we wish to separate a source signal with positive or negative kurtosis. In some cases this is a restriction; e.g. if all the independent components have the same sign, then an algorithm trying to find a component with the opposite sign will fail. It would be useful to have an algorithm that works for both cases with no prior assumptions. Here, we derive such an algorithm by using the algorithm in (24). The result is a learning rule that separates an independent component whose kurtosis is of the same sign as the sign of the cubic term of the learning function $g(s)$. Then we can use a second unit to simultaneously estimate the sign of the kurtosis of the output of the first unit (Delfosse and Loubaton, 1995). Thus, we get *a two-unit system that separates an independent component of any (non-zero) kurtosis.*

Taking, for simplicity, $g(y) = -y^3$ in learning rule (24), we get the learning rule

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[\sigma\mathbf{x}(t)(\mathbf{w}(t)^T\mathbf{x}(t))^3 - (\|\mathbf{w}(t)\|^4 - 1)\mathbf{w}(t)] \tag{25}$$

To estimate the appropriate σ in parallel, we need a second unit that estimates on-line the kurtosis of the output $\mathbf{w}(t)^T\mathbf{x}(t)$ of the first unit. This can be done simply by a separate on-line estimate of the fourth moment $\widehat{m^4(t)}$:

$$\widehat{m^4(t+1)} = (1-\nu)\widehat{m^4(t)} + \nu(\mathbf{w}(t)^T\mathbf{x}(t))^4 \tag{26}$$

where ν is a small constant, and $m^4(0)$ is some rough estimate of the fourth moment. Then we obtain the corresponding on-line estimate $\widehat{\mathrm{kurt}}\,(t)$ of the kurtosis:

$$\widehat{\mathrm{kurt}}\,(t) = \widehat{m^4(t)} - 3\|\mathbf{w}\|^4.$$

Now, we can replace σ in learning rule (25) by the sign of this estimate of kurtosis, and get the *general learning rule*

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[\,\mathrm{sign}(\,\widehat{\mathrm{kurt}}\,(t))\mathbf{x}(t)(\mathbf{w}(t)^T\mathbf{x}(t))^3 - (\|\mathbf{w}(t)\|^4 - 1)\mathbf{w}(t)] \tag{27}$$

4 One-Unit Learning Rules for Non-Sphered Data

4.1 Changing the Forgetting Term

It is not very difficult to modify the learning rules introduced above so that they require no preliminary sphering. The point is to change the constraint (forgetting) term so that it constrains the variance of the linear output, instead of simply the norm of the weight vector.

We shall still denote by $\mathbf{x}$ the input of the neuron, which is now *redefined* to be equal to the raw, non-sphered, mixture data $\mathbf{v}$ of eq. (1). This means that we have simply

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{28}$$

Defining, as before, $\mathbf{z} = \mathbf{A}^T\mathbf{w}$, we still have the following equalities:

$$\text{kurt}\,(\mathbf{w}^T\mathbf{x}) = \text{kurt}\,(\mathbf{z}^T\mathbf{s}) = \sum_i z_i^4\,\text{kurt}\,s_. \tag{29}$$

Thus, we can still find the independent components by minimizing or maximizing the kurtosis of the linear neural output $(\mathbf{w}^T\mathbf{x})$. However, the constraint of the optimization becomes more complicated. The theoretical constraint $\|\mathbf{z}\| = 1$ is no longer equivalent to $\|\mathbf{w}\| = 1$, because $\mathbf{A}$ is not, in general, orthogonal. Instead, the constraint can be expressed in terms of the variance of the output, because by definition, $\|\mathbf{z}\|^2 = \mathbf{w}^T\mathbf{A}\mathbf{A}^T\mathbf{w} = \mathbf{w}^T\mathbf{A}E\{\mathbf{s}\mathbf{s}^T\}\mathbf{A}^T\mathbf{w} = E\{(\mathbf{w}^T\mathbf{x})^2\}$.

Therefore, we can separate an independent component by finding a solution of

$$\max/\min_{E\{(\mathbf{w}^T\mathbf{x})^2\}=1} E\{(\mathbf{w}^T\mathbf{x})^4\} \tag{30}$$

where we have used the fact that under the given constraint, $\text{kurt}\,(\mathbf{w}^T\mathbf{x}) = E\{(\mathbf{w}^T\mathbf{x})^4\} - 3$. The constraint $E\{(\mathbf{w}^T\mathbf{x})^2\} = 1$ may be difficult to take into account explicitly. However, we can achieve a similar result by introducing a penalty (constraint) term in the function to be minimized. Denoting the covariance of $\mathbf{x}$ by $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$, we have $E\{(\mathbf{w}^T\mathbf{x})^2\} = \mathbf{w}^T\mathbf{C}\mathbf{w}$. Thus we have the following unconstrained problem:

$$\max/\min E\{(\mathbf{w}^T\mathbf{x})^4 - F(\mathbf{w}^T\mathbf{C}\mathbf{w})\} \tag{31}$$

where the function F is a penalty coefficient similar to those in Theorem 1. The learning rules to be introduced below are simply gradient descent or ascent algorithms for this optimization problem.

In fact, comparing eq. (17) and eq. (31), it is easy to see that Theorem 1 can be modified so that it applies for the non-sphered case. We simply need to consider forgetting terms of the form $f(\mathbf{w}^T\mathbf{C}\mathbf{w})\mathbf{C}\mathbf{w}$, instead of $f(\|\mathbf{w}\|^2)\mathbf{w}$. On the other hand, in the averaged differential equation, $\mathbf{C}\mathbf{w}$ is equivalent to $(\mathbf{w}^T\mathbf{x})\mathbf{x}$. This is how we obtain the algorithms below. The convergence of the algorithms in this section is thus implied by the proof of Theorem 1, as explained in Appendix A.

4.2 Basic One-unit Learning Rules

4.2.1 Separating One Independent Component of Negative Kurtosis

Let us assume, that the observed signal vector $\mathbf{x}(t)$ is a linear combination of independent components $\mathbf{s}(t)$ according to eq. (28). In contrast to the preceding section, $\mathbf{x}$ need not be sphered. Furthermore, let us assume that at least one of the independent components has a *negative* kurtosis. Then we can separate one of those independent components using the following learning rule for the weight vector $\mathbf{w}$ of a neuron:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t))] \tag{32}$$

where $\{\mathbf{x}(t), t = 0, 1, ...\}$ is the sequence of observed signals, the initial value $\mathbf{w}(0)$ is chosen randomly, and $\mu(t)$ is the learning rate, which must fulfill the conditions usual in stochastic approximation. The non-linear learning function g is basically the same polynomial as above:

$$g(y) = y - y^3 \tag{33}$$

As in the learning rules for sphered data, we could also introduce multiplying coefficients in the function g. However, for simplicity, we ignore such scaling coefficients in this section. This learning rule was introduced independently by Malouche and Macchi (1996).

The cubic (anti-Hebbian) part of the learning function g is trying to minimize the kurtosis of the linear output $\mathbf{w}(t)^T\mathbf{x}(t)$. The linear Hebbian part of g is due to the constraint of this minimization, and prevents $\mathbf{w}$ from converging to $\mathbf{0}$.

In Appendix A it will be proven that algorithm (32) estimates one of the independent components of negative kurtosis in the same sense as the algorithms of the preceding section. This means that the linear output $\mathbf{w}(t)^T\mathbf{x}(t)$ of the neuron converges to $cs_i(t)$ where $s_i(t)$ is an independent component whose kurtosis is negative, and c is a scalar constant. Thus we can say that the neuron learns to separate one of the independent components.

4.2.2 Separating One Independent Component of Positive Kurtosis

To separate an independent component of *positive* kurtosis (assuming, of course, that at least one such independent component exists), we use the following learning rule:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[\mathbf{x}(t)g_{\mathbf{w}(t)}(\mathbf{w}(t)^T\mathbf{x}(t))] \tag{34}$$

where all notation is as with algorithm (8), expect that the learning function $g_{\mathbf{w}}$ is defined as follows:

$$g_{\mathbf{w}}(y) = -y(\mathbf{w}^T\mathbf{C}\mathbf{w})^2 + y^3 \tag{35}$$

where $\mathbf{C}$ is the covariance matrix of $\mathbf{x}(t)$, i.e. $\mathbf{C} = E\{\mathbf{x}(t)\mathbf{x}(t)^T\}$. The problem with this definition of $g_{\mathbf{w}}$ is that we need an estimate of the covariance matrix. Therefore, the learning rule is not, strictly speaking, a one-unit rule. This problem can be circumvented in two ways. First, the neuron might estimate locally the variance $E\{(\mathbf{w}^T\mathbf{x})^2\}$ with a larger time-window, and use it instead of $\mathbf{w}^T\mathbf{C}\mathbf{w}$. A second method is to redefine $g_{\mathbf{w}}$ as follows:

$$g_{\mathbf{w}}(y) = -y\|\mathbf{w}\|^4 + y^3 \tag{36}$$

After these modifications, the neuron needs no supplementary information. However, if $g_{\mathbf{w}}$ is defined as in (36), the convergence properties cannot be proven rigorously. If $\mathbf{C}$ is badly conditioned, spurious minima may appear.

The behaviour of algorithm (34) is completely analogous to algorithm (8). The linear output $\mathbf{w}(t)^T\mathbf{x}(t)$ of the neuron converges (up to a multiplicative constant) to one of the independent components $s_i(t), i = 1...m$ whose kurtosis is positive.

4.3 General Two-Unit Learning Rule for Non-Sphered Data

As in the sphered case in subsection 3.5, we can combine learning rules (32) and (34) to obtain a learning rule that separates a source whose kurtosis is of the same sign as the sign of the cubic term of the learning function, and then use a second unit to simultaneously estimate the sign of the kurtosis of the output of the first unit.

To estimate the appropriate sign of the learning term in parallel, we need two separate on-line estimates of the second $(\widehat{m^2}(t))$ and the fourth $(\widehat{m^4}(t))$ moments, which then enable the estimation of the kurtosis:

$$\begin{aligned} \widehat{m^2}(t+1) &= (1-\nu)\widehat{m^2}(t) + \nu(\mathbf{w}(t)^T\mathbf{x}(t))^2 && (37) \\ \widehat{m^4}(t+1) &= (1-\nu)\widehat{m^4}(t) + \nu(\mathbf{w}(t)^T\mathbf{x}(t))^4 && (38) \\ \widehat{\mathrm{kurt}}\,(t) &= \widehat{m^4}(t) - 3(\widehat{m^2}(t))^2. && (39) \end{aligned}$$

Now, we can replace σ in learning rules by the sign of this estimate of kurtosis, and get the *general learning rule*

$$\mathbf{w}(t+1) = \mathbf{w}(t)+\mu(t)[\ \mathrm{sign}(\ \widehat{\mathrm{kurt}}\,(t))\mathbf{x}(t)(\mathbf{w}(t)^T\mathbf{x}(t))^3+\mathbf{x}(t)(\mathbf{w}(t)^T\mathbf{x}(t))(1-\|\mathbf{w}(t)\|^4)] \qquad (40)$$

where $\widehat{\mathrm{kurt}}\,(t)$ is the estimate of kurtosis given by the supplementary unit, and $\|\mathbf{w}\|^4$ may be replaced by $(\mathbf{w}^T\mathbf{C}\mathbf{w})^2$ for better convergence.

5 The Multi-Unit Learning Rules

In this section, we introduce an extension of the above one-unit rules for estimating all independent components. Based on the general multi-unit rule, two different ways of extracting all independent components are presented.

The one-unit rules of the preceding sections allowed us to extract one of the independent components from a linear mixture. No method was presented to choose the independent component that was extracted. Often what one needs is the whole set of original signals, or at least several of them. Note that, contrary to deterministic iteration algorithms, the initial value $\mathbf{w}(0)$ cannot be used for this purpose because of the random wandering of the solution in the beginning of any algorithm based on stochastic approximation.

In this section we present different algorithms for this purpose. Recall that after sphering, the (transformed) ICA basis vectors, or the columns of the matrix $\mathbf{B}$ of eq. (3), are orthogonal. Our algorithms for sphered data will find a transformed ICA basis vector corresponding to an independent component with kurtosis of a given sign *and* orthogonal to a given set of transformed ICA basis vectors. Then we will be able to extract all the independent components. For non-sphered data, this idea is generalized by introducing a feedback that decorrelates the outputs of the neurons.

5.1 Parallel Networks for Sphered Data

In some on-line PCA learning rules like the SGA algorithm (Oja and Karhunen, 1985) and the GHA algorithm (Sanger, 1989), a special *hierarchical orthogonalizing* feedback term was used, similar to deflation-type numerical algorithms for solving the eigenvectors of the covariance matrix. The same idea carries over to the more complicated ICA learning rules. Let us introduce an orthogonalizing feedback term in the general learning rule (15), to obtain the following multi-unit algorithm:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[-\sigma\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t)) - f(\|\mathbf{w}\|^2)\mathbf{w}(t) - d\sum_{k=1}^{N}\mathbf{q}^k(\mathbf{w}(t)^T\mathbf{q}^k)] \tag{41}$$

where $\mathbf{q}^k, k = 1..N$ are the (not necessarily normalized) ICA basis vectors already found, and d is a constant to be defined. Of course, if $N = 0$, this is the general one-unit algorithm of Section 3. The algorithm can be used for finding all the ICA basis vectors in a hierarchal network (Karhunen et al., 1997b), where the sum on the right-most term is over the neurons that are higher in the hierarchy. Then the weight vector of any neuron in the network will converge to a (transformed) ICA basis vector *not* equal to one of the basis vectors found by the preceding neurons. The conditions are as in Theorem 1; in addition, d (the power of feedback), must be large enough. Let us state this as a theorem (proven in Appendix A):

Theorem 2 *In algorithm (41), suppose $\mathbf{x}(t)$ is a sphered transformation of a linear mixture of independent signals, as explained in Sections 2 and 3. Suppose that $\frac{\mathbf{q}^k}{\|\mathbf{q}^k\|}, k = 1..N$ are transformed ICA basis vectors. Call S the set of their indices as columns of $\mathbf{B}$. In addition to the hypotheses of Theorem 1, suppose that i in condition 1) can be chosen so that $i \notin S$, and make the following hypothesis*
4) for any $i \in S$ and $\alpha \neq 0$ we have

$$d > [-\sigma a + b(\max(\sigma\, kurt\, s_i, 0) + 3\sigma)\alpha^2 - f(\alpha^2)]/M \tag{42}$$

where $M = \min \|\mathbf{q}^k\|^2$.

Then $\mathbf{w}(t)$ converges, up to a constant, to one of the columns $\mathbf{b}_i$ of the mixing matrix $\mathbf{B}$, where i fullfills condition 1), and $i \notin S$.

As in the one-unit case, algorithm (41) can be interpreted as a stochastic gradient descent to minimize the function

$$J(\mathbf{w}) = \sigma E\{G(\mathbf{w}^T\mathbf{x})\} + \frac{1}{2}F(\|\mathbf{w}\|^2) + \frac{d}{2}\sum_{k=1}^{N}({\mathbf{q}^k}^T\mathbf{w}(t))^2$$

where G and F are as in Section 3. The last term is again a penalty term, which prevents the algorithm from converging to previously found basis vectors. It is clear that d must be large enough, which corresponds to condition 4) of Theorem 2. The exact evaluation of the minimal value for d is rather tedious. See below for minimal values in the most important cases.

It is quite easy to modify the learning rule in (41) so that the network becomes *symmetric*. To achieve this, it is simply enough to take the sum in the last term over all the neurons in the network. Then we obtain a system that resembles a non-linear version of the symmetric PCA subspace rule (Oja, 1992). Such an algorithm is interesting because the computations can be considered more parallel than the computations in a hierarchical network. Mathematical analysis of such a network, however, becomes extremely complicated.

5.2 Determining the Force of Feedback

It remains to determine the minimal value of d that guarantees convergence in the conditions of Theorem 2. We do this separately for the cases of positive or negative kurtosis. To be able to use the same value of d for all units, we must first determine the constants a and b so that $M \geq 1$ in (42), and then evaluate (42). For algorithm (8) with feedback, we get the inequalities

$$a - 1 \geq 3b, d > a - 1 \tag{43}$$

In the case of algorithm (13) with feedback, we must have

$$b \geq \frac{1}{k}, d > \frac{b^2 k^2}{4} \tag{44}$$

where $k = \max_i E\{s_i^4\}$. In practice, it is enough to take any $k \geq \max_i E\{s_i^4\}$.

5.3 A Sequential Approach

The problem of not knowing the signs of the kurtoses of the ICs can also be solved, instead of estimating the kurtosis in parallel, by the following sequential method. We first use one algorithm with feedback to extract, one by one, all independent components with kurtosis of a given sign, and when there are no more left, we extract the remaining independent components with an algorithm that works for ICs whose kurtosis is of the other sign. Because we can normalize every (transformed) ICA basis vector after it has been found, we do not need to determine a and b as in the hierarchical network. However, d must be large enough. Choosing a, b and d so that we have both (43) and (44) is enough in any case.

We also need a way of determining whether we have already extracted all independent components of a given sign. This can easily be accomplished by comparing the norm of the vector $\mathbf{w}$ after convergence. First, suppose we use algorithm (8) with feedback, i.e. algorithm (41) with $\sigma = -1, f(y) \equiv 1$, and g as in (9). If and only if

$$\|\mathbf{w}\|^2 \geq \frac{a-1}{3b}, \tag{45}$$

the algorithm converged properly, and $\mathbf{w}$ is an ICA basis vector. Otherwise, there are no more independent components of negative kurtosis left unextracted, and one must begin to extract the remaining independent components using algorithm (13) with feedback. Second, suppose we begin extraction with algorithm (13). Then the corresponding criterion is

$$\|\mathbf{w}\|^2 \geq 3b \tag{46}$$

These criteria are direct consequences of equality (55) in Appendix A. The sequential method is illustrated in the next section.

5.4 Feedback for Raw Data

It is easy to see that if the data $\mathbf{x}$ is not sphered, we must generalize the feedbacks introduced above so as to obtain a *decorrelating* feedback. Taking for example the learning rule (32), we would thus obtain as the hierarchical multi-unit learning rule:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu(t)[\mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t)) - d\sum_{k=1}^{N}\mathbf{x}({\mathbf{q}^k}^T\mathbf{x})(\mathbf{w}(t)^T\mathbf{C}\mathbf{q}^k)] \tag{47}$$

where $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$ is the covariance matrix of the data. The symmetric version can be obtained by the same logic.

5.5 Bigradient Feedback

Finally, we show how the bigradient approach, developed in (Wang et al., 1995; Karhunen et al., 1997b) for sphered data, can be adapted to non-sphered data. Thus we obtain a decorrelating bigradient feedback. The bigradient feedback term also contains a constraint term. As an example, let us take the simple cubic one-unit learning rule. Assuming we have a system of N neurons with weight vectors $\mathbf{w}^i, i = 1...N$, which define the columns of the weight matrix $\mathbf{W}$, the weight matrix is updated according to the learning rule

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \mu(t)\sigma\mathbf{x}(t)(\mathbf{x}(t)^T\mathbf{W}(t))^3 + \alpha\mathbf{C}\mathbf{W}(t)(\mathbf{I} - \mathbf{W}(t)^T\mathbf{C}\mathbf{W}(t)) \tag{48}$$

where α is a constant in the range $[0.5, 1]$, $\mu(t)$ is the ordinary learning rate sequence, $\mathbf{C}$ is the covariance matrix of the inputs, and the cubic function is applied separately on every component of the vector. If the data is sphered, the learning rule simplifies to that presented in (Wang et al., 1995; Karhunen et al., 1997b), as then $\mathbf{C} = \mathbf{I}$. A hierarchical bigradient feedback is also possible, see (Wang et al., 1995; Karhunen et al., 1997b). The proof and the motivation for this bigradient approach are identical to those used in the sphered case; the only difference is that instead of dot-products of weight vectors, one must consider covariances of outputs.

6 Simulation Results

To illustrate the algorithms, we applied the sequential method to blind separation of four source signals from linear mixtures.

The source signals were artificially generated to allow easy visual inspection of the results. The sources were samples of 200 values, depicted in Figure 1 in Appendix B. Two of the signals have a negative kurtosis, and the other two have a positive kurtosis. The mixing matrix $\mathbf{A}$ was randomly chosen. The mixed signals are depicted in Figure 2.

As explained in Section 2, the observed data, i.e. the mixed signals, was first sphered. Sphered data is shown in Figure 3. No separation was achieved by the sphering alone.

First, we chose to separate signals of negative kurtosis. This choice has no influence on the results. We used algorithm (8), i.e. algorithm (15) with $\sigma = 1$ and $f \equiv 1$, which is also equivalent to (41) with $N = 0$. As g we used the polynomial in (16) (and not the tanh function), with $a = 1.6$ and $b = .2$. To avoid numerical instability, b was chosen small. The choice of a was quite arbitrary; this particular choice was made only to simplify the right-hand side of the criterion in (45). Of course, to separate the first signal, no feedback was used. Later on, the force of feedback was chosen to be $d = 3$, based on inequalities (43) and (44).

Running the algorithm once, we obtained a weight vector $\mathbf{q}^1$. Applying criterion (45), to $\mathbf{q}^1$, we saw the algorithm had converged correctly, and we had one estimated source signal, depicted in the upper left-hand corner of Figure 4. Note that no order is defined between source signals, so there is no way to choose which one of the signals with negative kurtosis is found first.

Then we ran algorithm (41), with the same settings but with one feedback, i.e. $N = 1$, and $\mathbf{q}^1$ was the vector found in the first run. Again, criterion (45) was fulfilled and we had estimated a second source signal, shown in the upper right-hand corner of Figure 4.

After the third run, the criterion in (45) was no more fulfilled. This means that no more signals of negative kurtosis were left to extract. Thus we started using algorithm (13) with feedback to separate the remaining signals. b was chosen small (0.3) to prevent numerical instability. Thus, we were able to separate the two signals depicted in the lower row of Figure 4.

Visual comparison of Figures 1 and 4 confirms the validity of our algorithms.

7 Conclusions

We have presented one-unit neural learning rules for estimation of individual independent components in Independent Component Analysis. Both rules for whitened and non-whitened data were discussed. Independent components of any kurtosis can be estimated by using a special second unit that estimates the kurtosis of the neuronal output in parallel. Moreover, a system of several units may be taught to separate several independent components. The convergence of these algorithms can be analytically proven without any unnecessary restrictions.

The basic methods presented above admit several extensions. First, the issue of locality may be important in some applications and in the case of biologically inspired modelling. The rather non-local, but computationally simple feedback used above can be replaced by a more local kind of feedback (Hyvärinen, 1996a). Second, instead of the simple linear mixtures as in eq. (1), we can also consider more general, non-linear mixtures. In (Pajunen et al., 1996), the Self-Organizing Map was used for non-linear blind source separation for sources of negative kurtosis. Third, the computations needed in these neural learning rules can also be performed in a computationally attractive way using a fixed-point algorithm, developed in (Hyvärinen and Oja, 1997). Fourth, a large family of contrast functions was introduced in (Hyvärinen, 1997a). These contrast functions may be used instead of kurtosis in the learning rules (Hyvärinen, 1997a; Hyvärinen and Oja, 1996a), thus obtaining estimators that often have better statistical properties than those using kurtosis (Hyvärinen, 1997b).

References

Bell, A. and Sejnowski, T. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159.

Bell, A. and Sejnowski, T. (1997). The 'independent components' of natural scenes are edge filters. *Vision Research*. To appear.

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

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

Delfosse, N. and Loubaton, P. (1995). Adaptive blind separation of independent sources: a deflation approach. *Signal Processing*, 45:59–83.

Donoho, D. (1981). On minimum entropy deconvolution. In *Applied Time Series Analysis II*, pages 565–608. Academic Press.

Hurri, J., Hyvärinen, A., Karhunen, J., and Oja, E. (1996). Image feature extraction using independent component analysis. In *Proc. NORSIG'96*, pages 475–478, Espoo, Finland.

Hyvärinen, A. (1996a). Purely local neural principal component and independent component learning. In *Proc. Int. Conf. on Artificial Neural Networks*, pages 139–144, Bochum, Germany.

Hyvärinen, A. (1996b). Simple one-unit neural algorithms for blind source separation and blind deconvolution. In *Proc. Int. Conf. on Neural Information Processing*, pages 1201–1206, Hong Kong.

Hyvärinen, A. (1997a). A family of fixed-point algorithms for independent component analysis. In *Proc. ICASSP'97*, pages 3917–3920, Munich, Germany.

Hyvärinen, A. (1997b). One-unit contrast functions for independent component analysis: A statistical analysis. Submitted.

Hyvärinen, A. and Oja, E. (1996a). Independent component analysis by general non-linear hebbian-like learning rules. Technical Report A41, Helsinki University of Technology, Laboratory of Computer and Information Science. Submitted to a journal.

Hyvärinen, A. and Oja, E. (1996b). A neuron that learns to separate one independent component from linear mixtures. In *Proc. IEEE Int. Conf. on Neural Networks*, pages 62–67, Washington, D.C.

Hyvärinen, A. and Oja, E. (1996c). One-unit learning rules for independent component analysis. In *NIPS*96*, Denver, Colorado.

Hyvärinen, A. and Oja, E. (1997). A fast fixed-point algorithm for independent component analysis. *Neural Computation*. To appear.

Jutten, C. and Herault, J. (1991). Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24:1–10.

Karhunen, J., Hyvärinen, A., Vigario, R., Hurri, J., and Oja, E. (1997a). Applications of neural blind separation to signal and image processing. In *Proc. ICASSP'97*, pages 131–134, Munich, Germany.

Karhunen, J., Oja, E., Wang, L., Vigario, R., and Joutsensalo, J. (1997b). A class of neural networks for independent component analysis. *IEEE Trans. on Neural Networks*. To appear.

Kendall, M. and Stuart, A. (1958). *The Advanced Theory of Statistics*. Charles Griffin & Company.

Kushner, H. and Clark, D. (1978). *Stochastic approximation methods for constrained and unconstrained systems*. Springer-Verlag.

Malouche, Z. and Macchi, O. (1996). Extended anti-Hebbian adaptation for unsupervised source extraction. In *Proc. ICASSP'97*, pages 1664–1667, Atlanta, Georgia.

Oja, E. (1992). Principal components, minor components, and linear neural networks. *Neural Networks*, 5:927–935.

Oja, E. and Karhunen, J. (1985). On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Math. Analysis and Applications*, 106:69–84.

Pajunen, P., Hyvärinen, A., and Karhunen, J. (1996). Nonlinear blind source separation by self-organizing maps. In *Proc. Int. Conf. on Neural Information Processing*, pages 1207–1210, Hong Kong.

Sanger, T. (1989). Optimal unsupervised learning in a single-layered linear feedforward network. *Neural Networks*, 2:459–473.

Shalvi, O. and Weinstein, E. (1990). New criteria for blind deconvolution of nonminimum phase systems (channels). *IEEE Trans. on Information Theory*, 36(2):312–321.

Wang, L., Karhunen, J., and Oja, E. (1995). A bigradient optimization approach for robust PCA, MCA, and source separation. In *Proc. IEEE Int. Conf on Neural Networks '95*, pages 1684–1689, Perth, Australia.

A Appendix: Proofs of Theorems 1 and 2

First, it is clear that Theorem 1 is a special case of Theorem 2. Thus it is enough to prove Theorem 2.

Algorithm (41) is a stochastic gradient descent for the function

$$J(\mathbf{w}) = -\sigma E\{G(\mathbf{w}^T\mathbf{x})\} + \frac{1}{2}F(\|\mathbf{w}\|^2) + \frac{d}{2}\sum_{k=1}^{N}(\mathbf{w}^T\mathbf{q}^k)^2 \tag{49}$$

Therefore, what we need to prove is that the local minima of J are exactly points of the type $\pm C_i\mathbf{b}_i$, i.e. the columns of the mixing matrix (up to a constant). According to the theory of stochastic approximation (Kushner and Clark, 1978; Delfosse and Loubaton, 1995), if we make some not unrealistic technical assumptions, and choose μ so that

$$\sum \mu(t) = \infty, \sum \mu(t)^2 < \infty$$

this proves the almost sure convergence of the algorithm. Note that a stochastic gradient descent for J is equivalent to the averaged differential equation

$$d\mathbf{w}/dt = -\sigma E\{\mathbf{x}g(\mathbf{x}^T\mathbf{w})\} - f(\|\mathbf{w}\|^2)\mathbf{w} - d\sum_{k=1}^{N}(\mathbf{w}^T\mathbf{q}^k)\mathbf{q}^k \tag{50}$$

However, the interpretation of algorithm (41) as a gradient descent has some advantages: First, it shows that the differential equation (50) cannot get stuck in cycles, and second, it shows us the connection to the maximization or minimization of kurtosis.

To begin with the proof, let us first permute the columns of $\mathbf{B}$ so that the vectors $\mathbf{q}^k$ are the N first columns of $\mathbf{B}$, i.e. so that $S = 1, 2, ..., N$. The components of $\mathbf{s}$ must be permuted accordingly. Recall that ICA basis vectors and independent components are defined only up to a permutation.

Next, we make the following changes of variables : $\mathbf{z} = \mathbf{B}^T\mathbf{w}$ and $\mathbf{s} = \mathbf{B}^T\mathbf{x}$, (where we drop the index k for simplicity). By the orthogonality of $\mathbf{B}$, we get

$$J(\mathbf{z}) = -\sigma E\{G(\mathbf{z}^T\mathbf{s})\} + \frac{1}{2}F(\|\mathbf{z}\|^2) + d_k/2\sum_{k=1}^{N} z_k^2 \tag{51}$$

where we defined $d_k := d\|\mathbf{q}^k\|^2$ and used the fact that $\mathbf{B}^T\mathbf{q}^k = \|\mathbf{q}^k\|\mathbf{e}_k$, with $\mathbf{e}_i$ being the i-th canonical base vector, i.e.

$$e_i^j = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

The convergence of $\mathbf{x}$ to one of the columns $\mathbf{b}_i$ of the mixing matrix (up to a constant), is now equivalent to the convergence of $\mathbf{z}$ to $C\mathbf{e}_i$.

Note that after the same changes of variables, the function to be optimized in the *not sphered* case becomes also equal to (51). Thus the following proof proves also the convergence of the algorithms in the case of data that is not sphered.

To find the critical points of J, i.e. points where $\nabla J = 0$, we compute

$$\partial J/\partial z_i = -\sigma E\{s_i g(\mathbf{s}^T\mathbf{z})\} - f(\|\mathbf{z}\|^2)\mathbf{z} - d_i 1_{i\leq N} z_i \tag{52}$$

Let g be defined as above, i.e.

$$g(y) = ay - by^3$$

where $a \geq 0$ and $b > 0$. Using the independence and normalization of the s_i, we get an explicit formula for the expectation:

$$E\{s_i g(\mathbf{s}^T\mathbf{z})\} = az_i - bE\{s_i^4\}z_i^3 - 3b\sum_{k\neq i} z_k^2 z_i$$

Which gives us an explicit expression for the equation (52):

$$\partial J/\partial z_i = -\sigma a z_i + \sigma b E\{s_i^4\}z_i^3 + 3\sigma b\sum_{k\neq i} z_k^2 z_i - f(\|\mathbf{z}\|^2)z_i - d_i 1_{i\leq N} z_i \tag{53}$$

$$= -\sigma a z_i + \sigma b \operatorname{kurt} s_i z_i^3 + 3\sigma b\|\mathbf{z}\|^2 z_i - f(\|\mathbf{z}\|^2)z_i - d_i 1_{i\leq N} z_i \tag{54}$$

If $\mathbf{z} = \mathbf{c}$ is a critical point of J, we get

$$-a\sigma + b\sigma E\{s_i^4\}c_i^2 + 3\sigma b\sum_{k\neq i} c_k^2 - f(\|\mathbf{c}\|^2) - d_i 1_{i\leq N} = 0 \tag{55}$$

for every i such that $c_i \neq 0$.

Now we can distinguish 5 kinds of possible critical points for J.

1. the point $\mathbf{c} = 0$
2. the points $\mathbf{c} = \pm C_i\mathbf{e}_i, i > N$ with $\sigma \operatorname{kurt} s_i < 0$
3. the points $\mathbf{c} = \pm C_i\mathbf{e}_i, i > N$ with $\sigma \operatorname{kurt} s_i > 0$
4. the points $\mathbf{c} = \pm C_i\mathbf{e}_i, i = 1..N$
5. any other points $\mathbf{c}$, i.e. points that have at least two non-zero components.

We shall now prove that in the second case $\mathbf{c}$ is a strict local minimum, the fourth case is not possible, and in the other cases we do not have a local minimum. Together with condition 2) of the Theorem, which guarantees that the solution remains bounded, this will complete the proof.

To analyze the nature of the critical point $\mathbf{c}$, we calculate its Hessian matrix $\mathbf{D}$, i.e. the matrix of second-order derivatives in the point $\mathbf{c}$:

$$\mathbf{D}_{ij} = \frac{\partial^2 J}{\partial z_i \partial z_j}$$

For the diagonal elements of $\mathbf{D}$, we get first by differentiation of F:

$$\mathbf{D}_{ii} = -\sigma a + 3\sigma b E\{s_i^4\}c_i^2 + -3\sigma b\sum_{k\neq j} c_k^2 - f(\|\mathbf{c}\|^2) - 2f'(\|\mathbf{c}\|^2)c_i^2 - d_i 1_{i\leq N} \tag{56}$$

Using (55), this becomes:

$$\mathbf{D}_{ii} = \begin{cases} 2\sigma b E\{s_i^4\}c_i^2 - 2f'(\|\mathbf{c}\|^2)c_i^2 & \text{for } c_i \neq 0 \\ -\sigma a + 3\sigma b \sum_{k\neq j} c_k^2 - f(\|\mathbf{c}\|^2) - d_i 1_{i\leq N} & \text{for } c_i = 0 \end{cases}$$

For non-diagonal elements we have by simple differentiation:

$$\mathbf{D}_{ij} = [6\sigma b - 2f'(\|\mathbf{c}\|^2)]c_j c_i. \tag{57}$$

The proof of the theorem is based on the well-known fact that if the matrix $\mathbf{D}$ is negative definite in a critical point, then the critical point is a strict minimum, and if $\mathbf{D}$ is not negative semi-definite, then the point is not a minimum.

In case 1), we have:

$$\mathbf{D} = \mathrm{diag}(k_1, k_2, ...)$$

with $k_i = -\sigma a - f(0) - d1_{i\leq N}$. By condition 4) of the theorem, this is either positive for $i > N$, in which case $\mathbf{D}$ is not negative semi-definite, or the second part of condition 3) is true, which also implies that the point $\mathbf{c} = 0$ is not a minimum.

In cases 2) and 3), first note that this is indeed an equilibrium for $C_i = \alpha$ because of condition 1), cf. (55). We can assume without loss of generality that $i = N + 1$. Then $\mathbf{D}$ becomes:

$$\mathbf{D} = \mathrm{diag}(K_1, K_1, ..., K_1, K_2, K_3, ..., K_3)$$

with

$$K_1 = -\sigma a + 3\sigma b\alpha^2 - f(\alpha^2) - d$$
$$K_2 = (2\sigma b E\{s_1^4\} - 2f'(\alpha^2))\alpha^2$$
$$K_3 = -\sigma a + 3\sigma b\alpha^2 - f(\alpha^2)$$

K_2 is negative by (20). Then (19) implies

$$-\sigma a + \sigma b \,\mathrm{kurt}\, s_i \alpha^2 + s3\alpha^2 - f(\alpha^2) = 0$$

which, as $d > 0$, implies that $K_1 < 0$ and $K_3 < 0$ if $\sigma \,\mathrm{kurt}\, s_i < 0$, and that $K_3 > 0$ if $\sigma \,\mathrm{kurt}\, s_i > 0$. Thus $\mathbf{D}$ is negative definite in case 2), and not negative semi-definite if case 3).

In case 4), we have in fact no critical point by condition 4) of the theorem, cf. (54).

In case 5), first note that by (54), condition 4) of Theorem 2 immediately implies $c_i = 0$ for $i \leq N$. Moreover, (54) implies

$$\mathrm{kurt}\, s_i c_i^2 = \mathrm{kurt}\, s_j c_j^2$$

for all $i, j > N$, such that $c_i \neq 0, c_j \neq 0$ which means that all kurt s_i corresponding to $c_i \neq 0$ must be of the same sign. First, let us suppose that this sign is opposite to sign of σ. To prove that $\mathbf{D}$ is not negative semi-definite, we multiply it on both sides by $\mathbf{e}_k$ where $k > N$ is the index of a non-extracted independent component with $\sigma \,\mathrm{kurt}\, s_k > 0$ (such a k exists because of condition 1)). The product equals

$$-\sigma a + 3\sigma b\|\mathbf{c}\|^2 - f(\|\mathbf{c}\|)^2$$

which is, by (54), positive. Therefore, $\mathbf{D}$ is not negative semi-definite.

Second, let us suppose that the common sign of the kurtosis equals the sign of σ. We only need to consider the system of those z_i such that the corresponding $c_i \neq 0$.

Call the corresponding equilibrium $\mathbf{d}$. Delete irrelevant rows and columns from the matrix $\mathbf{D}$, and renumber the s_i correspondingly. Then $\mathbf{D}$ splits into two parts:

$$\mathbf{D} = -\sigma \mathbf{D}_0 + \mathbf{D}_1$$

where

$$\mathbf{D}_0 = \begin{pmatrix} -2bE\{s_1^4\}d_1^2 & -6bd_1d_2 & -6bd_1d_3 & -6bd_1d_40 & \dots \\ -6bd_2d_1 & -2bE\{s_2^4\}d_2^2 & -6bd_2d_3 & -6bd_2d_40 & \dots \\ -6bd_3d_1 & -6bd_3d_2 & -2bE\{s_3^4\}d_3^2 & -6bd_3d_40 & \dots \\ \vdots & \vdots & \vdots & \ddots & \end{pmatrix} \tag{58}$$

$$= -2b\,\mathrm{diag}(d_1, d_2 \dots) \begin{pmatrix} E\{s_1^4\} & 3 & 3 & 3 & \dots \\ 3 & E\{s_2^4\} & 3 & 3 & \dots \\ 3 & 3 & E\{s_3^4\} & 3 & \dots \\ \vdots & \vdots & \vdots & \ddots & \end{pmatrix} \mathrm{diag}(d_1, d_2, \dots) \tag{59}$$

and

$$\mathbf{D}_1 = \begin{pmatrix} -2f'(\|\mathbf{d}\|^2)d_1d_1 & -2f'(\|\mathbf{d}\|^2)d_1d_2 & \dots \\ -2f'(\|\mathbf{d}\|^2)d_2d_1 & -2f'(\|\mathbf{d}\|^2)d_2d_2 & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \tag{60}$$

By hypothesis, $\mathbf{D}$ has at least two columns and rows. To prove that this matrix is not negative semi-definite, it is enough to multiply it on both sides by the vector

$$\mathbf{a} = (-\frac{1}{d_1}, \frac{1}{d_2}, 0, 0, \dots)$$

We get:

$$\mathbf{a}^T\mathbf{D}\mathbf{a} = \mathbf{a}^T(-\sigma\mathbf{D}_0 + \mathbf{D}_1)\mathbf{a}$$
$$= 2\sigma b(E\{s_1^4\} - 3 + E\{s_2^4\} - 3) + 0$$

which is positive because of the hypotheses on kurtosis and σ, i.e. $\sigma\,\mathrm{kurt}\, s_1 < 0$ and $\sigma\,\mathrm{kurt}\, s_2 < 0$ But this means that $\mathbf{D}$ is not negative semi-definite.

Thus, we have proven Theorem 2.

2 Appendix: Simulations

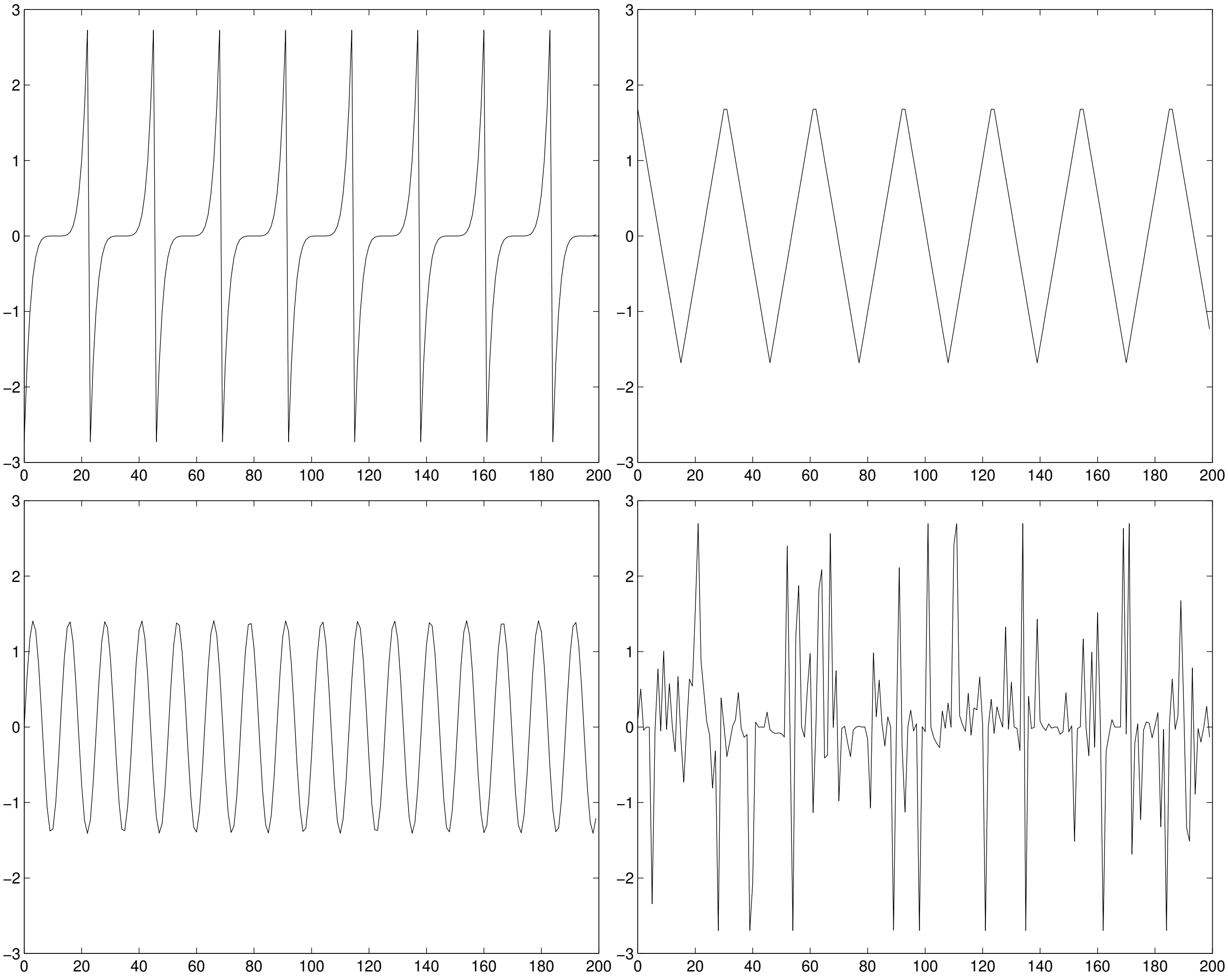

Figure 1: Original source signals. Two of the signals negative kurtoses, and two have positive kurtoses.

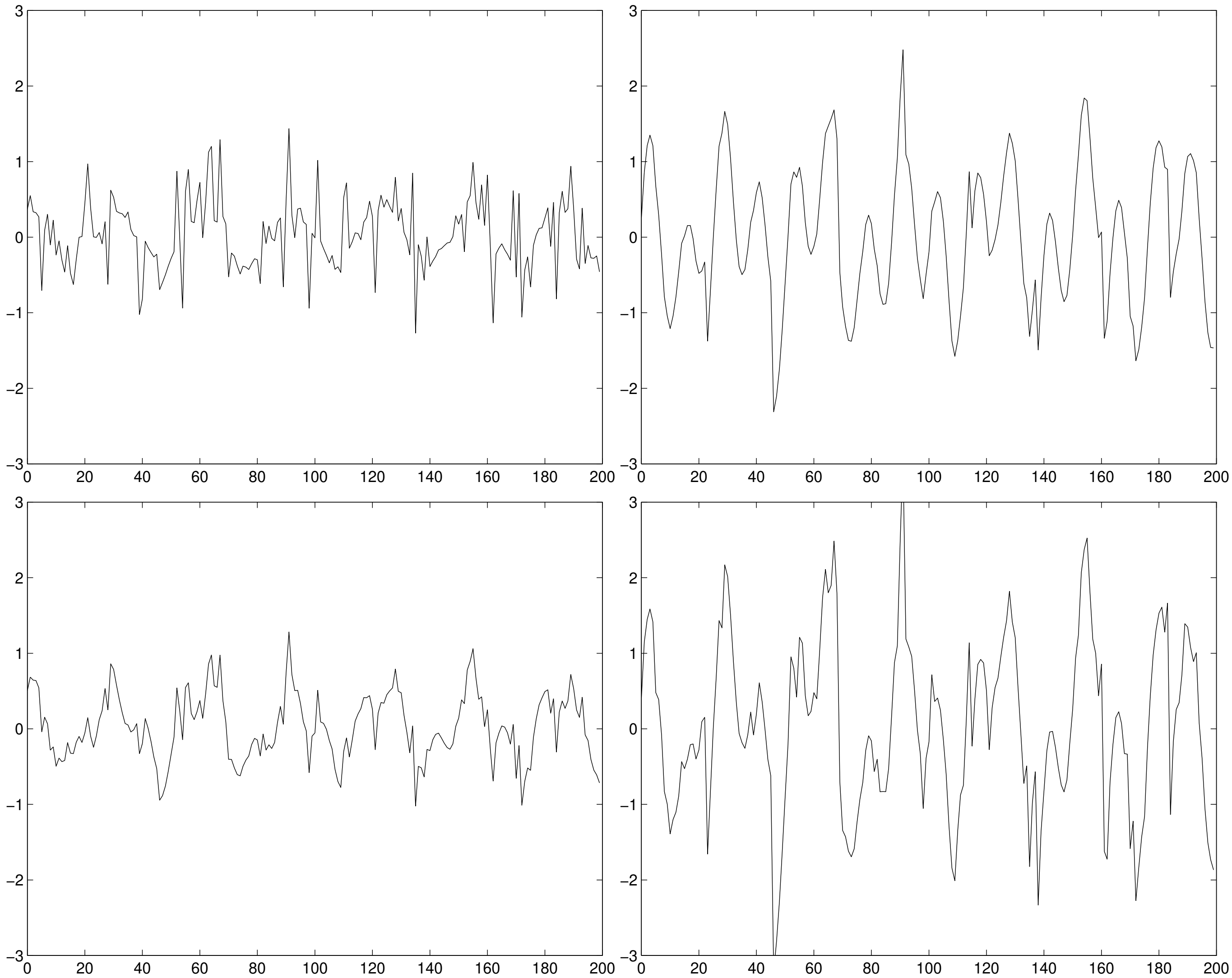

Figure 2: Mixed signals, i.e. observed data.

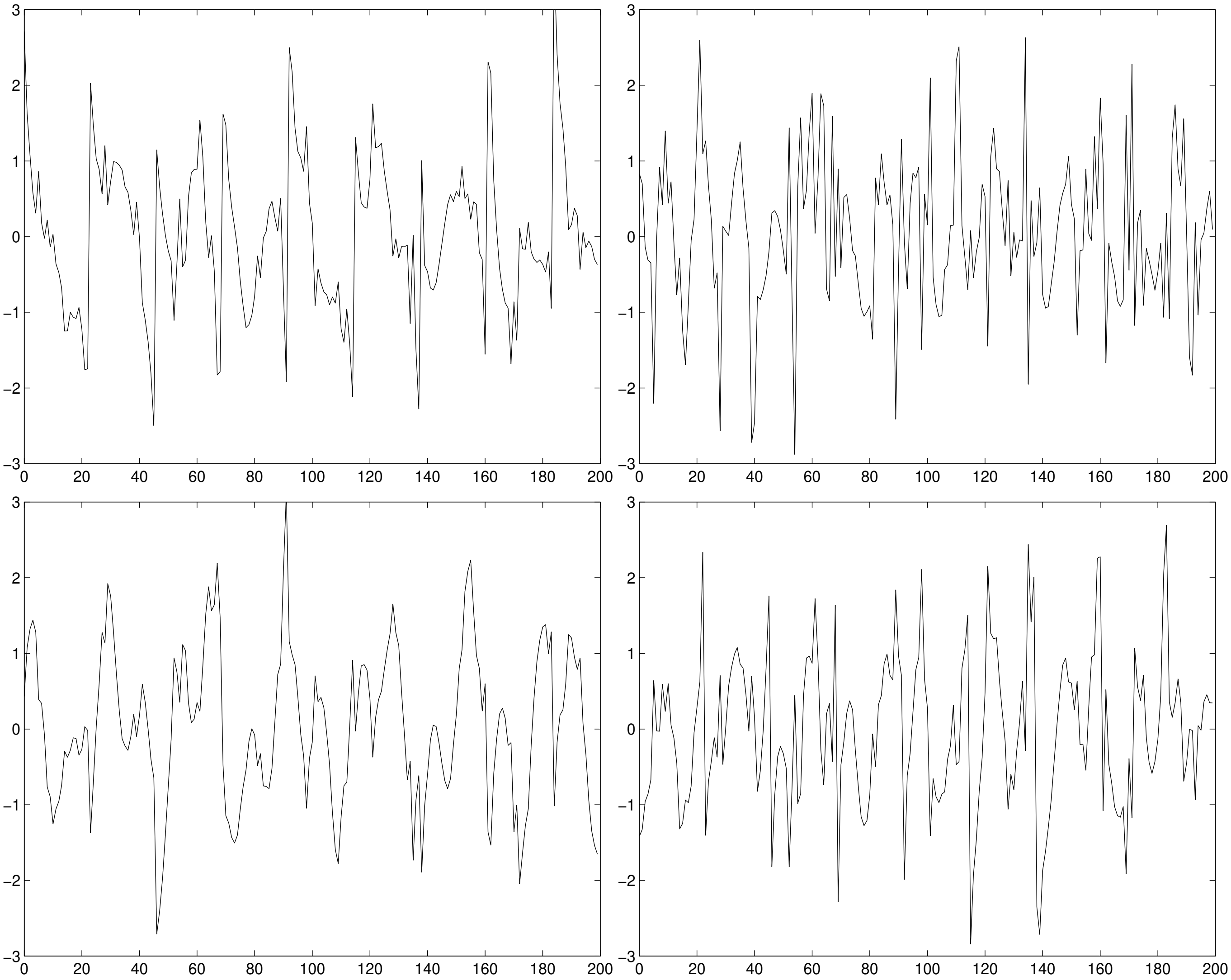

Figure 3: Observed data after sphering.

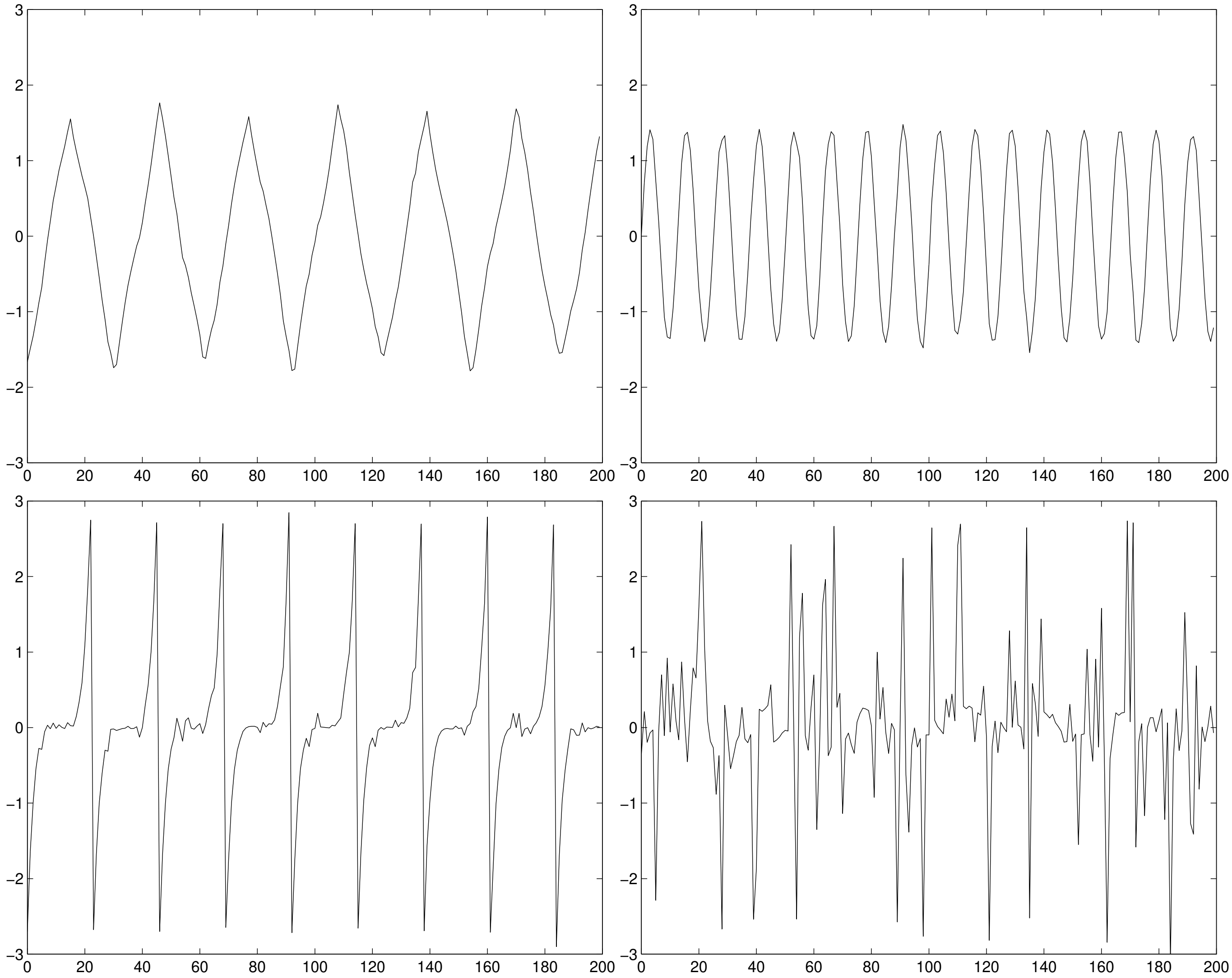

Figure 4: Separated signals, i.e. estimations of source signals.