Fast and Robust Fixed-Point Algorithms for Independent Component Analysis

Aapo Hyvärinen
Helsinki University of Technology
Laboratory of Computer and Information Science
P.O. Box 5400, FIN-02015 HUT, Finland
Email: aapo.hyvarinen@hut.fi
(To appear in IEEE Trans. on Neural Networks)

April 23, 1999

Abstract

Independent component analysis (ICA) is a statistical method for transforming an observed multidimensional random vector into components that are statistically as independent from each other as possible. In this paper, we use a combination of two different approaches for linear ICA: Comon's information-theoretic approach and the projection pursuit approach. Using maximum entropy approximations of differential entropy, we introduce a family of new contrast (objective) functions for ICA. These contrast functions enable both the estimation of the whole decomposition by minimizing mutual information, and estimation of individual independent components as projection pursuit directions. The statistical properties of the estimators based on such contrast functions are analyzed under the assumption of the linear mixture model, and it is shown how to choose contrast functions that are robust and/or of minimum variance. Finally, we introduce simple fixed-point algorithms for practical optimization of the contrast functions. These algorithms optimize the contrast functions very fast and reliably.

1 Introduction

A central problem in neural network research, as well as in statistics and signal processing, is finding a suitable representation or transformation of the data. For computational and conceptual simplicity, the representation is often sought as a *linear* transformation of the original data. Let us denote by $\mathbf{x} = (x_1, x_2, ..., x_m)^T$ a zero-mean m-dimensional random variable that can be observed, and by $\mathbf{s} = (s_1, s_2, ..., s_n)^T$ its n-dimensional transform. Then the problem is to determine a constant (weight) matrix $\mathbf{W}$ so that the linear transformation of the observed variables

$$\mathbf{s} = \mathbf{W}\mathbf{x} \tag{1}$$

has some suitable properties. Several principles and methods have been developed to find such a linear representation, including principal component analysis [30], factor analysis [15], projection pursuit [12, 16], independent component analysis [27], etc. The transformation may be defined using such criteria as optimal dimension reduction, statistical 'interestingness' of the resulting components s_i, simplicity of the transformation, or other criteria, including application-oriented ones.

We treat in this paper the problem of estimating the transformation given by (linear) independent component analysis (ICA) [7, 27]. As the name implies, the basic goal in determining the transformation is to find a representation in which the transformed components s_i are statistically

as independent from each other as possible. Thus this method is a special case of redundancy reduction [2].

Two promising applications of ICA are blind source separation and feature extraction. In *blind source separation* [27], the observed values of $\mathbf{x}$ correspond to a realization of an m-dimensional discrete-time signal $\mathbf{x}(t)$, $t = 1, 2, ...$ Then the components $s_i(t)$ are called source signals, which are usually original, uncorrupted signals or noise sources. Often such sources are statistically independent from each other, and thus the signals can be recovered from linear mixtures x_i by finding a transformation in which the transformed signals are as independent as possible, as in ICA. In *feature extraction* [4, 25], s_i is the coefficient of the i-th feature in the observed data vector $\mathbf{x}$. The use of ICA for feature extraction is motivated by results in neurosciences that suggest that the similar principle of redundancy reduction [2, 32] explains some aspects of the early processing of sensory data by the brain. ICA has also applications in *exploratory data analysis* in the same way as the closely related method of projection pursuit [16, 12].

In this paper, new objective (contrast) functions and algorithms for ICA are introduced. Starting from an information-theoretic viewpoint, the ICA problem is formulated as minimization of mutual information between the transformed variables s_i, and a new family of contrast functions for ICA is introduced (Section 2). These contrast functions can also be interpreted from the viewpoint of projection pursuit, and enable the sequential (one-by-one) extraction of independent components. The behavior of the resulting estimators is then evaluated in the framework of the linear mixture model, obtaining guidelines for choosing among the many contrast functions contained in the introduced family. Practical choice of the contrast function is discussed as well, based on the statistical criteria together with some numerical and pragmatic criteria (Section 3). For practical maximization of the contrast functions, we introduce a novel family of fixed-point algorithms (Section 4). These algorithms are shown to have very appealing convergence properties. Simulations confirming the usefulness of the novel contrast functions and algorithms are reported in Section 5, together with references to real-life experiments using these methods. Some conclusions are drawn in Section 6.

2 Contrast Functions for ICA

2.1 ICA data model, minimization of mutual information, and projection pursuit

One popular way of formulating the ICA problem is to consider the estimation of the following generative model for the data [1, 3, 5, 6, 23, 24, 27, 28, 31]:

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{2}$$

where $\mathbf{x}$ is an observed m-dimensional vector, $\mathbf{s}$ is an n-dimensional (latent) random vector whose components are assumed mutually independent, and $\mathbf{A}$ is a constant $m \times n$ matrix to be estimated. It is usually further assumed that the dimensions of $\mathbf{x}$ and $\mathbf{s}$ are equal, i.e., $m = n$; we make this assumption in the rest of the paper. A noise vector may also be present. The matrix $\mathbf{W}$ defining the transformation as in (1) is then obtained as the (pseudo)inverse of the estimate of the matrix $\mathbf{A}$. Non-Gaussianity of the independent components is necessary for the identifiability of the model (2), see [7].

Comon [7] showed how to obtain a more general formulation for ICA that does not need to assume an underlying data model. This definition is based on the concept of mutual information. First, we define the differential entropy H of a random vector $\mathbf{y} = (y_1, ..., y_n)^T$ with density $f(.)$ as follows [33]:

$$H(\mathbf{y}) = -\int f(\mathbf{y}) \log f(\mathbf{y}) \mathrm{d}\mathbf{y} \tag{3}$$

Differential entropy can be normalized to give rise to the definition of negentropy, which has the appealing property of being invariant for linear transformations. The definition of negentropy J is given by

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y}) \tag{4}$$

where $\mathbf{y}_{gauss}$ is a Gaussian random vector of the same covariance matrix as $\mathbf{y}$. Negentropy can also be interpreted as a measure of nongaussianity [7]. Using the concept of differential entropy, one can define the mutual information I between the n (scalar) random variables $y_i, i = 1...n$ [8, 7]. Mutual information is a natural measure of the dependence between random variables. It is particularly interesting to express mutual information using negentropy, constraining the variables to be *uncorrelated*. In this case, we have [7]

$$I(y_1, y_2, ..., y_n) = J(\mathbf{y}) - \sum_i J(y_i). \tag{5}$$

Since mutual information is the information-theoretic measure of the independence of random variables, it is natural to use it as the criterion for finding the ICA transform. Thus we define in this paper, following [7], the ICA of a random vector $\mathbf{x}$ as an invertible transformation $\mathbf{s} = \mathbf{W}\mathbf{x}$ as in (1) where the matrix $\mathbf{W}$ is determined so that the *mutual information of the transformed components s_i is minimized*. Note that mutual information (or the independence of the components) is not affected by multiplication of the components by scalar constants. Therefore, this definition only defines the independent components up to some multiplicative constants. Moreover, the constraint of uncorrelatedness of the s_i is adopted in this paper. This constraint is not strictly necessary, but simplifies the computations considerably.

Because negentropy is invariant for invertible linear transformations [7], it is now obvious from (5) that finding an invertible transformation $\mathbf{W}$ that minimizes the mutual information is roughly equivalent to *finding directions in which the negentropy is maximized*. This formulation of ICA also shows explicitly the connection between ICA and projection pursuit [11, 12, 16, 26]. In fact, finding a single direction that maximizes negentropy is a form of projection pursuit, and could also be interpreted as estimation of a single independent component [24].

2.2 Contrast Functions through Approximations of Negentropy

To use the definition of ICA given above, a simple estimate of the negentropy (or of differential entropy) is needed. We use here the new approximations developed in [19], based on the maximum entropy principle. In [19] it was shown that these approximations are often considerably more accurate than the conventional, cumulant-based approximations in [7, 1, 26]. In the simplest case, these new approximations are of the form:

$$J(y_i) \approx c[E\{G(y_i)\} - E\{G(\nu)\}]^2 \tag{6}$$

where G is practically any non-quadratic function, c is an irrelevant constant, and ν is a Gaussian variable of zero mean and unit variance (i.e., standardized). The random variable y_i is assumed to be of zero mean and unit variance. For symmetric variables, this is a generalization of the cumulant-based approximation in [7], which is obtained by taking $G(y_i) = y_i^4$. The choice of the function G is deferred to Section 3.

The approximation of negentropy given above in (6) gives readily a new objective function for estimating the ICA transform in our framework. First, to find *one* independent component, or projection pursuit direction as $y_i = \mathbf{w}^T\mathbf{x}$, we maximize the function J_G given by

$$J_G(\mathbf{w}) = [E\{G(\mathbf{w}^T\mathbf{x})\} - E\{G(\nu)\}]^2 \tag{7}$$

where $\mathbf{w}$ is an m-dimensional (weight) vector constrained so that $E\{(\mathbf{w}^T\mathbf{x})^2\} = 1$ (we can fix the scale arbitrarily). Several independent components can then be estimated one-by-one using a deflation scheme, see Section 4.

Second, using the approach of minimizing mutual information, the above one-unit contrast function can be simply extended to compute the whole matrix $\mathbf{W}$ in (1). To do this, recall from (5) that mutual information is minimized (under the constraint of decorrelation) when the sum of the negentropies of the components in maximized. Maximizing the sum of n one-unit contrast functions, and taking into account the constraint of decorrelation, one obtains the following optimization problem:

$$\begin{aligned} &\text{maximize } \sum_{i=1}^{n} J_G(\mathbf{w}_i) \text{ wrt. } \mathbf{w}_i, i = 1, ..., n \\ &\text{under constraint } E\{(\mathbf{w}_k^T\mathbf{x})(\mathbf{w}_j^T\mathbf{x})\} = \delta_{jk} \end{aligned} \tag{8}$$

where at the maximum, every vector $\mathbf{w}_i, i = 1, .., n$ gives one of the rows of the matrix $\mathbf{W}$, and the ICA transformation is then given by $\mathbf{s} = \mathbf{W}\mathbf{x}$. Thus we have defined our ICA estimator by an optimization problem. Below we analyze the properties of the estimators, giving guidelines for the choice of G, and propose algorithms for solving the optimization problems in practice.

3 Analysis of estimators and choice of contrast function

3.1 Behavior under the ICA data model

In this subsection, we analyze the behavior of the estimators given above when the data follows the ICA data model (2), with a square mixing matrix. For simplicity, we consider only the estimation of a single independent component, and neglect the effects of decorrelation. Let us denote by $\hat{\mathbf{w}}$ a vector obtained by maximizing J_G in (7). The vector $\hat{\mathbf{w}}$ is thus an estimator of a row of the matrix $\mathbf{A}^{-1}$.

3.1.1 Consistency

First of all, we prove that $\hat{\mathbf{w}}$ is a (locally) consistent estimator for one component in the ICA data model. To prove this, we have the following theorem:

Theorem 1 *Assume that the input data follows the ICA data model in (2), and that G is a sufficiently smooth even function. Then the set of local maxima of $J_G(\mathbf{w})$ under the constraint $E\{(\mathbf{w}^T\mathbf{x})^2\} = 1$, includes the i-th row of the inverse of the mixing matrix $\mathbf{A}$ such that the corresponding independent component s_i fulfills*

$$E\{s_i g(s_i) - g'(s_i)\}[E\{G(s_i)\} - E\{G(\nu)\}] > 0 \tag{9}$$

where $g(.)$ is the derivative of $G(.)$, and ν is a standardized Gaussian variable.

This theorem can be considered a corollary of the theorem in [24]. The condition in Theorem 1 seems to be true for most reasonable choices of G, and distributions of the s_i. In particular, if $G(u) = u^4$, the condition is fulfilled for any distribution of non-zero kurtosis. In that case, it can also be proven that there are no spurious optima [9].

3.1.2 Asymptotic variance

Asymptotic variance is one criterion for choosing the function G to be used in the contrast function. Comparison of, say, the traces of the asymptotic covariance matrices of two estimators enables direct

comparison of the mean-square error of the estimators. In [18], evaluation of asymptotic variances was addressed using a related family of contrast functions. In fact, it can be seen that the results in [18] are valid even in this case, and thus we have the following theorem:

Theorem 2 *The trace of the asymptotic (co)variance of* $\hat{\mathbf{w}}$ *is minimized when* G *is of the form*

$$G_{opt}(u) = k_1 \log f_i(u) + k_2 u^2 + k_3 \tag{10}$$

where $f_i(.)$ *is the density function of* s_i*, and* k_1, k_2, k_3 *are arbitrary constants.*

For simplicity, one can choose $G_{opt}(u) = \log f_i(u)$. Thus the optimal contrast function is the same as the one obtained by the maximum likelihood approach [34], or the infomax approach [3]. Almost identical results have also been obtained in [5] for another algorithm. The theorem above treats, however, the one-unit case instead of the multi-unit case treated by the other authors.

3.1.3 Robustness

Another very attractive property of an estimator is robustness against outliers [14]. This means that single, highly erroneous observations do not have much influence on the estimator. To obtain a simple form of robustness called B-robustness, we would like the estimator to have a bounded influence function [14]. Again, we can adapt the results in [18]. It turns out to be impossible to have a completely bounded influence function, but we do have a simpler form of robustness, as stated in the following theorem:

Theorem 3 *Assume that the data* $\mathbf{x}$ *is whitened (sphered) in a robust manner (see Section 4 for this form of preprocessing). Then the influence function of the estimator* $\hat{\mathbf{w}}$ *is never bounded for all* $\mathbf{x}$*. However, if* $h(u) = ug(u)$ *is bounded, the influence function is bounded in sets of the form* $\{\mathbf{x} \mid \hat{\mathbf{w}}^T\mathbf{x}/\|\mathbf{x}\| > \epsilon\}$ *for every* $\epsilon > 0$*, where* g *is the derivative of* G*.*

In particular, if one chooses *a function* $G(u)$ *that is bounded*, h is also bounded, and $\hat{\mathbf{w}}$ is rather robust against outliers. If this is not possible, one should at least choose a function $G(u)$ that does not grow very fast when $|u|$ grows.

3.2 Practical choice of contrast function

3.2.1 Performance in the exponential power family

Now we shall treat the question of choosing the contrast function G in practice. It is useful to analyze the implications of the theoretical results of the preceding section by considering the following exponential power family of density functions:

$$f_\alpha(s) = k_1 \exp(k_2 |s|^\alpha) \tag{11}$$

where α is a positive parameter, and k_1, k_2 are normalization constants that ensure that f_α is a probability density of unit variance. For different values of alpha, the densities in this family exhibit different shapes. For $0 < \alpha < 2$, one obtains a sparse, super-Gaussian density (i.e., a density of positive kurtosis). For $\alpha = 2$, one obtains the Gaussian distribution, and for $\alpha > 2$, a sub-Gaussian density (i.e., a density of negative kurtosis). Thus the densities in this family can be used as examples of different non-Gaussian densities.

Using Theorem 2, one sees that in terms of asymptotic variance, an optimal contrast function for estimating an independent component whose density function equals f_α, is of the form:

$$G_{opt}(u) = |u|^\alpha \tag{12}$$

where the arbitrary constants have been dropped for simplicity. This implies roughly that for super-Gaussian (resp. sub-Gaussian) densities, the optimal contrast function is a function that grows *slower than quadratically* (resp. *faster than quadratically*). Next, recall from Section 3.1.3 that if $G(u)$ grows fast with $|u|$, the estimator becomes highly non-robust against outliers. Taking also into account the fact that most independent components encountered in practice are super-Gaussian [3, 25], one reaches the conclusion that as a general-purpose contrast function, one should choose a function G that resembles rather

$$G_{opt}(u) = |u|^{\alpha}, \text{where } \alpha < 2. \tag{13}$$

The problem with such contrast functions is, however, that they are not differentiable at 0 for $\alpha \leq 1$. Thus it is better to use approximating differentiable functions that have the same kind of qualitative behavior. Considering $\alpha = 1$, in which case one has a double exponential density, one could use instead the function $G_1(u) = \log\cosh a_1 u$ where $a_1 \geq 1$ is a constant. Note that the derivative of G_1 is then the familiar tanh function (for $a_1 = 1$). In the case of $\alpha < 1$, i.e., highly super-Gaussian independent components, one could approximate the behavior of G_{opt} for large u using a Gaussian function (with a minus sign): $G_2(u) = -\exp(-a_2 u^2/2)$, where a_2 is a constant. The derivative of this function is like a sigmoid for small values, but goes to 0 for larger values. Note that this function also fulfills the condition in Theorem 3, thus providing an estimator that is as robust as possible in the framework of estimators of type (8). As regards the constants, we have found experimentally $1 \leq a_1 \leq 2$ and $a_2 = 1$ to provide good approximations.

3.2.2 Choosing the Contrast Function in Practice

The theoretical analysis given above gives some guidelines as for the choice of G. In practice, however, there are also other criteria that are important, in particular the following two.

First, we have computational simplicity: The contrast function should be fast to compute. It must be noted that polynomial functions tend to be faster to compute than, say, the hyperbolic tangent. However, non-polynomial contrast functions could be replaced by piecewise linear approximations without losing the benefits of non-polynomial functions.

The second point to consider is the order in which the components are estimated, if one-by-one estimation is used. We can influence this order because the basins of attraction of the maxima of the contrast function have different sizes. Any ordinary method of optimization tends to first find maxima that have large basins of attraction. Of course, it is not possible to determine with certainty this order, but a suitable choice of the contrast function means that independent components with certain distributions tend to be found first. This point is, however, so application-dependent that we cannot say much in general.

Thus, we reach the following general conclusion. We have basically the following choices for the contrast function (for future use, we also give their derivatives):

$$G_1(u) = \frac{1}{a_1}\log\cosh(a_1 u), \qquad g_1(u) = \tanh(a_1 u) \tag{14}$$

$$G_2(u) = -\frac{1}{a_2}\exp(-a_2 u^2/2), \quad g_2(u) = u\exp(-a_2 u^2/2) \tag{15}$$

$$G_3(u) = \frac{1}{4}u^4, \qquad g_3(u) = u^3 \tag{16}$$

where $1 \leq a_1 \leq 2, a_2 \approx 1$ are constants, and piecewise linear approximations of (14) and (15) may also be used. The benefits of the different contrast functions may be summarized as follows:

- G_1 is a good general-purpose contrast function.
- when the independent components are highly super-Gaussian, or when robustness is very important, G_2 may be better.

- if computational overhead must be reduced, piecewise linear approximations of G_1 and G_2 may be used.
- using kurtosis, or G_3, is justified on statistical grounds only for estimating sub-Gaussian independent components when there are no outliers.

Finally, we emphasize in contrast to many other ICA methods, our framework provides estimators that work for (practically) any distributions of the independent components and for any choice of the contrast function. The choice of the contrast function is only important if one wants to optimize the performance of the method.

4 Fixed-point algorithms for ICA

4.1 Introduction

In the preceding sections, we introduced new contrast (or objective) functions for ICA based on minimization of mutual information (and projection pursuit), analyzed some of their properties, and gave guidelines for the practical choice of the function G used in the contrast functions. In practice, one also needs an algorithm for maximizing the contrast functions in (7) or (8).

A simple method to maximize the contrast function would be to use stochastic gradient descent; the constraint could be taken into account by a bigradient feedback. This leads to neural (adaptive) algorithms that are closely related related to those introduced in [24]. We show in the Appendix B how to modify the algorithms in [24] to minimize the contrast functions used in this paper.

The advantage of neural on-line learning rules is that the inputs $\mathbf{x}(t)$ can be used in the algorithm at once, thus enabling faster adaptation in a non-stationary environment. A resulting trade-off, however, is that the convergence is slow, and depends on a good choice of the learning rate sequence, i.e. the step size at each iteration. A bad choice of the learning rate can, in practice, destroy convergence. Therefore, it would important in practice to make the learning faster and more reliable. This can be achieved by the fixed-point iteration algorithms that we introduce here. In the fixed-point algorithms, the computations are made in batch (or block) mode, i.e., a large number of data points are used in a single step of the algorithm. In other respects, however, the algorithms may be considered neural. In particular, they are parallel, distributed, computationally simple, and require little memory space. We will show below that the fixed-point algorithms have very appealing convergence properties, making them a very interesting alternative to adaptive learning rules in environments where fast real-time adaptation is not necessary.

Note that our basic ICA algorithms require a preliminary sphering or whitening of the data $\mathbf{x}$, though also some versions for non-sphered data will be given. Sphering means that the original observed variable, say $\mathbf{v}$ is linearly transformed to a variable $\mathbf{x} = \mathbf{Q}\mathbf{v}$ 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. For details, see [7, 12].

4.2 Fixed-point algorithm for one unit

To begin with, we shall derive the fixed-point algorithm for one unit, with sphered data. First note that the maxima of $J_G(\mathbf{w})$ are obtained at certain optima of $E\{G(\mathbf{w}^T\mathbf{x})\}$. According to the Kuhn-Tucker conditions [29], the optima of $E\{G(\mathbf{w}^T\mathbf{x})\}$ under the constraint $E\{(\mathbf{w}^T\mathbf{x})^2\} = \|\mathbf{w}\|^2 = 1$ are obtained at points where

$$E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w} = 0 \tag{17}$$

where β is a constant that can be easily evaluated to give $\beta = E\{\mathbf{w}_0^T\mathbf{x}g(\mathbf{w}_0^T\mathbf{x})\}$, where $\mathbf{w}_0$ is the value of $\mathbf{w}$ at the optimum. Let us try to solve this equation by Newton's method. Denoting the

function on the left-hand side of (17) by F, we obtain its Jacobian matrix $JF(\mathbf{w})$ as

$$JF(\mathbf{w}) = E\{\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{I} \tag{18}$$

To simplify the inversion of this matrix, we decide to approximate the first term in (18). Since the data is sphered, a reasonable approximation seems to be $E\{\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})\} \approx E\{\mathbf{x}\mathbf{x}^T\}E\{g'(\mathbf{w}^T\mathbf{x})\} = E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{I}$. Thus the Jacobian matrix becomes diagonal, and can easily be inverted. We also approximate β using the current value of $\mathbf{w}$ instead of $\mathbf{w}_0$. Thus we obtain the following approximative Newton iteration:

$$\begin{aligned} \mathbf{w}^+ &= \mathbf{w} - [E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta] \\ \mathbf{w}^* &= \mathbf{w}^+/\|\mathbf{w}^+\| \end{aligned} \tag{19}$$

where $\mathbf{w}^*$ denotes the new value of $\mathbf{w}$, $\beta = E\{\mathbf{w}^T\mathbf{x}g(\mathbf{w}^T\mathbf{x})\}$, and the normalization has been added to improve the stability. This algorithm can be further simplified by multiplying both sides of the first equation in (19) by $\beta - E\{g'(\mathbf{w}^T\mathbf{x})\}$. This gives the following *fixed-point algorithm*

$$\boxed{\begin{aligned} \mathbf{w}^+ &= E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{w} \\ \mathbf{w}^* &= \mathbf{w}^+/\|\mathbf{w}^+\| \end{aligned}} \tag{20}$$

which was introduced in [17] using a more heuristic derivation. An earlier version (for kurtosis only) was derived as a fixed-point iteration of a neural learning rule in [23], which is where its name comes from. We retain this name for the algorithm, although in the light of the above derivation, it is rather a Newton method than a fixed-point iteration.

Due to the approximations used in the derivation of the fixed-point algorithm, one may wonder if it really converges to the right points. First of all, since only the Jacobian matrix is approximated, any convergence point of the algorithm must be a solution of the Kuhn-Tucker condition in (17). In Appendix A it is further proven that the algorithm does converge to the right extrema (those corresponding to maxima of the contrast function), under the assumption of the ICA data model. Moreover, it is proven that the convergence is quadratic, as usual with Newton methods. In fact, if the densities of the s_i are symmetric, the convergence is even cubic. The convergence proven in the Appendix is local. However, in the special case where kurtosis is used as a contrast function, i.e., if $G(u) = u^4$, the convergence is proven globally.

The above derivation also enables a useful modification of the fixed-point algorithm. It is well-known that the convergence of the Newton method may be rather uncertain. To ameliorate this, one may add a step size in (19), obtaining the *stabilized fixed-point algorithm*

$$\boxed{\begin{aligned} \mathbf{w}^+ &= \mathbf{w} - \mu[E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta] \\ \mathbf{w}^* &= \mathbf{w}^+/\|\mathbf{w}^+\| \end{aligned}} \tag{21}$$

where $\beta = E\{\mathbf{w}^T\mathbf{x}g(\mathbf{w}^T\mathbf{x})\}$ as above, and μ is a step size parameter that may change with the iteration count. Taking a μ that is much smaller than unity (say, 0.1 or 0.01), the algorithm (21) converges with much more certainty. In particular, it is often a good strategy to start with $\mu = 1$, in which case the algorithm is equivalent to the original fixed-point algorithm in (20). If convergence seems problematic, μ may then be decreased gradually until convergence is satisfactory. Note that we thus have a continuum between a Newton optimization method, corresponding to $\mu = 1$, and a gradient descent method, corresponding to a very small μ.

The fixed-point algorithms may also be simply used for the original, that is, not sphered data. Transforming the data back to the non-sphered variables, one sees easily that the following modification of the algorithm (20) works for non-sphered data:

$$\begin{aligned} \mathbf{w}^+ &= \mathbf{C}^{-1}E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{w} \\ \mathbf{w}^* &= \mathbf{w}^+/\sqrt{(\mathbf{w}^+)^T\mathbf{C}\mathbf{w}^+} \end{aligned} \tag{22}$$

where $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$ is the covariance matrix of the data. The stabilized version, algorithm (21), can also be modified as follows to work with non-sphered data:

$$\mathbf{w}^+ = \mathbf{w} - \mu[\mathbf{C}^{-1}E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta]$$
$$\mathbf{w}^* = \mathbf{w}^+/\sqrt{(\mathbf{w}^+)^T\mathbf{C}\mathbf{w}^+} \tag{23}$$

Using these two algorithms, one obtains directly an independent component as the linear combination $\mathbf{w}^T\mathbf{x}$, where $\mathbf{x}$ need not be sphered (prewhitened). These modifications presuppose, of course, that the covariance matrix is not singular. If it is singular or near-singular, the dimension of the data must be reduced, for example with PCA [7, 28].

In practice, the expectations in the fixed-point algorithms must be replaced by their estimates. The natural estimates are of course the corresponding sample means. Ideally, all the data available should be used, but this is sometimes not a good idea because the computations may become too demanding. Then the averages can be estimated using a smaller sample, whose size may have a considerable effect on the accuracy of the final estimates. The sample points should be chosen separately at every iteration. If the convergence is not satisfactory, one may then increase the sample size. A reduction of the step size μ in the stabilized version has a similar effect, as is well-known in stochastic approximation methods [24, 28].

4.3 Fixed-point algorithm for several units

The one-unit algorithm of the preceding subsection can be used to construct a system of n neurons to estimate the whole ICA transformation using the multi-unit contrast function in (8). To prevent different neurons from converging to the same maxima we must *decorrelate* the outputs $\mathbf{w}_1^T\mathbf{x}, ..., \mathbf{w}_n^T\mathbf{x}$ after every iteration. We present here three methods for achieving this. These methods do not assume that the data is sphered. If it is, the covariance matrix $\mathbf{C}$ can simply be omitted in the following formulas.

A simple way of achieving decorrelation is a deflation scheme based on a Gram-Schmidt-like decorrelation. This means that we estimate the independent components one by one. When we have estimated p independent components, or p vectors $\mathbf{w}_1, ..., \mathbf{w}_p$, we run the one-unit fixed-point algorithm for $\mathbf{w}_{p+1}$, and after every iteration step subtract from $\mathbf{w}_{p+1}$ the 'projections' $\mathbf{w}_{p+1}^T\mathbf{w}_j\mathbf{w}_j, j = 1, ..., p$ of the previously estimated p vectors, and then renormalize $\mathbf{w}_{p+1}$:

1. Let $\mathbf{w}_{p+1} = \mathbf{w}_{p+1} - \sum_{j=1}^{p} \mathbf{w}_{p+1}^T\mathbf{C}\mathbf{w}_j\mathbf{w}_j$
2. Let $\mathbf{w}_{p+1} = \mathbf{w}_{p+1}/\sqrt{\mathbf{w}_{p+1}^T\mathbf{C}\mathbf{w}_{p+1}}$ (24)

In certain applications, however, it may be desired to use a symmetric decorrelation, in which no vectors are 'privileged' over others [28]. This can be accomplished, e.g., by the classical method involving matrix square roots,

$$\text{Let } \mathbf{W} = (\mathbf{W}\mathbf{C}\mathbf{W}^T)^{-1/2}\mathbf{W} \tag{25}$$

where $\mathbf{W}$ is the matrix $(\mathbf{w}_1, ..., \mathbf{w}_n)^T$ of the vectors, and the inverse square root $(\mathbf{W}\mathbf{C}\mathbf{W}^T)^{-1/2}$ is obtained from the eigenvalue decomposition of $\mathbf{W}\mathbf{C}\mathbf{W}^T = \mathbf{E}\mathbf{D}\mathbf{E}^T$ as $(\mathbf{W}\mathbf{C}\mathbf{W}^T)^{-1/2} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{E}^T$. A simpler alternative is the following iterative algorithm,

1. Let $\mathbf{W} = \mathbf{W}/\sqrt{\|\mathbf{W}\mathbf{C}\mathbf{W}^T\|}$
Repeat 2. until convergence:
2. Let $\mathbf{W} = \frac{3}{2}\mathbf{W} - \frac{1}{2}\mathbf{W}\mathbf{C}\mathbf{W}^T\mathbf{W}$ (26)

The norm in step 1 can be almost any ordinary matrix norm, e.g., the 2-norm or the largest absolute row (or column) sum (but not the Frobenius norm). The convergence of the orthonormalization

method in (26), which may be considered a variation of Potter's formula (see [5]), is proven in the Appendix.

Finally, let us note that explicit inversion of the matrix $\mathbf{C}$ in (22) or (23) can be avoided by using the identity $\mathbf{C}^{-1} = \mathbf{W}^T\mathbf{W}$ which is valid for any decorrelating $\mathbf{W}$. This gives raise to a fixed-point algorithm in which neither sphering nor inversion of the covariance matrix is needed. In fact, such an algorithm can be considered as a fixed-point algorithm for maximum likelihood estimation of the ICA data model, see [21].

4.4 Properties of the Fixed-Point Algorithm

The fixed-point algorithm and the underlying contrast functions have a number of desirable properties when compared with existing methods for ICA.

- The convergence is cubic (or at least quadratic), under the assumption of the ICA data model (for a proof, see the convergence proof in the Appendix). This is in contrast to gradient descent methods, where the convergence is only linear. This means a very fast convergence, as has been confirmed by simulations and experiments on real data (see Section 5).
- Contrary to gradient-based algorithms, there are no step size parameters to choose (in the original fixed-point algorithm). This means that the algorithm is easy to use. Even in the stabilized version, reasonable values for the step size parameter are very easy to choose.
- The algorithm finds directly independent components of (practically) any non-Gaussian distribution using any nonlinearity g. This is in contrast to many algorithms, where some estimate of the probability distribution function has to be first available, and the nonlinearity must be chosen accordingly.
- The performance of the method can be optimized by choosing a suitable nonlinearity g. In particular, one can obtain algorithms that are robust and/or of minimum variance.
- The independent components can be estimated one by one, which is roughly equivalent to doing projection pursuit.
- The fixed-point algorithm inherits most of the advantages of neural algorithms: It is parallel, distributed, computationally simple, and requires little memory space. Stochastic gradient methods seem to be preferable only if fast adaptivity in a changing environment is required.

5 Simulation and experimental results

First, we investigated the robustness of the contrast functions. We generated four artificial source signals, two of which were sub-Gaussian, and two were super-Gaussian. The source signals were mixed using several different random matrices, whose elements were drawn from a standardized Gaussian distribution. To test the robustness of our algorithms, *four outliers* whose values were ± 10 were added in random locations. The fixed-point algorithm for sphered data was used with the three different contrast functions in eq. (14–16), and symmetric orthogonalization. Since the robust estimation of the covariance matrix is a classical problem independent of the robustness of our contrast functions, we used in this simulation a hypothetical robust estimator of covariance, which was simulated by estimating the covariance matrix from the original data without outliers. In all the runs, it was observed that the estimates based on kurtosis (16) were essentially worse than the others, and estimates using G_2 in (15) were slightly better than those using G_1 in (14). These results confirm the theoretical predictions on robustness in Section 3.

To investigate the *asymptotic variance*, i.e., efficiency, of the estimators, we performed simulations in which the 3 different contrast functions were used to estimate one independent component from a mixture of 4 identically distributed independent components. We also used three different distributions of the independent components: uniform, double exponential (or Laplace), and the distribution of the third power of a Gaussian variable. The asymptotic mean absolute deviations (which is a robustified measure of error) between the components of the obtained vectors and the correct solutions were estimated and averaged over 1000 runs for each combination of non-linearity and distribution of independent component. The results in the basic, noiseless case are depicted in Fig. 1. As one can see, the estimates using kurtosis were essentially worse for super-Gaussian independent components. Especially the strongly super-Gaussian independent component (cube of Gaussian) was estimated considerably worse using kurtosis. Only for the sub-Gaussian independent component, kurtosis was better than the other contrast functions. There was no clear difference between the performances of the contrast functions G_1 and G_2. Next, the experiments were repeated with added Gaussian noise whose energy was 10% of the energy of the independent components. The results are shown in Fig. 2. This time, kurtosis did not perform better even in the case of the sub-Gaussian density. The robust contrast functions seem to be somewhat robust against Gaussian noise as well.

We also studied the *speed of convergence* of the fixed-point algorithms. Four independent components of different distributions (two subgaussian and two supergaussian) were artificially generated, and the symmetric version of the fixed-point algorithm for sphered data was used. The data consisted of 1000 points, and the whole data was used at every iteration. We observed that for all three contrast functions, only *three* iterations were necessary, on the average, to achieve the maximum accuracy allowed by the data. This illustrates the fast convergence of the fixed-point algorithm. In fact, a comparison of our algorithm with other algorithms was performed in [13], showing that the fixed-point algorithm gives approximately the same statistical efficiency as other algorithms, but with a fraction of the computational cost.

Experiments on different kinds of real life data have also been performed using the contrast functions and algorithms introduced above. These applications include artifact cancellation in EEG and MEG [36, 37], decomposition of evoked fields in MEG [38], and feature extraction of image data [35, 25]. These experiments further validate the ICA methods introduced in this paper. A Matlab^{TM} implementation of the fixed-algorithm is available on the World Wide Web free of charge [10].

6 Conclusions

The problem of linear independent component analysis (ICA), which is a form of redundancy reduction, was addressed. Following Comon [7], the ICA problem was formulated as the search for a linear transformation that minimizes the mutual information of the resulting components. This is roughly equivalent to finding directions in which negentropy is maximized and which can also be considered projection pursuit directions [16]. The novel approximations of negentropy introduced in [19] were then used for constructing novel contrast (objective) functions for ICA. This resulted in a generalization of the kurtosis-based approach in [7, 9], and also enabled estimation of the independent components one by one. The statistical properties of these contrast functions were analyzed in the framework of the linear mixture model, and it was shown that for suitable choices of the contrast functions, the statistical properties were superior to those of the kurtosis-based approach. Next, a new family of algorithms for optimizing the contrast functions were introduced. This was the family of fixed-point algorithms that are not neural in the sense that they are non-adaptive, but share the other benefits of neural learning rules. The main advantage of the fixed-point algorithms is that their convergence can be shown to be very fast (cubic or at least quadratic). Combining the good statistical properties (e.g. robustness) of the new contrast functions, and the good algorithmic

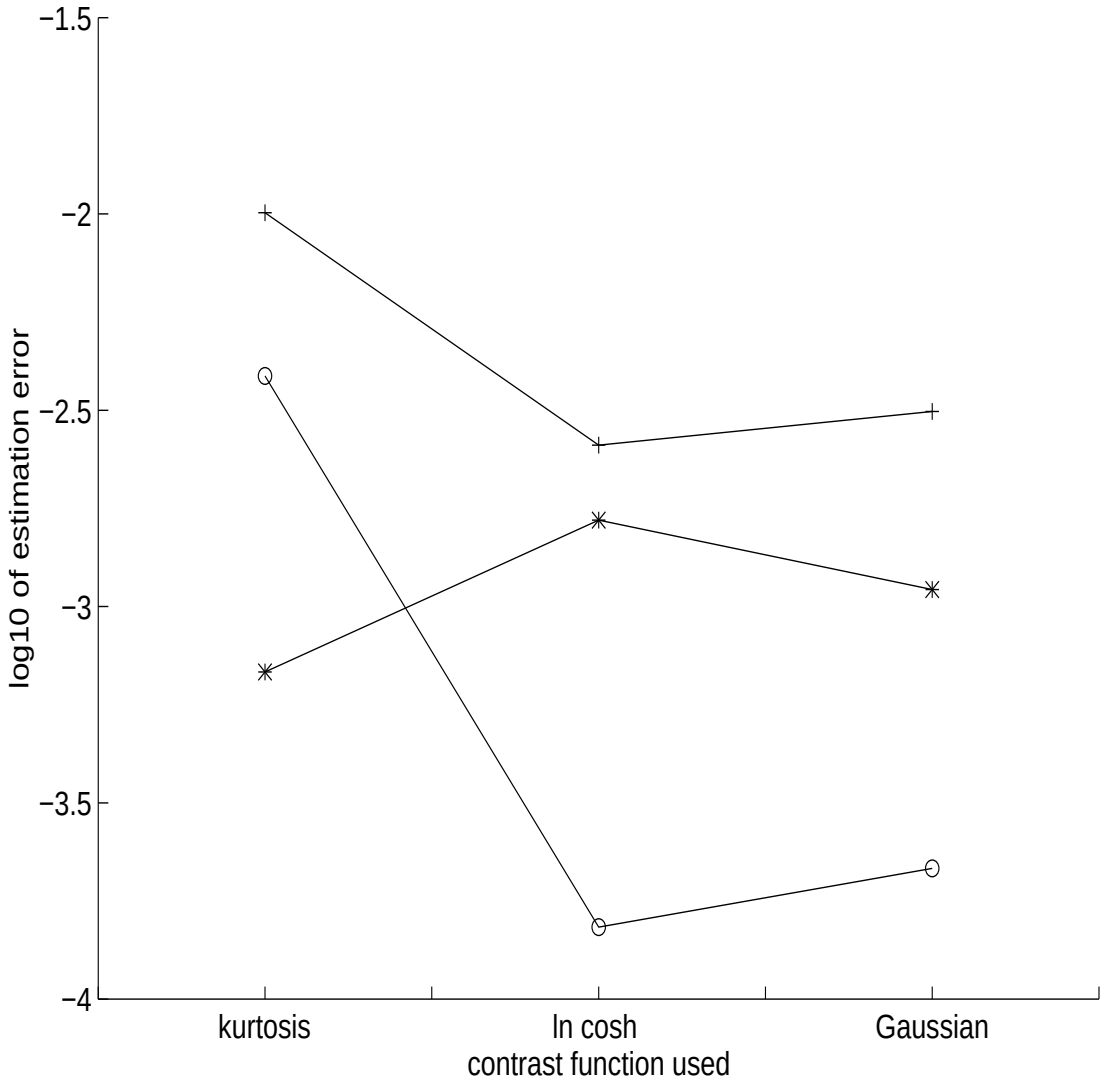

Figure 1: Finite-sample estimation errors plotted for different contrast functions and distributions of the independent components, in the noiseless case. Asterisk: uniform distribution. Plus sign: Double exponential. Circle: cube of Gaussian.

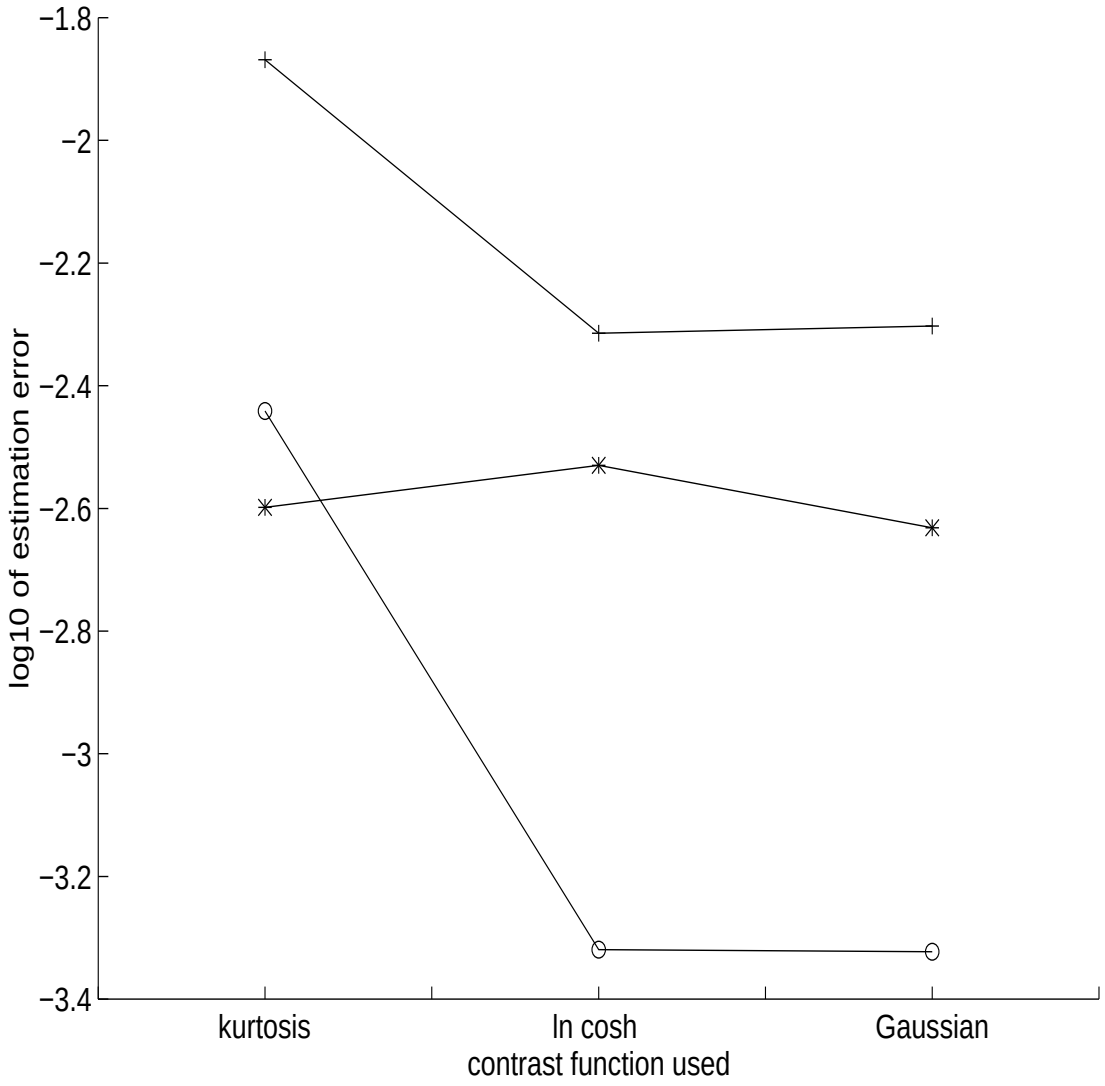

Figure 2: The noisy case. Finite-sample estimation errors plotted for different contrast functions and distributions of the independent components. Asterisk: uniform distribution. Plus sign: Double exponential. Circle: cube of Gaussian.

properties of the fixed-point algorithm, a very appealing method for ICA was obtained. Simulations as well as applications on real-life data have validated the novel contrast functions and algorithms introduced. Some extensions of the methods introduced in this paper are presented in [20], in which the problem of noisy data is addressed, and in [22], which deals with the situation where there are more independent components than observed variables.

References

[1] S. Amari, A. Cichocki, and H.H. Yang. A new learning algorithm for blind source separation. In *Advances in Neural Information Processing 8*, pages 757–763. MIT Press, Cambridge, MA, 1996.

[2] H. B. Barlow. Possible principles underlying the transformations of sensory messages. In W. A. Rosenblith, editor, *Sensory Communication*, pages 217–234. MIT Press, 1961.

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

[4] A.J. Bell and T.J. Sejnowski. The 'independent components' of natural scenes are edge filters. *Vision Research*, 37:3327–3338, 1997.

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

[6] A. Cichocki and R. Unbehauen. *Neural Networks for Signal Processing and Optimization.* Wiley, 1994.

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

[8] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, 1991.

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

[10] The FastICA MATLAB package. Available at `http://www.cis.hut.fi/projects/ica/fastica/`.

[11] J. H. Friedman and J. W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Trans. of Computers*, c-23(9):881–890, 1974.

[12] J.H. Friedman. Exploratory projection pursuit. *J. of the American Statistical Association*, 82(397):249–266, 1987.

[13] X. Giannakopoulos, J. Karhunen, and E. Oja. Experimental comparison of neural ICA algorithms. In *Proc. Int. Conf. on Artificial Neural Networks (ICANN'98)*, pages 651–656, Skövde, Sweden, 1998.

[14] F.R. Hampel, E.M. Ronchetti, P.J. Rousseuw, and W.A. Stahel. *Robust Statistics.* Wiley, 1986.

[15] H. H. Harman. *Modern Factor Analysis.* University of Chicago Press, 2nd edition, 1967.

[16] P.J. Huber. Projection pursuit. *The Annals of Statistics*, 13(2):435–475, 1985.

[17] A. Hyvärinen. A family of fixed-point algorithms for independent component analysis. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'97)*, pages 3917–3920, Munich, Germany, 1997.

[18] A. Hyvärinen. One-unit contrast functions for independent component analysis: A statistical analysis. In *Neural Networks for Signal Processing VII (Proc. IEEE Workshop on Neural Networks for Signal Processing)*, pages 388–397, Amelia Island, Florida, 1997.

[19] A. Hyvärinen. New approximations of differential entropy for independent component analysis and projection pursuit. In *Advances in Neural Information Processing Systems 10*, pages 273–279. MIT Press, 1998.

[20] A. Hyvärinen. Fast independent component analysis with noisy data using gaussian moments. In *Proc. Int. Symp. on Circuits and Systems*, Orlando, Florida, 1999. To appear.

[21] A. Hyvärinen. The fixed-point algorithm and maximum likelihood estimation for independent component analysis. *Neural Processing Letters*, 1999. To appear.

[22] A. Hyvärinen, R. Cristescu, and E. Oja. A fast algorithm for estimating overcomplete ICA bases for image windows. In *Proc. Int. Joint Conf. on Neural Networks*, Washington, D.C., 1999.

[23] A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9(7):1483–1492, 1997.

[24] A. Hyvärinen and E. Oja. Independent component analysis by general nonlinear Hebbian-like learning rules. *Signal Processing*, 64(3):301–313, 1998.

[25] A. Hyvärinen, E. Oja, P. Hoyer, and J. Hurri. Image feature extraction by sparse coding and independent component analysis. In *Proc. Int. Conf. on Pattern Recognition (ICPR'98)*, pages 1268–1273, Brisbane, Australia, 1998.

[26] M.C. Jones and R. Sibson. What is projection pursuit ? *J. of the Royal Statistical Society, ser. A*, 150:1–36, 1987.

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

[28] J. Karhunen, E. Oja, L. Wang, R. Vigario, and J. Joutsensalo. A class of neural networks for independent component analysis. *IEEE Trans. on Neural Networks*, 8(3):486–504, 1997.

[29] D. G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, 1969.

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

[31] E. Oja. The nonlinear PCA learning rule in independent component analysis. *Neurocomputing*, 17(1):25–46, 1997.

[32] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[33] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, 3rd edition, 1991.

[34] D.-T. Pham, P. Garrat, and C. Jutten. Separation of a mixture of independent sources through a maximum likelihood approach. In *Proc. EUSIPCO*, pages 771–774, 1992.

[35] J. H. van Hateren and A. van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proc. Royal Society ser. B*, 265:359–366, 1998.

[36] R. Vigário. Extraction of ocular artifacts from EEG using independent component analysis. *Electroenceph. clin. Neurophysiol.*, 103(3):395–404, 1997.

[37] R. Vigário, V. Jousmäki, M. Hämäläinen, R. Hari, and E. Oja. Independent component analysis for identification of artifacts in magnetoencephalographic recordings. In *Advances in Neural Information Processing 10 (Proc. NIPS'97)*, pages 229–235, Cambridge, MA, 1998. MIT Press.

[38] R. Vigário, J. Särelä, and E. Oja. Independent component analysis in wave decomposition of auditory evoked fields. In *Proc. Int. Conf. on Artificial Neural Networks (ICANN'98)*, pages 287–292, Skövde, Sweden, 1998.

A Appendix: Proofs

A.1 Proof of convergence of algorithm (20)

The convergence is proven under the assumptions that first, the data follows the ICA data model (2) and second, that the expectations are evaluated exactly. We must also make the following technical assumption:

$$E\{s_i g(s_i) - g'(s_i)\} \neq 0, \text{for any } i \tag{27}$$

which can be considered a generalization of the condition, valid when we use kurtosis as contrast, that the kurtosis of the independent components must be non-zero. If (27) is true for a subset of independent components, we can estimate just those independent components.

To begin with, make the change of variable $\mathbf{z} = \mathbf{A}^T\mathbf{w}$, as above, and assume that $\mathbf{z}$ is in the neighbourhood of a solution (say, $z_1 \approx 1$ as above). As shown in proof of Theorem 1 (see [24]), the change in z_1 is then of a lower order than the change in the other coordinates, due to the constraint $\|\mathbf{z}\| = 1$. Then we can expand the terms in (20) using a Taylor approximation for g and g', first obtaining

$$g(\mathbf{z}^T\mathbf{s}) = g(z_1 s_1) + g'(z_1 s_1)\mathbf{z}_{-1}^T\mathbf{s}_{-1} + \frac{1}{2}g''(z_1 s_1)(\mathbf{z}_{-1}^T\mathbf{s}_{-1})^2 \tag{28}$$

$$+\frac{1}{6}g'''(z_1 s_1)(\mathbf{z}_{-1}^T\mathbf{s}_{-1})^3 + O(\|\mathbf{z}_{-1}\|^4) \tag{29}$$

and then

$$g'(\mathbf{z}^T\mathbf{s}) = g'(z_1 s_1) + g''(z_1 s_1)\mathbf{z}_{-1}^T\mathbf{s}_{-1} \tag{30}$$

$$+\frac{1}{2}g'''(z_1 s_1)(\mathbf{z}_{-1}^T\mathbf{s}_{-1})^2 + O(\|\mathbf{z}_{-1}\|^3) \tag{31}$$

where $\mathbf{z}_{-1}$ and $\mathbf{s}_{-1}$ are the vectors $\mathbf{z}$ and $\mathbf{s}$ without their first components. Thus we obtain, using the independence of the s_i, and doing some tedious but straight-forward algebraic manipulations,

$$z_1^+ = E\{s_1 g(z_1 s_1) - g'(z_1 s_1)\} + O(\|z_{-1}\|^2) \tag{32}$$

$$z_i^+ = \frac{1}{2}\,\text{skew}\,(s_i)E\{g''(s_1)\}z_i^2$$

$$+\frac{1}{6}\,\text{kurt}\,(s_i)E\{g'''(s_1)\}z_i^3 + O(\|z_{-1}(\|^4), \text{for } i > 1 \tag{33}$$

We obtain also

$$\mathbf{z}^* = \mathbf{z}^+/\|\mathbf{z}^+\| \tag{34}$$

This shows clearly that under the assumption (27), the algorithm converges (locally) to such a vector $\mathbf{z}$ that $z_1 = \pm 1$ and $z_i = 0$ for $i > 1$. This means that $\mathbf{w} = (\mathbf{A}^T)^{-1}\mathbf{z}$ converges, up to the sign, to one of the rows of the inverse of the mixing matrix $\mathbf{A}$, which implies that $\mathbf{w}^T\mathbf{x}$ converges to one of the s_i. Moreover, if $E\{g''(s_1)\} = 0$, i.e. if the s_i has a symmetric distribution, as is usually the case, (33) shows that the convergence is cubic. In other cases, the convergence is quadratic. In addition, if $G(u) = u^4$, the local approximations above are exact, and the convergence is global.

A.2 Proof of convergence of (26)

Denote by $\mathbf{W}_+$ the result of applying once the iteration step 2 in (26) on $\mathbf{W}$. Let $\mathbf{WCW}^T = \mathbf{EDE}^T$ be the eigenvalue decomposition of $\mathbf{WCW}^T$. Then we have

$$\mathbf{W}_+\mathbf{C}\mathbf{W}_+^T = \frac{9}{4}\mathbf{EDE}^T - \frac{3}{2}\mathbf{ED}^2\mathbf{E}^T + \frac{1}{4}\mathbf{ED}^3\mathbf{E}^T \tag{35}$$

$$= \mathbf{E}(\frac{9}{4}\mathbf{D} - \frac{3}{2}\mathbf{D}^2 + \frac{1}{4}\mathbf{D}^3)\mathbf{E}^T \tag{36}$$

Note that due to the normalization, i.e. division of $\mathbf{W}$ by $\sqrt{\|\mathbf{WCW}^T\|}$, all the eigenvalues of $\mathbf{WCW}^T$ are in the interval $[0, 1]$. Now, according to (35), for every eigenvalue of $\mathbf{WCW}^T$, say λ_i, $\mathbf{W}_+\mathbf{C}\mathbf{W}_+^T$ has a corresponding eigenvalue $h(\lambda_i)$ where $h(.)$ is defined as:

$$h(\lambda) = \frac{9}{4}\lambda - \frac{3}{2}\lambda^2 + \frac{1}{4}\lambda^3 \tag{37}$$

Thus, after k iterations, the eigenvalues of $\mathbf{WCW}^T$ are obtained as $h(h(h(...h(\lambda_i))))$, where h is applied k times on the λ_i, which are the eigenvalues of $\mathbf{WCW}^T$ for the original matrix before the iterations. Now, we have always $h(\lambda) > \lambda$ for $0 < \lambda < 1$. It is therefore clear that all the eigenvalues of $\mathbf{WCW}^T$ converge to 1, which means that $\mathbf{WCW}^T \to \mathbf{I}$, Q.E.D. Moreover, it is not difficult to see that the convergence is quadratic.

B Appendix: Adaptive neural algorithms

Let us consider sphered data only. Taking the instantaneous gradient of the approximation of negentropy in (7) with respect to $\mathbf{w}$, and taking the normalization $\|\mathbf{w}\|^2 = 1$ into account, one obtains the following Hebbian-like learning rule

$$\Delta\mathbf{w} \propto r\mathbf{x}g(\mathbf{w}^T\mathbf{x}), \text{ normalize } \mathbf{w} \tag{38}$$

where $r = E\{G(\mathbf{w}^T\mathbf{x})\} - E\{G(\nu)\}$. This is equivalent to the learning rule in [24], except that the self-adaptation constant r is different.

To find the whole n-dimensional transform $\mathbf{s} = \mathbf{W}\mathbf{x}$, one can then use a network of n neurons, each of which learns according to eq. (38). Of course, some kind of feedback is then necessary. In [24], it was shown how to add a bigradient feedback to the learning rule. Denoting by $\mathbf{W} = (\mathbf{w}_1, ..., \mathbf{w}_n)^T$ the weight matrix whose rows are the weight vectors $\mathbf{w}_i$ of the neurons, we obtain:

$$\begin{aligned}\mathbf{W}(t+1) = \mathbf{W}(t) + \mu(t)\text{diag}(r_i(t))g(\mathbf{W}(t)\mathbf{x}(t))\mathbf{x}(t)^T \\ +\frac{1}{2}(\mathbf{I} - \mathbf{W}(t)\mathbf{W}(t)^T)\mathbf{W}(t)\end{aligned} \tag{39}$$

where $\mu(t)$ is the learning rate sequence, and the function $g(.) = G'(.)$ is applied separately on every component of the vector $\mathbf{W}(t)\mathbf{x}(t)$. In this most general version of the learning rule, the $r_i, i = 1...n$ are estimated separately for each neuron, as given above (see also [24]). They may also be fixed using prior knowledge.