Independent Component Analysis by Minimization of Mutual Information

Aapo Hyvärinen

Helsinki University of Technology
Department of Computer Science and Engineering
Laboratory of Computer and Information Science

Report A46

Otaniemi 1997

Independent Component Analysis by Minimization of Mutual Information

Aapo Hyvärinen

Helsinki University of Technology
Department of Computer Science and Engineering
Laboratory of Computer and Information Science
Rakentajanaukio 2 C, FIN-02150 Espoo, FINLAND

aapo.hyvarinen@hut.fi
http://www.cis.hut.fi/~aapo

Report A46
August 1997

ISBN 951-22-3720-2
ISSN 1455-0784

Independent Component Analysis by Minimization of Mutual Information

Aapo Hyvärinen
Helsinki University of Technology
Laboratory of Computer and Information Science
P.O. Box 2200, FIN-02015 HUT, Finland
Email: aapo.hyvarinen@hut.fi

September 8, 1997

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, the linear version of the ICA problem is approached from an information-theoretic viewpoint, using Comon's framework of minimizing mutual information of the components. Using maximum entropy approximations of differential entropy, we introduce a family of new contrast (objective) functions for ICA, which can also be considered 1-D projection pursuit indexes. The statistical properties of the estimators based on such contrast functions are analyzed under the assumption of the linear mixture model. It is shown how to choose optimal contrast functions according to different criteria. Novel algorithms for maximizing the contrast functions are then introduced. Hebbian-like learning rules are shown to result from gradient descent methods. Finally, in order to speed up the convergence, a family of fixed-point algorithms for maximization of the contrast functions is introduced.

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. It is important for subsequent analysis of the data, whether it be pattern recognition, data compression, de-noising, visualization or anything else, that the data is represented in a manner that facilitates the analysis. 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. It is assumed throughout the paper that the number of components is not larger than the number of observed variable, i.e., $n \leq m$. Then the problem is to determine a constant matrix $\mathbf{M}$ so that the linear transformation of the observed variables

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

has some suitable properties. Several principles and methods have been developed to find such a linear representation, including principal component analysis [27], factor analysis [13], projection pursuit [11, 15], independent component analysis [23], 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.

The topic of this paper is one of the methods for finding a linear representation, which is independent component analysis (ICA) [8, 23]. 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* [23], 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 signal sources are 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, 24], 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, 31] 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 [15, 11] (see Section 2.4).

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 (Section 2). Then a new family of contrast functions for ICA, based on maximum entropy approximations of differential entropy and mutual information, is introduced (Section 3). The behavior of the estimators based on maximizing such contrast functions is then evaluated in the framework of the linear mixture model of blind source separation (Section 4). In particular, it is shown that it is possible to obtain estimators that have statistical properties superior to those of the well-known cumulant-based approach. The practical choice of the contrast function is discussed, based on the statistical criteria together with some numerical and pragmatic criteria (Section 5). Novel algorithms for maximization the contrast functions, which corresponds to (approximate) minimization of mutual information, are then introduced (Section

6). The optimization may be performed either by Hebbian-like learning rules, or by a simple fixed-point algorithm that has very appealing convergence properties. Simulations confirming the usefulness of the novel contrast functions and algorithms are reported in Section 7, together with references to real-life experiments using these methods. Some conclusions are drawn in Section 8.

2 Independent Component Analysis by Minimization of Mutual Information

2.1 The ICA data model

One popular way of formulating the ICA problem is to consider the parametric estimation of the following generative model for the data [23]:

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

where $\mathbf{x}$ is an observed m-dimensional vector, $\mathbf{s}$ is an n-dimensional (latent) random vector whose components are assumed mutually independent, $\mathbf{A}$ is a constant $m \times n$ matrix to be estimated, and it is usually assumed that the dimensions of $\mathbf{x}$ and $\mathbf{s}$ are equal, i.e., $m = n$. A noise vector may also be present. The matrix $\mathbf{M}$ defining the transformation as in (1) is then obtained as the (pseudo)inverse of the estimate of the matrix $\mathbf{A}$. This formulation is basically the same as classical factor analysis [13], with the crucial difference that the independent components s_i are here assumed to be non-Gaussian. Indeed, the non-Gaussianity is necessary for the identifiability of the model (2), together with the assumption $n \leq m$ (for details, see [8]). This formulation, originally proposed for blind source separation [23], simplifies the ICA problem considerably, and has therefore been adopted by most researchers in ICA or blind source separation [5, 6, 7, 23, 25, 28]. However, such a model seems to be realistic only in some situations, like blind source separation. To achieve a more general method that can be applied in a wider range of applications, for example, in feature extraction and exploratory data analysis, we decide *not* to assume the data model (2) in this paper. Instead, we make no special assumptions on the data, and determine the transformation $\mathbf{M}$ in (1) according to suitable information-theoretic criteria. In Section 4, we show that the resulting method is also an adequate way of estimating the data model in (2).

2.2 Information-theoretic concepts

To obtain a more general formulation for ICA than the one offered by the ICA data model (2), we need to use information-theoretic criteria, following [8]. In this subsection, we recall the basic information-theoretic definitions and properties needed. The fundamental information-theoretic concept for continuous variables is differential entropy. The differential entropy H of a random vector $\mathbf{y}$ with density $f(.)$ is defined as [32]:

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

Differential entropy is not invariant for scale transformations. This lack of invariance is unpractical for many purposes. Therefore one defines the negentropy J, or negative normalized entropy, as follows

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

where $\mathbf{y}_{gauss}$ is a Gaussian random variable of the same covariance matrix as $\mathbf{y}$. Negentropy is invariant for invertible linear transformations. It is also always non-negative, and is zero if and only if $\mathbf{y}$ has a Gaussian distribution [8]. Negentropy has two main interpretations. First, it can be interpreted as the amount of structure of the distribution of $\mathbf{y}$. It is largest when the distribution is clearly concentrated on certain values, i.e., when the variable is clearly clustered, or has a sparse distribution. Second, it is a measure of the 'non-Gaussianity' of $\mathbf{y}$. These interpretations are of course related, as the Gaussian distribution can be considered the least structured of all distributions.

Using the concept of differential entropy, one then defines the mutual information I between m (scalar) random variables, $y_i, i = 1...m$ as follows

$$I(y_1, y_2, ..., y_m) = \sum_{i=1}^{m} H(y_i) - H(\mathbf{y}). \tag{5}$$

The mutual information is a natural measure of the dependence between random variables. It is always non-negative, and zero if and only if the variables are statistically independent. The mutual information takes into account the whole dependence structure of the variables, and not only the covariance, like PCA and related methods. It is particularly interesting to express mutual information using negentropy [8]:

$$I(y_1, y_2, ..., y_m) = J(\mathbf{y}) - \sum_i J(y_i) + \frac{1}{2} \log \frac{\prod c_{ii}}{\det \mathbf{C}^y} \tag{6}$$

where $\mathbf{C}^y$ is the covariance matrix of $\mathbf{y}$, and the c_{ii} are its diagonal elements. If the y_i are *uncorrelated*, the third term is 0, and we thus obtain

$$I(y_1, y_2, ..., y_m) = J(\mathbf{y}) - \sum_i J(y_i). \tag{7}$$

This fundamental relation will be used extensively in what follows.

2.3 Defining ICA by Mutual Information

Now we are ready to define ICA by information-theoretic principles (this definition follows [8]). To begin with, we consider only invertible transformations of $\mathbf{x}$, i.e., we set $n = m$ in (1). Since mutual information is the natural information-theoretic measure of the independence of random variables, we decide to use it as the criterion for finding the ICA transform. Thus we define the ICA of a random vector $\mathbf{x}$ as an invertible transformation $\mathbf{s} = \mathbf{M}\mathbf{x}$ as in (1) where the matrix $\mathbf{M}$ is determined so that the *mutual information of the transformed components s_i is minimized*. This choice can also be motivated using the data model (2): it can be shown [8] that the mutual information corresponds to the Kullback-Leibler divergence that should be minimized to estimate the matrix $\mathbf{A}$. 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.

Because negentropy is invariant for invertible linear transformations [8], it is now obvious from (7) that finding an invertible transformation $\mathbf{M}$ that minimizes the mutual information is roughly equivalent to *finding directions in which the negentropy is maximized*. (More precisely, it is roughly equivalent to finding 1-D subspaces such that the projections in those subspaces have maximum negentropy). This equivalence is valid strictly when the *components s_i are constrained to be uncorrelated*, and $n = m$, which means that $\mathbf{M}$ is invertible.

Intuitively, this result means that one must find uncorrelated directions in which the distribution of the data is as non-Gaussian as possible. The constraint of uncorrelatedness is in fact not necessary, but simplifies the computations considerably, as one can then use the simpler form in (7) instead of the more complicated form in (6). Therefore, the constraint of uncorrelatedness of the s_i is adopted in this paper.

2.4 Relation to Projection Pursuit

It is interesting to note how this formulation of ICA makes explicit the connection between ICA and projection pursuit. Projection pursuit [10, 11, 15, 22] is a technique developed in statistics for finding 'interesting' projections of multidimensional data. Such projections can then be used for optimal visualization of the data, and for such purposes as density estimation and regression. In basic (1-D) projection pursuit, we try to find directions such that the projections of the data in those directions have 'interesting' distributions, i.e., display some structure. It has been argued by Huber [15] and by Jones and Sibson [22] that the Gaussian distribution is the least interesting one, and that the most interesting directions are those that show the least Gaussian distribution. This is exactly what is attained by finding directions in which negentropy in maximized.

Thus, in the general formulation, ICA can be considered a variant of projection pursuit. All the objective functions and algorithms presented below could also be called projection pursuit 'indexes' and algorithms. In particular, the projection pursuit formulation gives a meaningful interpretation of the situation where $n < m$, i.e., one computes a smaller number of independent components s_i than the dimension of the original variable $\mathbf{x}$ is. Such a transformation can be interpreted as a hybrid of projection pursuit and ICA. (Such a situation could also be interpreted using the ICA data model (2), and saying that one only estimates n independent components instead of m.). The formulation using mutual information, on the other hand, is then no more strictly valid, because it assumes that the transformation is invertible.

3 Contrast Functions through Approximations of Negentropy

3.1 Cumulant-based approximations of negentropy

Above it was shown how the search for the ICA transform can be reduced to the search for directions in which negentropy is maximized (or normalized entropy is minimized). Thus we have simplified the n-dimensional problem of minimizing mutual information to the separate maximization of n 1-D negentropies. Unfortunately, the estimation of negentropy is still a difficult problem. To use the definition of differential entropy in Eq. (3), an estimate of the density is needed. Therefore, simpler approximations of (neg)entropy have been proposed both in the context of projection pursuit [22], and independent component analysis [8, 1].

Usually, negentropy has been approximated using the so-called cumulants of the distribution. Assume, for simplicity, that the 1-D variable x is centered and normalized so that it has zero mean and unit variance. The third cumulant, or skewness, of x can then be defined as $\text{skew}\,(x) = E\{x^3\}$. It is a measure of the lack of symmetry of the distribution of x. The fourth cumulant, or kurtosis, of x can be defined as $\text{kurt}\,(x) = E\{x^4\} - 3$. Using these two cumulants, the following approximation for negentropy was derived in [22]:

$$J(x) \approx \frac{1}{12}(\,\text{skew}\,(x))^2 + \frac{1}{48}(\,\text{kurt}\,(x))^2. \tag{8}$$

Very similar approximations were derived later in [8] and [1]. Often the distributions are assumed to be symmetric, which implies that skewness is zero and Eq. (8) simplifies to $J(x) \propto (\text{ kurt } (x))^2$.

However, such cumulant-based methods often provide a rather poor approximation of entropy. Intuitively, there are two main reasons for this. Firstly, finite-sample estimators of higher-order cumulants are highly sensitive to outliers: their values may depend on only a few, possibly erroneous, observations with large values [15]. This means that outliers may completely determine the estimates of cumulants, thus making them useless. Secondly, even if the cumulants were estimated perfectly, they measure mainly the tails of the distribution, and are largely unaffected by structure near the centre of the distribution [11]. In contrast, entropy measures mainly the structure of the distribution near its centre. See also the analysis of Section 4, which shows that in the framework of the ICA data model (2), the performance properties of the cumulant-based estimators are quite suboptimal. The main argument in favor of the cumulant-based approach is that it can be shown to be asymptotically optimal for sums of i.i.d. variables [8]. However, this is by definition not the case in directions near the directions of the independent components, and therefore this argument does not seem compelling.

3.2 Maximum entropy approximation

To avoid the problems encountered with the preceding approximations of negentropy, new approximations were developed in [18]. These approximation were based on the maximum-entropy principle.

Assume that the scalar random variable x is of zero mean and unit variance, and that the information available on its density $f(.)$ is of the form

$$\int f(\xi) G_i(\xi) d\xi = c_i, \text{ for } i = 1, ..., p, \tag{9}$$

which means in practice that we have estimated the expectations $E\{G_i(x)\}$ of p different functions of x. Since we are not assuming any model for the random variable x, the estimation of the entropy of x using this information is not a well-defined problem: there exist an infinite number of distributions for which the constraints in (9) are fulfilled, but whose entropies are very different from each other. In particular, the differential entropy reaches $-\infty$ in the limit where x takes only a finite number of values (i.e., is discrete-valued). A simple solution to this dilemma is the maximum entropy method. This means that we compute the *maximum* entropy that is compatible with the constraints, or measurements, in (9); this is a well-defined problem. The maximum entropy, or further approximations thereof, can then be used as an approximation of the entropy of x. Note that maximum entropy corresponds to minimum negentropy.

In [18], an approximation of the maximum entropy compatible with the measurements in (9) was derived as

$$J(x) \approx \sum_{i=1}^{p} k_i [E\{G_i(x)\} - E\{G_i(\nu)\}]^2, \tag{10}$$

where k_i are some positive constants, and ν is a Gaussian variable of zero mean and unit variance (i.e., standardized). The variable x is assumed to be of zero mean and unit variance, and the functions G_i must be orthogonal in a certain sense [18]. Note that even in cases where this approximation is not very accurate, (10) can be used to construct a projection pursuit index that is consistent in the sense that it is always non-negative, and equal to zero

if x has a Gaussian distribution. In the case where the additional information consists of a single expectation, which means $p = 1$, the approximation becomes

$$J(x) \propto [E\{G(x)\} - E\{G(\nu)\}]^2 \tag{11}$$

for practically any non-quadratic function G. This is clearly a generalization of the approximation in (8), if x is symmetric. Indeed, taking $G(x) = x^4$, one obtains the square of kurtosis, or the second term in (8). Using two functions, one even and one odd, direct generalizations of (8) are obtained. In most cases, however, distributions used in ICA are symmetric, and the first term, measuring skewness, is zero. Therefore, the simple approximation in (11) seems to be enough for most ICA applications. In [18] it was shown that for suitable choices of G, the approximation given by (11) is clearly superior to the approximation given by the kurtosis-based approach. The choice of the function G is deferred to Section 5.

3.3 Novel contrast functions for ICA

The approximation of negentropy given above in (11) can be used to define new objective functions for determining the ICA transform, according to the definition of Section 2.3. Basically, what we want to do is maximize the following function that approximates negentropy:

$$J_G(\mathbf{w}) = [E\{G(\mathbf{w}^T\mathbf{x})\} - E\{G(\nu)\}]^2 \tag{12}$$

where $\mathbf{w}$ is an m-dimensional (weight) vector, it is assumed that $E\{(\mathbf{w}^T\mathbf{x})^2\} = 1$, and ν is a standardized Gaussian random variable. Maximizing J_G allows one to find *one* independent component, or projection pursuit direction. Therefore, we call (12) a one-unit contrast function; for simplicity, we also sometimes call G the contrast function.

The one-unit contrast function in (12) can then be simply extended to compute the whole matrix $\mathbf{M}$ in (1). To do this, recall from (7) 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:

$$\begin{aligned} &\text{maximize } \sum_{i=1}^{n} J_G(\mathbf{w}_i) \text{ wrt. } \mathbf{w}_i, i = 1, ..., n \\ &\text{under constraint } E\{(\mathbf{w}_k^T\mathbf{x})(\mathbf{w}_j^T\mathbf{x})\} = \delta_{jk} \end{aligned} \tag{13}$$

where at the maximum, every vector $\mathbf{w}_i, i = 1, .., n$ gives one of the rows of the matrix $\mathbf{M}$, and the ICA transformation is then given by $\mathbf{s} = \mathbf{M}\mathbf{x}$. The ICA transformation given by (13) is thus one that approximately minimizes the mutual information of the resulting components $s_i = \mathbf{w}_i^T\mathbf{x}$. In the projection pursuit interpretation, on the other hand, Eq. (13) gives n projection pursuit directions that are constrained to be decorrelated.

Thus we have generalized the kurtosis-based approach of Comon [8] (and related to those in [22, 1]) to a wide class of non-quadratic contrast functions. This generalization was motivated by the desire to obtain better approximations of negentropy and mutual information, which give better estimators of the ICA transformation, as defined by minimization of mutual information. The next section analyzes the statistical properties of the resulting estimators in the framework of the ICA data model, and shows that the generalization gives better estimators in that framework as well.

4 Behavior under the ICA data model

4.1 Introduction

In this section, we analyze the behavior of the estimators given above when the data follows the ICA data model (2). We consider three fundamental properties (consistency, asymptotic variance, and robustness) of the estimator (13) of the matrix $\mathbf{A}$ in the ICA data model (2). For simplicity, we consider only the estimation of a single independent component, and neglect the effects of decorrelation. Let us denote by $\hat{\mathbf{m}}$ a vector obtained by maximizing J_G in (12), which is equivalent to using (13) for $n = 1$. The vector $\hat{\mathbf{m}}$ is thus an estimator of a row of the matrix $\mathbf{A}^{-1}$.

4.2 Consistency

First of all, since the approximation of mutual information in (11) may be rather crude, one may wonder if the estimator obtained by maximizing (12) really converges to the direction of one of the independent components, assuming the ICA data model. It can be proven that this is so, under rather mild conditions. In particular, we have the following theorem on the consistency of the estimators:

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*

$$E\{s_i g(s_i) - g'(s_i)\}[E\{G(s_i)\} - E\{G(\nu)\}] > 0 \tag{14}$$

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

This theorem is proven in the Appendix; it could also be considered a corollary of the theorem in [21]. Note that if $\mathbf{w}$ equals the i-th row of $\mathbf{A}^{-1}$, the linear combination equals the i-th independent component: $\mathbf{w}^T\mathbf{x} = \pm s_i$. 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].

As for the estimation of the whole matrix $\mathbf{A}^{-1}$, the above theorem implies that the estimator obtained from (13) is (locally) consistent, since the independent components correspond to different local maxima of J_G in uncorrelated directions, and the constraint in (13) is nothing else than a constraint of decorrelation (with normalization).

4.3 Asymptotic variance

In practice, one usually has only a finite sample of N observations of the vector $\mathbf{x}$. Therefore, the expectations in the definition of J_G are in fact replaced by sample means. This results in certain errors in the estimator $\hat{\mathbf{m}}$, and it is desired to make these errors as small as possible. A classical measure of this error is asymptotic (co)variance, which means the limit of the covariance matrix of $\hat{\mathbf{m}}\sqrt{N}$ as $N \to \infty$. (It is here assumed that the estimators and the distributions concerned fulfill some regularity conditions which imply that $\hat{\mathbf{m}}\sqrt{N}$ is asymptotically Gaussian [14]). This gives an approximation of the mean-square error of $\hat{\mathbf{m}}$. Comparison of, say, the traces of the asymptotic variances of two estimators enables direct comparison of the accuracy of two estimators. Fortunately, one can solve analytically for the asymptotic variance of $\hat{\mathbf{m}}$, obtaining the following theorem:

Theorem 2 *The trace of the asymptotic variance of* $\hat{\mathbf{m}}$ *as defined by the maximization in (13) for the estimation of a single independent component* s_i *in the model (2), equals*

$$V_G = C(\mathbf{A}) \frac{E\{g^2(s_i)\} - (E\{s_i g(s_i)\})^2}{(E\{s_i g(s_i) - g'(s_i)\})^2}, \tag{15}$$

where g *is the derivative of* G*, and* $C(\mathbf{A})$ *is a constant that depends only on* $\mathbf{A}$*.*

This theorem is proven in the Appendix.

Thus the comparison of the asymptotic variances of two estimators of the form in (13), but for two different contrast functions G, boils down to a comparison of the V_G's. In particular, one can use variational calculus to find a G that minimizes V_G. Thus one obtains the following theorem, which is proven in the Appendix:

Theorem 3 *The trace of the asymptotic variance of* $\hat{\mathbf{m}}$ *is minimized when* G *is of the form*

$$G_{opt}(u) = k_1 \log f_i(u) + k_2 u^2 + k_3 \tag{16}$$

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 for several units by the maximum likelihood approach [33], or the infomax approach [3]. Almost identical results have also been obtained in [6] for another multi-unit algorithm. The theorems above treat, however, the one-unit case instead of the multi-unit case, and are thus applicable to estimation of a subset of the independent components, and may also be applicable to blind deconvolution (see [21]).

4.4 Robustness

Another very attractive property of an estimator is robustness against outliers [12, 14]. This means that single, highly erroneous observations do not have much influence on the estimator.

In this section, we shall treat the question: How does the robustness of the estimator $\hat{\mathbf{m}}$ depend on the choice of the function G? Note that the robustness of $\hat{\mathbf{m}}$ depends also on the method of estimation used in constraining the variance of $\mathbf{w}^T\mathbf{x}$ to equal unity in (13). This is a problem independent of the choice of G. In the following, we assume that this constraint is implemented in a robust way. In particular, we assume that the data is sphered (whitened) in a robust manner (for details on sphering, see Section 6.2), in which case the constraint reduces to $\|\mathbf{w}\| = 1$. Several robust estimators of the variance of $\mathbf{w}^T\mathbf{x}$ or of the covariance matrix of $\mathbf{x}$ are presented in the literature; see [12, 14].

The robustness of the estimator $\hat{\mathbf{m}}$ in (13) can be analyzed using the theory of M-estimators [12, 14]. Without going into technical details, the definition of an M-estimator can be formulated as follows: an estimator is called an M-estimator if it is defined as the solution $\hat{\theta}$ for θ of

$$E\{\psi(\mathbf{x}, \theta)\} = 0 \tag{17}$$

where $\mathbf{x}$ is a random vector and ψ is some function defining the estimator. In particular, the estimator $\hat{\mathbf{m}}$ is an M-estimator. To see this, define $\theta = (\mathbf{w}, \lambda)$, where λ is the Lagrangian multiplier associated with the constraint. Using the Kuhn-Tucker conditions, the estimator $\hat{\mathbf{m}}$ can then be formulated as the solution of equation (17) where $\psi = \psi_J$ is defined as follows (for sphered data):

$$\psi_J(\mathbf{x}, \theta) = \begin{pmatrix} \mathbf{x} g(\mathbf{w}^T\mathbf{x}) + k\lambda\mathbf{w} \\ \|\mathbf{w}\|^2 - 1 \end{pmatrix} \tag{18}$$

where k is an irrelevant constant.

The analysis of robustness of an M-estimator is based on the concept of an influence function, $\mathrm{IF}(\mathbf{x}, \hat{\theta})$. (For simplicity, we only consider here the influence function at the model distribution [12]). Intuitively speaking, the influence function measures the influence of single observations on the estimator. It would be desirable to have an influence function that is bounded as a function of $\mathbf{x}$, as this implies that even the influence of a far-away outlier is 'bounded', and cannot change the estimate too much. This requirement leads to one definition of robustness, which is called B-robustness. An estimator is called B-robust, if its influence function is bounded as a function of $\mathbf{x}$, i.e., $\sup_{\mathbf{x}} \| \mathrm{IF}(\mathbf{x}, \hat{\theta}) \|$ is finite for every $\hat{\theta}$. Even if the influence function is not bounded, it should grow as slowly as possible when $\|\mathbf{x}\|$ grows, to reduce the distorting effect of outliers.

It can be shown [12] that the influence function of an M-estimator equals

$$\mathrm{IF}(\mathbf{x}, \hat{\theta}) = \mathbf{B}\psi(\mathbf{x}, \hat{\theta}) \tag{19}$$

where $\mathbf{B}$ is an irrelevant invertible matrix that does not depend on $\mathbf{x}$. On the other hand, using our definition of ψ_J, and denoting by $\gamma = \mathbf{w}^T\mathbf{x}/\|\mathbf{x}\|$ the cosine of the angle between $\mathbf{x}$ and $\mathbf{w}$, one obtains easily

$$\|\psi(\mathbf{x}, (\mathbf{w}, \lambda))\|^2 = k_1 \frac{1}{\gamma^2} h^2(\mathbf{w}^T\mathbf{x}) + k_2 h(\mathbf{w}^T\mathbf{x}) + k_3 \tag{20}$$

where k_1, k_2, k_3 are constants that do not depend on $\mathbf{x}$, and $h(u) = ug(u)$. Thus we see that the robustness of $\hat{\mathbf{m}}$ essentially depends on the behavior of the function $h(u)$. The slower $h(u)$ grows, the more robust the estimator. However, the estimator cannot be really B-robust, because the γ in the denominator prevents the influence function from being bounded for all $\mathbf{x}$. In particular, outliers that are almost orthogonal to $\hat{\mathbf{m}}$, and have large norms, may still have a large influence on the estimator. These results are stated in the following theorem:

Theorem 4 *Assume that the data $\mathbf{x}$ is whitened (sphered) in a robust manner. Then the influence function of the estimator $\hat{\mathbf{m}}$ 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 $\{\mathbf{x} \mid \hat{\mathbf{m}}^T\mathbf{x}/\|\mathbf{x}\| > \epsilon\}$ 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 $\hat{\mathbf{m}}$ is rather robust against outliers. If this is not possible, one should at least choose a contrast function $G(u)$ that does not grow very fast when $|u|$ grows. If, in contrast, $G(u)$ grows very fast when $|u|$ grows, the estimates depend mostly on a few observations far from the origin. This leads to highly non-robust estimators, which can be completely misled by just a couple of bad outliers. This is the case, for example, when kurtosis is used as a contrast function, which is equivalent to using $\hat{\mathbf{m}}$ with $G(u) = u^4$.

Finally, let us note that the analysis made above is not restricted to the ICA problem, but is directly applicable to many other cases of restricted non-linear Hebbian learning as well. An important example is robust PCA algorithms [30].

4.5 Performance in the exponential power family

It is useful to analyze the implications of the theoretical results of the preceding sections by considering the following exponential power family of density functions:

$$f_\alpha(x) = k_1 \exp(k_2 |x|^\alpha) \tag{21}$$

where α is a positive constant, 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 $.5 < \alpha < 2$, 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 3, 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:

$$G_{opt}(u) = |u|^\alpha \tag{22}$$

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 4.4 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, 24], one reaches the conclusion that as a general-purpose contrast function, one should choose a function G that resembles rather

$$G_{opt}(u) = |u|^\alpha, \text{where } \alpha < 2. \tag{23}$$

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 $a_1 > 1$ is a moderately large 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 4, thus providing an estimator that is as robust as possible in the framework of estimators of type (13). As regards the constants, we have found experimentally $a_1 = 2$ and $a_2 = 1$ to provide good approximations. Note that there is a trade-off between the precision of the approximation and the smoothness of the resulting objective function.

5 Choosing the Contrast Function in Practice

Now we shall treat the question of choosing the contrast function G used in (13). First of all, the statistical analysis of Section 4 gives some criteria for the choice of the contrast function. Moreover, in [18], the problem of choosing the function G so that the approximation of negentropy becomes as exact as possible, was treated, and the recommendations were practically identical to the conclusions of Section 4.

However, in the practical choice of the contrast function, there are also other criteria that are important. In particular, the following two must be mentioned:

1. *Computational simplicity.* The contrast function should be fast to compute.
2. *Ordering of distributions.* This means that 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. This means that when the

independent components are estimated one-by-one, the order in which they appear is influenced by the choice of the contrast function. 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.

Regarding the first point, polynomial functions tend to be faster to compute than, say, the hyperbolic tangent. However, non-polynomial contrast functions may be replaced by piecewise polynomial approximations without losing the benefits of non-polynomial functions. For example, in the case of $g(u) = \tanh(a_2 u)$, one may define a piecewise linear approximation g so that $g(u) = a_3 u$ for $|u| < 1/a_3$ and $g(u) = \text{sign}(u)$ otherwise, where $a_3 \geq 1$ is a constant. (This amounts to using the so-called Huber function [12] as G).

Regarding the second point, we have empirically observed the following. Using kurtosis, one tends to find first the super-Gaussian components, and using the other functions given in this paper, one finds first the sub-Gaussian ones.

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

$$G_1(u) = \frac{1}{a_1} \log \cosh(a_1 u), \qquad g_1(u) = \tanh(a_1 u) \tag{24}$$

$$G_2(u) = -\frac{1}{a_2} \exp(-a_2 u^2/2), \quad g_2(u) = u \exp(-a_2 u^2/2) \tag{25}$$

$$G_3(u) = \frac{1}{4} u^4, \qquad g_3(u) = u^3 \tag{26}$$

where $a_1 \geq 1, a_2 \approx 1$ are constants, and piecewise linear approximations of (24) and (25) 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.
- in the special case where it is important to *first* find the super-Gaussian components, kurtosis can be used.

A different case that is worth mentioning is when the data contains *asymmetric* independent components. The contrast functions introduced above work perfectly well even if the independent components are not symmetric. However, if one wants to estimate especially the asymmetric independent components, this can be accomplished by using an odd contrast function, for example, $G(u) = u^3$. Then the method finds only asymmetric independent components.

Finally, note that 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.

6 Algorithms

6.1 Introduction

In the preceding sections, we introduced new contrast (or objective) functions for ICA based on minimization of mutual information, 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 function in (13). In this section, we introduce two methods of maximization suited for this task. First, adaptive neural learning rules are given in Section 6.3, and second, a fast batch algorithm, called the fixed-point algorithm, is introduced in Section 6.4. Before introducing the learning rules, however, we discuss the problem of preprocessing the data.

6.2 Preprocessing

First of all, note that it has been assumed throughout the paper that the data is zero-mean. Therefore, the very first step of preprocessing must consists of centering the data, i.e., subtracting the mean $E\{\mathbf{x}\}$ from the observed vectors $\mathbf{x}$.

Our basic ICA algorithms also 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 observed variable $\mathbf{x}$ is linearly transformed to a variable $\mathbf{v} = \mathbf{Q}\mathbf{x}$ such that the correlation matrix of $\mathbf{v}$ equals unity: $E\{\mathbf{v}\mathbf{v}^T\} = \mathbf{I}$. This transformation is always possible. Indeed, it can be accomplished by classical PCA [8, 11]. In addition to sphering, PCA may allow us to determine the number of independent components in the ICA data model (2): if 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 data model (2). Thus this reduction of dimension justifies the assumption $m = n$ in the model. Even if the data model is not assumed, it may be a good idea to reduce the dimension of the data, if the covariance matrix is near-singular. Conveniently, PCA allows both the reduction of dimension and sphering at the same time [8, 11].

The usefulness of sphering resides in the fact that after sphering, the constraints of decorrelation and unit variance in (13) can be expressed as the simpler constraints of orthogonality and unit norm. In other words,

$$E\{(\mathbf{w}_k^T\mathbf{v})(\mathbf{w}_j^T\mathbf{v})\} = \mathbf{w}_k^T\mathbf{w}_j. \tag{27}$$

The usefulness of the preliminary sphering transform is thus mainly computational. It does not essentially affect the final transforms s_i. Of course, the matrix $\mathbf{M}$ is changed since one now has $\mathbf{s} = \mathbf{M}\mathbf{v} = \mathbf{M}\mathbf{Q}\mathbf{x}$ and thus the complete transform equals $\mathbf{M}\mathbf{Q}$ instead of $\mathbf{M}$.

Similarly, for the ICA data model, one obtains after sphering $\mathbf{v} = \mathbf{B}\mathbf{s}$, where $\mathbf{B} = \mathbf{Q}\mathbf{A}$ is an *orthogonal* matrix, because $E\{\mathbf{v}\mathbf{v}^T\} = \mathbf{B}E\{\mathbf{s}\mathbf{s}^T\}\mathbf{B}^T = \mathbf{B}\mathbf{B}^T = \mathbf{I}$. (Here we define the independent components s_i to have unit variance, see [8].)

In the following, we shall thus assume that the data is sphered. For simplicity, the sphered data will be denoted by $\mathbf{x}$ and the transformation matrix by $\mathbf{M}$ (and the mixing matrix by $\mathbf{A}$), thus retaining the notation used in the preceding sections.

6.3 Adaptive Neural Algorithms

6.3.1 Basic One-unit Learning Rule

Taking the instantaneous gradient of the approximation of negentropy in (12) with respect to $\mathbf{w}$, and taking the normalization $E\{(\mathbf{w}^T\mathbf{x})^2\} = 1$ into account, one obtains the following Hebbian-like learning rule

$$\Delta\mathbf{w} \propto r\mathbf{x}g(\mathbf{w}^T\mathbf{x}), \text{ normalize } \mathbf{w} \tag{28}$$

where $r = E\{G(\mathbf{w}^T\mathbf{x})\} - E\{G(\nu)\}$, ν being a standardized Gaussian random variable. The normalization means that $\mathbf{w}$ is projected on the constraint set defined by $E\{(\mathbf{w}^T\mathbf{x})^2\} = \|\mathbf{w}\|^2 = 1$. For example, $\mathbf{w}$ may be divided by its norm. The constant r, which gives the learning rule a kind of 'self-adaptation' quality, can be easily estimated on-line as follows:

$$\Delta r \propto (G(\mathbf{w}^T\mathbf{x}) - E\{G(\nu)\}) - r \tag{29}$$

A neuron learning according to the learning rule (28) thus finds a direction of maximum negentropy, which may be interpreted as a projection pursuit direction. Under the assumption of the ICA data model, such a direction is also the direction of one of the independent components s_i.

The learning rule in (28) can be further simplified. First note that the constant r does not change the stationary points of the learning rule. Its sign does affect their stability, though. Therefore, one can replace the r by a its sign without essentially affecting the behavior of the learning rule. This is useful, for example, in cases where we have some a priori information on the distributions of the independent components. For example, speech signals are usually highly super-Gaussian. One might thus evaluate roughly $E\{G(s_i) - G(\nu)\}$ for some super-Gaussian independent components and then take this, or its sign, as the value of r. For example, if g is the tanh function, then $r = -1$ works for typical super-Gaussian independent components.

6.3.2 A Network of Several Neurons

To find the whole n-dimensional transform $\mathbf{s} = \mathbf{M}\mathbf{x}$, one can then use a network of n neurons, each of which learns according to eq. (28). Of course, some kind of feedback is then necessary to take into account the decorrelation constraints in (13), which prevent the weight vectors from converging to the same points. Because for sphered data the constraints can be expressed as simple orthogonality constraints, classical orthogonalizing feedbacks as in stochastic gradient ascent [29], Sanger's algorithm [35], or the bigradient rule [25] can be used. A more detailed discussion of such feedbacks can be found in, e.g., [19, 25]. For example, the symmetric bigradient feedback, which also contains the normalization, would yield the following universal learning rule[1] for the weight matrix $\mathbf{W} = (\mathbf{w}_1, ..., \mathbf{w}_n)$ whose columns are the weight vectors $\mathbf{w}_i$ of the neurons:

$$\begin{aligned}\mathbf{W}(t+1) = \mathbf{W}(t) &+ \mu(t)\mathbf{x}(t)g(\mathbf{x}(t)^T\mathbf{W}(t))\text{diag}(r_i(t)) \\ &+ \frac{1}{2}\mathbf{W}(t)(\mathbf{I} - \mathbf{W}(t)^T\mathbf{W}(t))\end{aligned} \tag{30}$$

where $\mu(t)$ is the learning rate sequence, and the function $g(.) = G'(.)$ is applied separately on every component of the row vector $\mathbf{x}(t)^T\mathbf{W}(t)$. In this most general version of the

[1] This learning rule is rather similar, but not identical, to the learning rule derived in [21] from a different approach. The difference is in the self-adaptation constant.

learning rule, the $r_i, i = 1...n$ are estimated separately for each neuron according to (29). They may also be fixed using prior knowledge, as explained above. Of course, the learning function g could also be different for each neuron. This is, however, not necessary: in contrast to many other neural learning rules, our learning rule can estimate independent components of practically any distribution with a single learning function. This is due to the 'self-adaptation' provided by the multiplicative constants r_i. After convergence, the complete transform is then given by $\mathbf{s} = \mathbf{W}^T\mathbf{x}$.

6.4 Fixed-point Algorithms

The advantage of neural on-line learning rules like those introduced above 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, some ways to make the learning faster and more reliable may be needed. The fixed-point iteration algorithms are such an alternative. Based on the learning rules introduced above, we introduce here a fixed-point algorithm. In this algorithm, 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 algorithm may be considered neural. In particular, it is parallel, distributed, computationally simple, and requires little memory space.

6.4.1 Fixed-point algorithm for one unit

To begin with, we shall derive the fixed-point algorithm for one unit. 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 [26], 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

$$E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w} = 0 \tag{31}$$

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 (31) by F, we obtain its Jacobian matrix $JF(\mathbf{w})$ as

$$JF(\mathbf{w}) = E\{\mathbf{x}\mathbf{x}^Tg'(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{I} \tag{32}$$

To simplify the inversion of this matrix, we decide to approximate the first term in (32). Since the data is sphered, a reasonable approximation seems to be $E\{\mathbf{x}\mathbf{x}^Tg'(\mathbf{w}^T\mathbf{x})\} = E\{\mathbf{x}\mathbf{x}^T\}E\{g'(\mathbf{w}^T\mathbf{x})\} = E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{I}$. 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:

$$\begin{aligned}\mathbf{w}^+ &= \mathbf{w} - [E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta] \\ \mathbf{w}^* &= \mathbf{w}^+/\|\mathbf{w}^+\|\end{aligned} \tag{33}$$

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 (33) by $\beta - E\{g'(\mathbf{w}^T\mathbf{x})\}$. This gives the following *fixed-point algorithm*

$$\begin{aligned}\mathbf{w}^+ &= E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{w} \\ \mathbf{w}^* &= \mathbf{w}^+/\|\mathbf{w}^+\|\end{aligned} \tag{34}$$

which was introduced in [17] using a more heuristic derivation. An earlier version (using kurtosis) was derived as a fixed-point iteration of a neural learning rule in [20], 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 (31). In the Appendix it is also 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 (33), obtaining the *stabilized fixed-point algorithm*

$$\begin{aligned} \mathbf{w}^+ &= \mathbf{w} - \mu[E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta] \\ \mathbf{w}^* &= \mathbf{w}^+/\|\mathbf{w}^+\| \end{aligned} \tag{35}$$

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 (35) 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 (34). 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 (34) works for non-sphered data:

$$\begin{aligned} \mathbf{w}^+ &= \mathbf{C}^{-1}E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{w} \\ \mathbf{w}^* &= \mathbf{w}^+/\sqrt{(\mathbf{w}^+)^T\mathbf{C}\mathbf{w}^+} \end{aligned} \tag{36}$$

where $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$ is the covariance matrix of the data. The stabilized version, algorithm (35), can also be modified as follows to work with non-sphered data:

$$\begin{aligned} \mathbf{w}^+ &= \mathbf{w} - \mu[\mathbf{C}^{-1}E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta] \\ \mathbf{w}^* &= \mathbf{w}^+/\sqrt{(\mathbf{w}^+)^T\mathbf{C}\mathbf{w}^+} \end{aligned} \tag{37}$$

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, as explained in subsection 6.2.

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 often not a good idea because the computations may become too demanding. Then the averages can estimated using a smaller sample, whose size may have a considerable effect on the accuracy of the final estimates. We have found a sample of 1000 observations of $\mathbf{x}$ to be reasonable in most circumstances. Note that 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 [21, 25].

6.4.2 Fixed-point algorithm for several units

As was the case with the neural learning rules, the one-unit algorithm of the preceding subsection can be used to construct a system of n neurons to estimate the whole ICA transformation $\mathbf{s} = \mathbf{Mx}$. 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 $\mathbf{w}_{p+1}$:

$$\begin{aligned} &1.\ \text{Let } \mathbf{w}_{p+1} = \mathbf{w}_{p+1} - \textstyle\sum_{j=1}^{p} \mathbf{w}_{p+1}^T\mathbf{C}\mathbf{w}_j\mathbf{w}_j \\ &2.\ \text{Let } \mathbf{w}_{p+1} = \mathbf{w}_{p+1}/\sqrt{\mathbf{w}_{p+1}^T\mathbf{C}\mathbf{w}_{p+1}} \end{aligned} \tag{38}$$

In certain applications, however, it may be desired to use a symmetric decorrelation, in which no vectors are 'privileged' over others [25]. This can be accomplished, e.g., by the classical method involving matrix square roots,

$$\text{Let } \mathbf{W} = \mathbf{W}(\mathbf{W}^T\mathbf{C}\mathbf{W})^{-1/2} \tag{39}$$

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

$$\begin{aligned} &1.\ \text{Let } \mathbf{W} = \mathbf{W}/\sqrt{\|\mathbf{W}^T\mathbf{C}\mathbf{W}\|} \\ &\text{Repeat 2. until convergence:} \\ &2.\ \text{Let } \mathbf{W} = \tfrac{3}{2}\mathbf{W} - \tfrac{1}{2}\mathbf{W}\mathbf{W}^T\mathbf{C}\mathbf{W} \end{aligned} \tag{40}$$

The norm in step 1 can be almost any ordinary matrix norm, e.g., the largest absolute row (or column) sum (but not the Frobenius norm). The convergence of the orthonormalization method in (40), which may be considered a variation of Potter's formula [34], is proven in the Appendix.

6.4.3 Properties of the Fixed-Point Algorithm

The fixed-point algorithm for (approximate) minimization of mutual information has a number of desirable properties.

- 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 7).

- 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, which is in contrast to many algorithms, where some estimate of the probability distribution function has to be first available.

- 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.

7 Simulation results

7.1 Blind source separation with outliers

The first simulation consisted of blind source separation in the presence of outliers. The purpose was to demonstrate the fact that the use of the contrast functions G_1 and G_2 in (24) and (25) makes the estimators more robust against outliers that the conventional estimators based on kurtosis, or G_3 in (26). We applied our algorithm to blind separation of four source signals that have visually appealing waveforms to illustrate this application for those not familiar with it. The first 100 values of the source signals, whose total length was 2000 points, are depicted in Fig. 1. The two signals are on the left are sub-Gaussian, and two on the right are super-Gaussian. These source signals were mixed using several random matrices, whose elements were drawn from a standardized Gaussian distribution, so as to obtain different mixed signals. To test the robustness of our algorithms, *four outliers* whose values were ± 10 were added in random locations. Then we used the methods introduced in this paper to estimate the original signals. Three different contrast functions were used. These were the G_i given in eq. (24–26), that is, cubic, log cosh, and the Gaussian function. 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.

The fixed-point algorithm was used for maximizing the contrast function. In all the runs, the following were observed:

- Estimates based on kurtosis (or the cubic non-linearity) were essentially worse than the others

- Estimates using G_2 in (25) were slightly better than those using G_1 in (24).

These points are clearly demonstrated in the results of a typical run depicted in Figs 2 to 5. Fig. 2 shows the set of mixtures that was used in the following simulations. Figs 3 to 5 show

the estimates of the source signals for the different non-linearities. Clearly, the estimates based of the functions in (24) and (25) are better than the estimate based on kurtosis, or (26).

7.2 Asymptotic variance

To investigate the asymptotic variance, i.e., efficiency, of the estimates obtained using generalized contrast functions, we performed simulations in which 3 different contrast functions were used to estimate one independent component from a mixture of 4 identically distributed independent components. The contrast functions used were the same three functions $G_i, i = 1, 2, 3$ in Eq. (24–26) as in the preceding simulation. 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 sample size was fixed at 1000 and the fixed-point algorithm was used to maximize the contrast function. The asymptotic mean absolute deviations 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. Mean absolute deviation was used instead of variance because it is a more robust measure of deviation.

The results in the basic, noiseless case are depicted in Fig. 6. 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. 7. This time, kurtosis did not perform better even in the case of the sub-Gaussian density. This result goes against the view that kurtosis would tolerate Gaussian noise well. Indeed, the theoretical arguments supporting that view neglect any finite-sample effects, and may thus have rather limited validity.

7.3 Speed of convergence

We also studied the convergence properties of the algorithms in more detail, in a third set of simulations. Four independent components of different distributions (uniform, binary, Laplace, and cube of a Gaussian variable) were used.

First, the neural learning rule in (30) was used. The learning rate was set as .05 for the first 2000 iterations, after which it was diminished to .01 to study the effect of the learning rate. The total number of data points used was 3000. The simulations were made for 10 different initial values of $\mathbf{W}$, and the results were averaged over trials. The results reported used the contrast function G_1, or log cosh. The results for the other contrast functions are essentially similar. We defined a simple error measure based on the matrix $\mathbf{W}^{-1}\mathbf{A}$ where $\mathbf{A}$ is the mixing matrix. (The error measure used for investigating the asymptotic variance is not suitable here, because it is not well defined far from convergence). The matrix $\mathbf{W}^{-1}\mathbf{A}$ should converge to a permutation matrix, up to multiplicative signs. (It does not converge, in general, to an identity matrix because the order and the multiplicative signs of the independent components cannot be determined.) Thus the sum of the squares of the elements of this matrix was computed, excluding the 4 largest elements.

Fig. 8 shows the values of the convergence index during learning. One sees that some conver-

gence was achieved at approximately 1000 iterations, but there were still large fluctuations in the convergence index. Decreasing the learning rate reduced greatly the fluctuations in the estimates. This illustrates the fact that annealing the learning rate makes the learning more accurate. A larger learning rate in the beginning, on the other hand, enabled faster learning.

Next we performed the same simulations using the fixed-point algorithm. The data consisted of 1000 points, and the whole data was used at every iteration, that is, the sample size was 1000. The symmetric version of the fixed-point algorithm for sphered data was used. The results are shown in Fig. 9. On the average, only three iterations were necessary to achieve the maximum accuracy allowed by the data. This illustrates also that the fixed-point algorithm is easier to use, since no annealing of the learning rate is necessary; in fact, no parameters at all needs to be determined in the basic case.

7.4 Real-life applications

Experiments on two kinds of real life data have also been performed using the contrast functions and algorithms introduced above. These applications are artifact cancellation in EEG by means of blind source separation [24, 36], and feature extraction of image data [16, 24]. These experiments further validate the ICA methods introduced in this paper.

8 Conclusions

The problem of linear independent component analysis (ICA), which is a form of redundancy reduction, was addressed. Following Comon [8], 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 maximal and which can also be considered projection pursuit directions [15]. The novel approximations of negentropy introduced in [18] were then used for constructing novel contrast (objective) functions for ICA. This resulted in a generalization of the kurtosis-based approach in [8, 9], and also enabled estimation of the independendent components one by one. The statistical properties of these contrast functions were analyzed in detail in the framework of the linear mixture model, and it was shown that for a suitable choices of the contrast functions, the statistical properties were superior to those of the kurtosis-based approach. Next, two classes of new algorithms for optimizing the contrast functions were introduced. The first class consisted of neural on-line learning rules using simple (self-adaptive) Hebbian-like learning. The second class consisted of fixed-point learning rules that are not neural in the sense that they are non-adaptive, but share the other benefits of neural learning rules. The convergence of the fixed-point learning rules was shown to be very fast (cubic or at least quadratic). Combining the good statistical properties of the new contrast functions, and the good algorithmic 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.

References

[1] S. Amari, A. Cichocki, and H.H. Yang. A new learning algorithm for blind source separation. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances*

in Neural Information Processing 8 (Proc. NIPS'95), 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*, 1997. To appear.

[5] J.-F. Cardoso. Iterative techniques for blind source separation using only fourth-order cumulants. In *Proc. EUSIPCO*, pages 739–742, Brussels, Belgium, 1992.

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

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

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

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

[10] 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.

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

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

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

[14] P.J. Huber. *Robust Statistics*. Wiley, 1981.

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

[16] J. Hurri, A. Hyvärinen, J. Karhunen, and E. Oja. Wavelets and natural image statistics. In *Proc. Scandinavian Conf. on Image Analysis '97*, Lappenranta, Finland, 1997.

[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. New approximations of differential entropy for independent component analysis and projection pursuit. Technical Report A47, Helsinki University of Technology, Laboratory of Computer and Information Science, 1997.

[19] A. Hyvärinen and E. Oja. Simple neuron models for independent component analysis. *Int. Journal of Neural Systems*, 7(6):671–687, 1996.

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

[21] A. Hyvärinen and E. Oja. Independent component analysis by general non-linear Hebbian-like learning rules. *Signal Processing*, 1997. To appear.

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

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

[24] J. Karhunen, A. Hyvärinen, R. Vigario, J. Hurri, and E. Oja. Applications of neural blind separation to signal and image processing. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'97)*, pages 131–134, Munich, Germany, 1997.

[25] 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.

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

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

[28] E. Oja. The nonlinear PCA learning rule in independent component analysis. *Neurocomputing*, 6, 1997. To appear.

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

[30] E. Oja and L.-Y. Wang. Robust fitting by nonlinear neural units. *Neural Networks*, 9:435–444, 1996.

[31] 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.

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

[33] 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.

[34] J. E. Potter. New statistical formulas. Technical report, Instrumentation Laboratory, MIT, 1963.

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

[36] R. Vigário. Extraction of ocular artifacts from eeg using independent component analysis. *Electroenceph. clin. Neurophysiol.*, 1997. To appear.

A Appendix: Proofs

A.1 Proof of Theorem 1

Make the change of coordinates $\mathbf{z} = \mathbf{A}^T\mathbf{w}$. Denote by $F(\mathbf{z})$ the function $F(\mathbf{z}) = E\{G(\mathbf{z}^T\mathbf{s})\} = E\{G(\mathbf{w}^T\mathbf{x})\}$. Then we can calculate the gradient $\nabla F(\mathbf{z}) = E\{\mathbf{s}g(\mathbf{z}^T\mathbf{s})\}$ and the Hessian as $\nabla^2 F(\mathbf{z}) = E\{\mathbf{s}\mathbf{s}^T g'(\mathbf{z}^T\mathbf{s})\}$. Without loss of generality, it is enough to analyze the stability of the point $\mathbf{z} = \mathbf{e}_1$, where $\mathbf{e}_1 = (1, 0, 0, 0, ...)$, which corresponds to $\mathbf{w}$ being one of the rows of $\mathbf{A}^{-1}$. (Because F is even, nothing changes for $\mathbf{z} = -\mathbf{e}_1$.) Evaluating the gradient and the Hessian at point $\mathbf{z} = \mathbf{e}_1$, we get

$$\nabla F(\mathbf{e}_1) = \mathbf{e}_1 E\{s_1 g(s_1)\} \tag{41}$$

and

$$\nabla^2 F(\mathbf{e}_1) = \mathrm{diag}(E\{s_1^2 g'(s_1)\}, E\{g'(s_1)\}, E\{g'(s_1)\}, ...) \tag{42}$$

Making a small perturbation $\epsilon = (\epsilon_1, \epsilon_2, ...)$, we obtain

$$F(\mathbf{e}_1 + \epsilon) = F(\mathbf{e}_1) + \epsilon^T \nabla F(\mathbf{e}_1) \tag{43}$$

$$+\frac{1}{2}\epsilon^T \nabla^2 F(\mathbf{e}_1)\epsilon + o(\|\epsilon\|^2) \tag{44}$$

$$= F(\mathbf{e}_1) + E\{s_1 g(s_1)\}\epsilon_1 + \frac{1}{2}[E\{s_1^2 g'(s_1)\}\epsilon_1^2 \tag{45}$$

$$+E\{g'(s_1)\}\sum_{i>1}\epsilon_i^2] + o(\|\epsilon\|^2) \tag{46}$$

Due to the constraint $\|\mathbf{z}\| = 1$ we get $\epsilon_1 = \sqrt{1 - \epsilon_2^2 - \epsilon_3^2 - ...} - 1$. Due to the fact that $\sqrt{1-\gamma} = 1 - \gamma/2 + o(\gamma)$, the term of order ϵ_1^2 in (46) is $o(\|\epsilon\|^2)$, i.e., of higher order, and can be neglected. Using the aforementioned first-order approximation for ϵ_1 we obtain $\epsilon_1 = -\sum_{i>1}\epsilon_i^2/2 + o(\|\epsilon\|^2)$, which finally gives

$$F(\mathbf{e}_1 + \epsilon) = F(\mathbf{e}_1) + [E\{g'(s_1) - s_1 g(s_1)\}]\frac{1}{2}\sum_{i>1}\epsilon_i^2 + o(\|\epsilon\|^2) \tag{47}$$

which clearly proves $\mathbf{z} = \mathbf{e}_1$ is an extremum of F, and its type is determined by the sign of $E\{g'(s_1) - s_1 g(s_1)\}$. Now, if the condition in Theorem 1 is true, then either the point is a maximum of F, and $E\{G(s_1)\} - E\{G(\nu)\}$ is positive, or the point is a minimum of F, and $E\{G(s_1)\} - E\{G(\nu)\}$ is negative. In either case, it is a maximum of J_G, Q.E.D.

A.2 Proof of Theorem 2

Making the change of variable $\mathbf{z} = \mathbf{A}^T\mathbf{w}$, the equation defining the solution $\hat{\mathbf{z}}$ for $\mathbf{z}$ becomes

$$\sum_t \mathbf{s}(t)g(\hat{\mathbf{z}}^T\mathbf{s}(t)) = \lambda\sum_t \mathbf{s}(t)\mathbf{s}(t)^T\hat{\mathbf{z}} \tag{48}$$

where $t = 1, .., T$ is the sample index, T is the sample size, $\mathbf{s}(t)$ are the realizations of $\mathbf{s}$ in the sample, and λ is a Lagrangian multiplier. Without loss of generality, let us assume that $\hat{\mathbf{z}}$ is near the ideal solution $\mathbf{z} = (1, 0, 0, ...)$. Note that due to the constraint $E\{(\mathbf{w}^T\mathbf{x})^2\} = \|\mathbf{z}\|^2 = 1$, the variance of the first component of $\hat{\mathbf{z}}$, denoted by $\hat{z}_1$, is of a smaller order than the variance of the vector of other components, denoted by $\hat{\mathbf{z}}_{-1}$ (see proof of Theorem 1). Excluding the first component in (48), and making the first-order approximation $g(\hat{\mathbf{z}}^T\mathbf{s}) =$

$g(s_1) + g'(s_1)\hat{\mathbf{z}}_{-1}^T \mathbf{s}_{-1} + o(\|\hat{\mathbf{z}}_{-1}\|)$, where also $\mathbf{s}_{-1}$ denotes $\mathbf{s}$ without its first component, one obtains after some simple manipulations

$$\frac{1}{\sqrt{T}} \sum_t \mathbf{s}_{-1}[g(s_1) - \lambda s_1] = \frac{1}{T} \sum_t \mathbf{s}_{-1}[-\mathbf{s}_{-1}^T g'(s_1) + \lambda \mathbf{s}_{-1}^T]\hat{\mathbf{z}}_{-1}\sqrt{T} + o(\|\hat{\mathbf{z}}_{-1}\|)/\sqrt{T} \quad (49)$$

where the sample index t has been dropped for simplicity. Making the first-order approximation $\lambda = E\{s_1 g(s_1)\}$, one can write (49) in the form $u = v\hat{\mathbf{z}}_{-1}\sqrt{T} + o(\|\hat{\mathbf{z}}_{-1}\|)/\sqrt{T}$ where v converges to the identity matrix multiplied by $E\{s_1 g(s_1)\} - E\{g'(s_1)\}$, and u converges to a random variable that has a normal distribution of zero mean whose covariance matrix equals the identity matrix multiplied by $E\{g^2(s_1)\} - (E\{s_1 g(s_1)\})^2$. This implies the theorem, since $\hat{\mathbf{z}}_{-1} = \mathbf{B}\hat{\mathbf{w}}$, where $\mathbf{B}$ is the inverse of $\mathbf{A}^T$ without its first row.

A.3 Proof of Theorem 3

Using basic variational calculus [26], and the equality $\int f(u)g'(u) = \int f'(u)g(u)$, one obtains at the optimum of V_G

$$f_i(u)g(u) - c_1 f_i(u)u = c_2 f_i(u)u + c_3 f_i'(u) \quad (50)$$

which gives

$$g(u) = (c_1 + c_2)u + c_3 f_i'(u)/f_i(u) \quad (51)$$

which immediately implies Theorem 3, since $G'(u) = g(u)$.

A.4 Proof of convergence of algorithm (34)

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:

$$E\{s_i g(s_i) - g'(s_i)\} \neq 0, \text{for any } i \quad (52)$$

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 (52) 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, 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 (34) using a Taylor approximation for g and g', first obtaining

$$g(\mathbf{z}^T\mathbf{s}) = g(z_1 s_1) + g'(z_1 s_1)\mathbf{z}_{-1}^T\mathbf{s}_{-1} + \frac{1}{2}g''(z_1 s_1)(\mathbf{z}_{-1}^T\mathbf{s}_{-1})^2 \quad (53)$$

$$+\frac{1}{6}g'''(z_1 s_1)(\mathbf{z}_{-1}^T\mathbf{s}_{-1})^3 + O(\|\mathbf{z}_{-1}\|^4) \quad (54)$$

and then

$$g'(\mathbf{z}^T\mathbf{s}) = g'(z_1 s_1) + g''(z_1 s_1)\mathbf{z}_{-1}^T\mathbf{s}_{-1} \quad (55)$$

$$+\frac{1}{2}g'''(z_1 s_1)(\mathbf{z}_{-1}^T\mathbf{s}_{-1})^2 + O(\|\mathbf{z}_{-1}\|^3) \quad (56)$$

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,

$$z_1^+ = E\{s_1 g(z_1 s_1) - g'(z_1 s_1)\} + O(\|z_{-1}\|^2) \tag{57}$$

$$z_i^+ = \frac{1}{2}\,\text{skew}\,(s_i) E\{g''(s_1)\} z_i^2$$
$$+\frac{1}{6}\,\text{kurt}\,(s_i) E\{g'''(s_1)\} z_i^3 + O(\|z_{-1}(\|^4), \text{for } i > 1 \tag{58}$$

We obtain also

$$\mathbf{z}^* = \mathbf{z}^+ / \|\mathbf{z}^+\| \tag{59}$$

This shows clearly that under the assumption (52), 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, (58) 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.5 Proof of convergence of (40)

Denote by $\mathbf{W}_+$ the result of applying once the iteration step 2 in (40) on $\mathbf{W}$. Let $\mathbf{W}^T\mathbf{C}\mathbf{W} = \mathbf{E}\mathbf{D}\mathbf{E}^T$ be the eigenvalue decomposition of $\mathbf{W}^T\mathbf{C}\mathbf{W}$. Then we have

$$\mathbf{W}_+^T\mathbf{C}\mathbf{W}_+ = \frac{9}{4}\mathbf{E}\mathbf{D}\mathbf{E}^T + \frac{3}{2}\mathbf{E}\mathbf{D}^2\mathbf{E}^T + \frac{1}{4}\mathbf{E}\mathbf{D}^3\mathbf{E}^T \tag{60}$$

$$= \mathbf{E}(\frac{9}{4}\mathbf{D} - \frac{3}{2}\mathbf{D}^2 + \frac{1}{4}\mathbf{D}^3)\mathbf{E}^T \tag{61}$$

Note that due to the normalization, i.e. division of $\mathbf{W}$ by $\sqrt{\|\mathbf{W}^T\mathbf{C}\mathbf{W}\|}$, all the eigenvalues of $\mathbf{W}^T\mathbf{C}\mathbf{W}$ are in the interval $[0, 1]$. Now, according to (60), for every eigenvalue of $\mathbf{W}^T\mathbf{C}\mathbf{W}$, say λ_i, $\mathbf{W}_+^T\mathbf{C}\mathbf{W}_+$ has a corresponding eigenvalue $h(\lambda_i)$ where $h(.)$ is defined as:

$$h(\lambda) = \frac{9}{4}\lambda - \frac{3}{2}\lambda^2 + \frac{1}{4}\lambda^3 \tag{62}$$

This function is plotted in Fig. 10 in the interval $[0, 1]$. Thus, after k iterations, the eigenvalues of $\mathbf{W}^T\mathbf{C}\mathbf{W}$ are obtained as $h(h(h(...h(\lambda_i))))$, where h is applied k times on the λ_i, which are the eigenvalues of $\mathbf{W}^T\mathbf{C}\mathbf{W}$ for the original matrix before the iterations. It is therefore clear that all the eigenvalues of $\mathbf{W}^T\mathbf{C}\mathbf{W}$ converge to 1, which means that $\mathbf{W}^T\mathbf{C}\mathbf{W} \to \mathbf{I}$, Q.E.D. Moreover, it is not difficult to see that the convergence is quadratic.

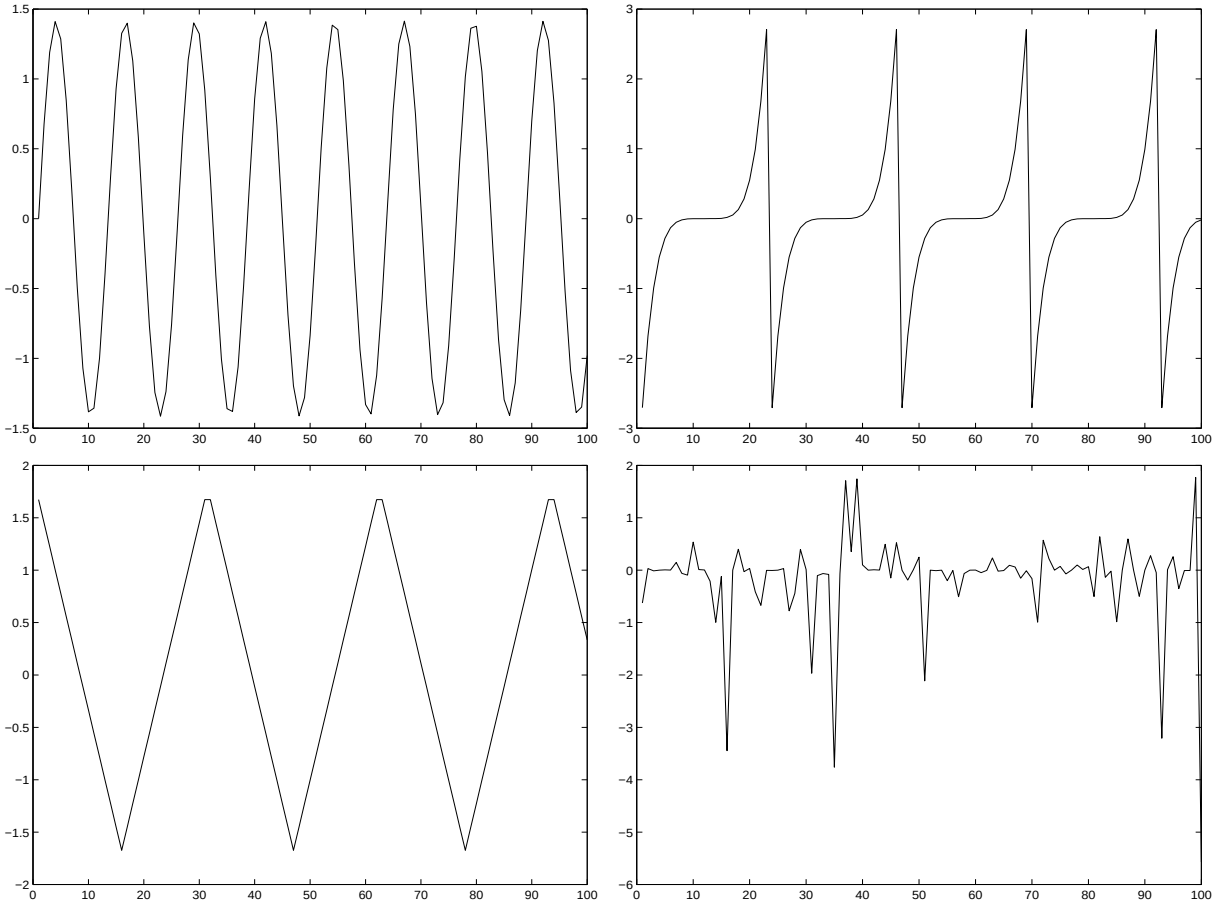

Figure 1: The source signals used in the simulations in Section 7.1. Note that all the figures show only the initial parts of the signals.

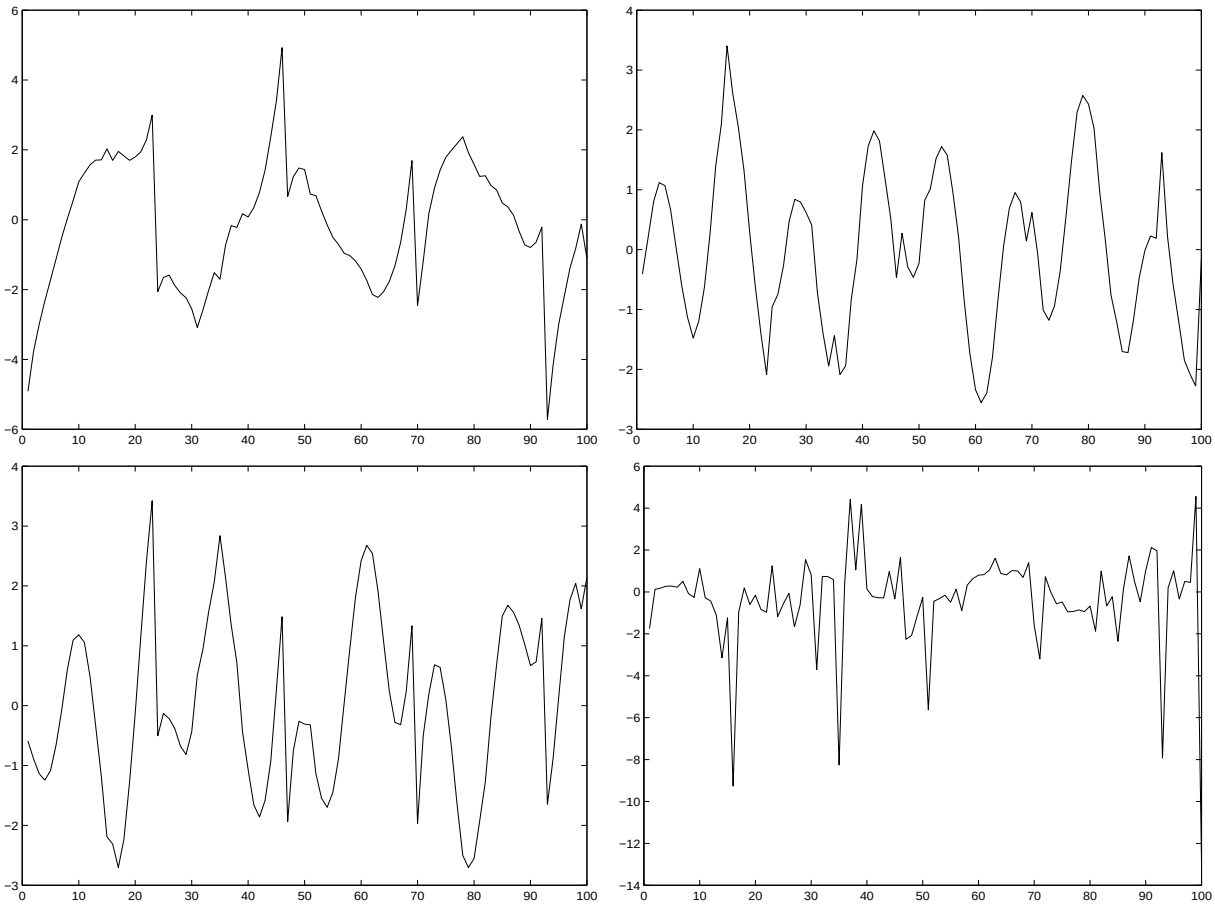

Figure 2: The set of random mixtures of source signals of Fig. 1 used in the following simulations. Heavy outliers, not visible here, were also added to the mixtures. Only this data was observed, and was used to estimate the source signals in Fig. 1

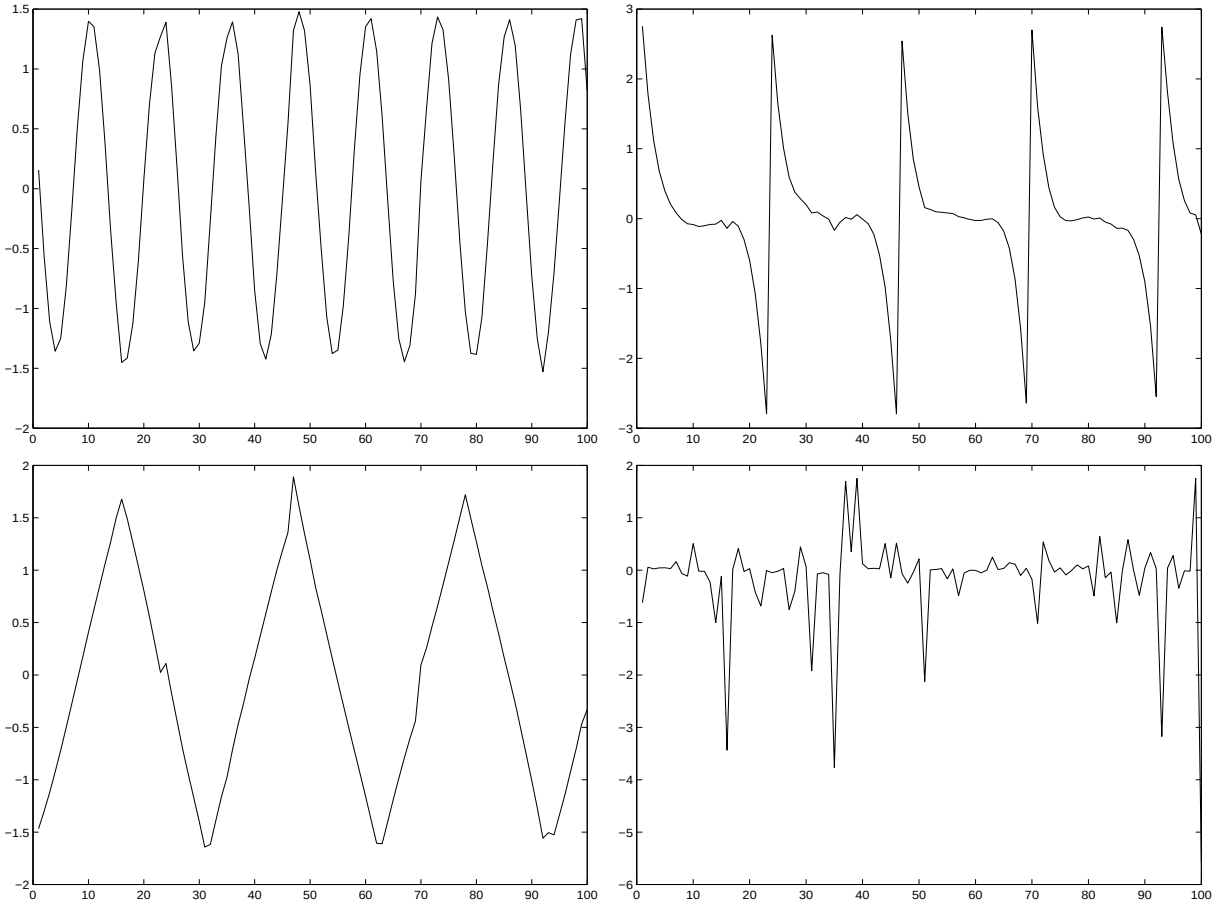

Figure 3: Estimates of source signals of Fig. 1 obtained using the log cosh function in (24) in the presence of outliers. The estimates are quite good in spite of the outliers.

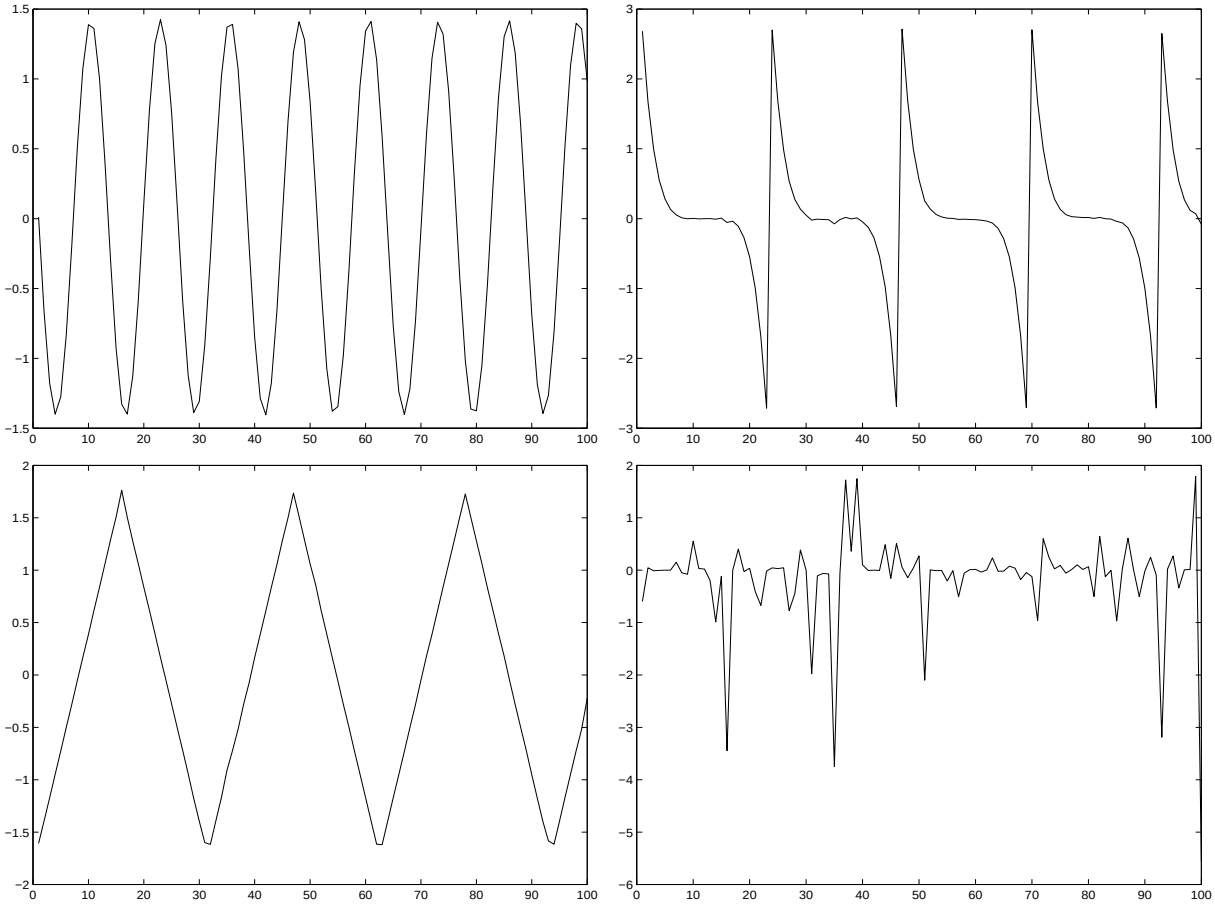

Figure 4: Estimates of source signals of Fig. 1 obtained using the Gaussian function in (25) in the presence of outliers. In spite of the outliers, the results are almost perfect.

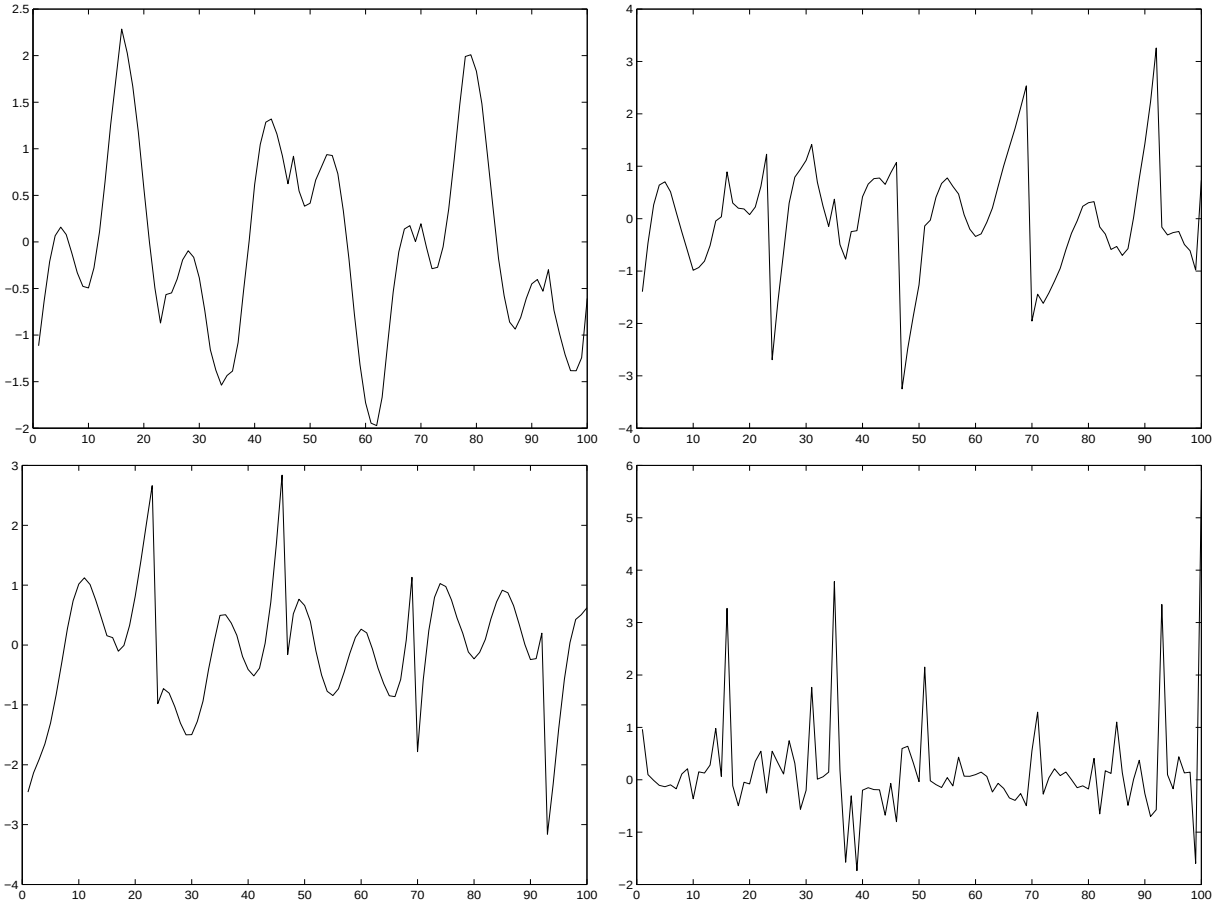

Figure 5: Estimates of source signals of Fig. 1 obtained using kurtosis as in (26) in the presence of outliers. The outliers deteriorated the estimates considerably.

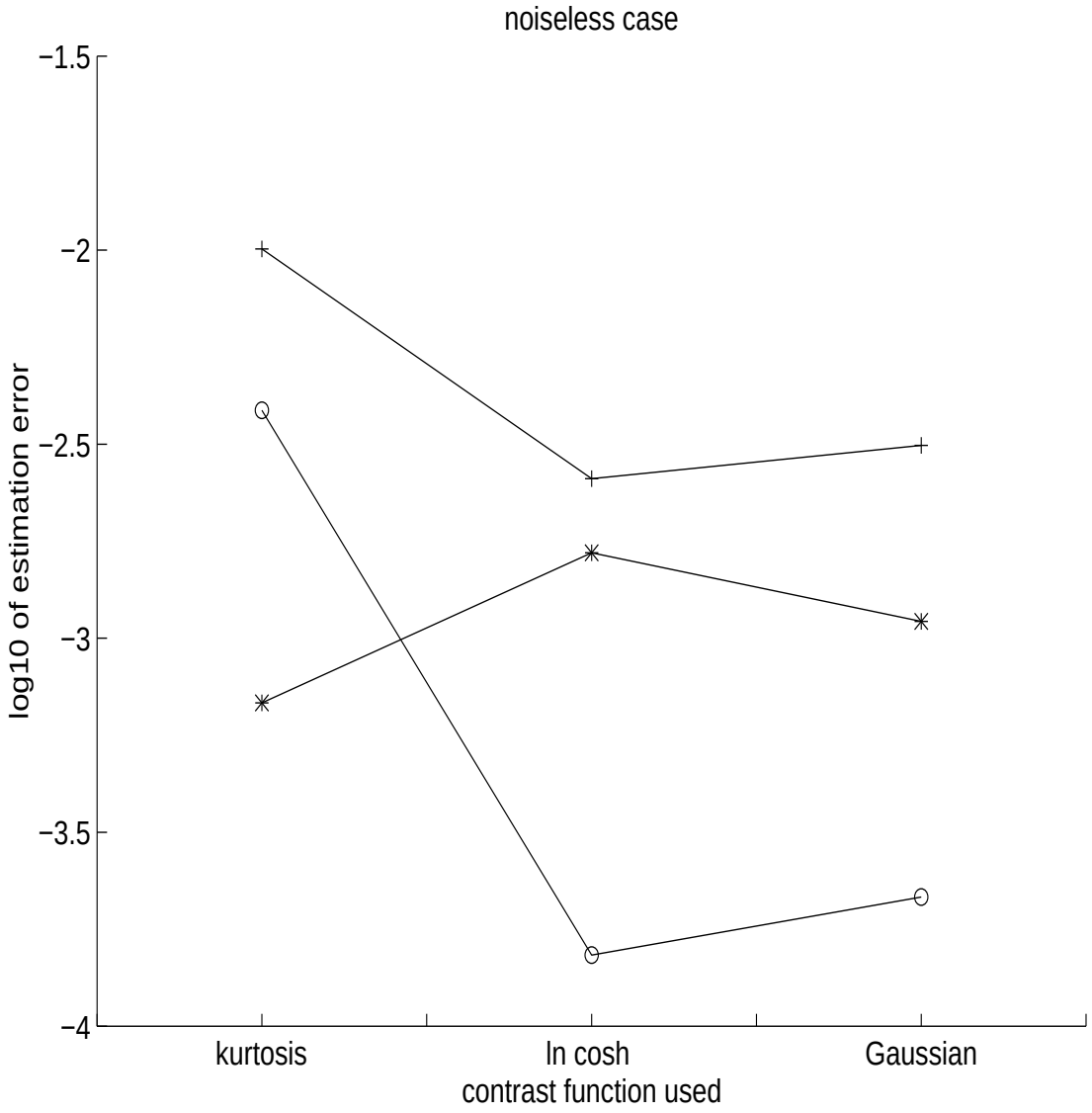

Figure 6: Finite-sample estimation errors plotted for different contrast functions and distributions of the independent components, in the noiseless case. Asterisk: uniform distribution. Plus sign: Double exponential. Circle: cube of Gaussian.

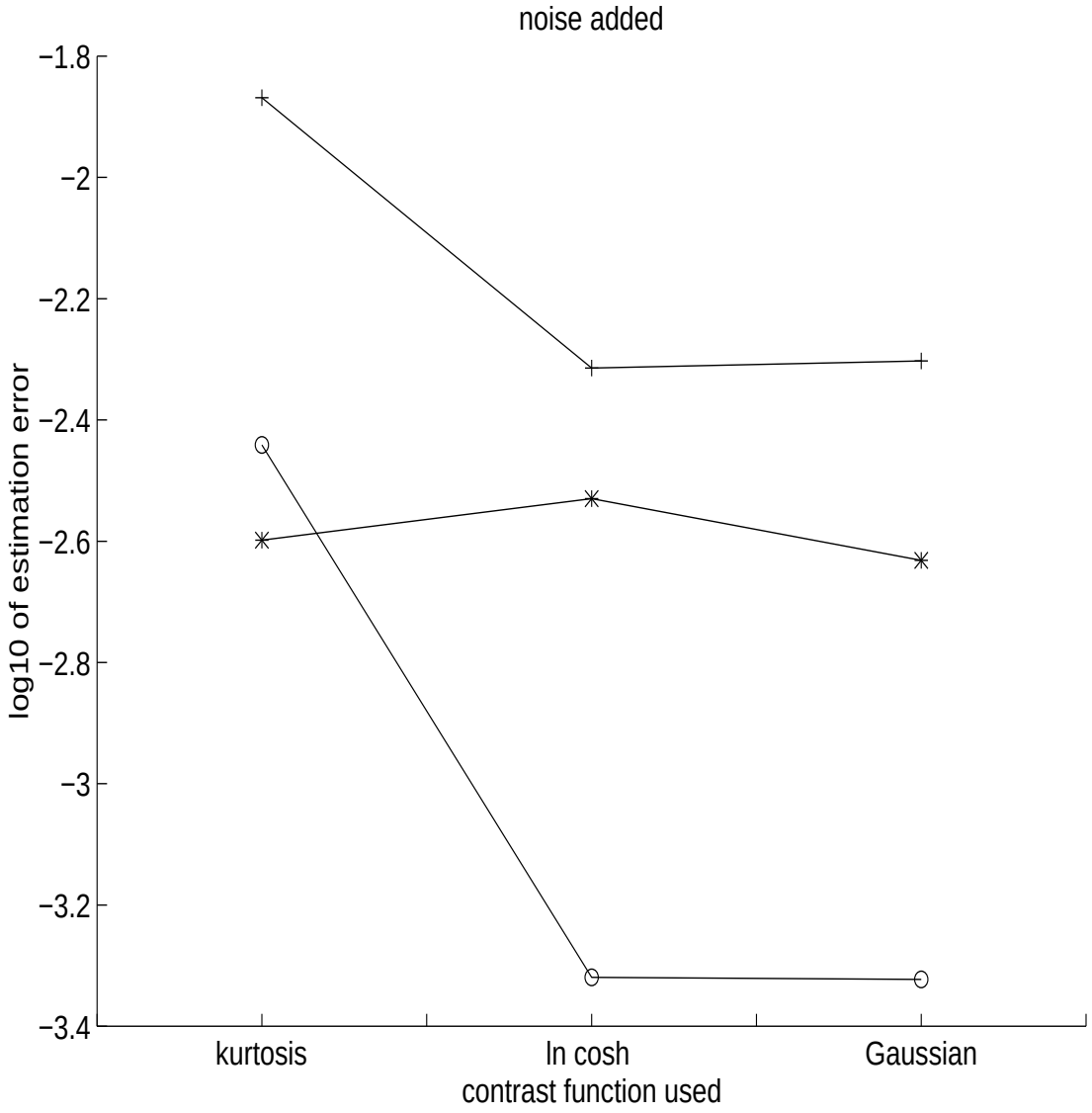

Figure 7: The noisy case. Finite-sample estimation errors plotted for different contrast functions and distributions of the independent components. Asterisk: uniform distribution. Plus sign: Double exponential. Circle: cube of Gaussian.

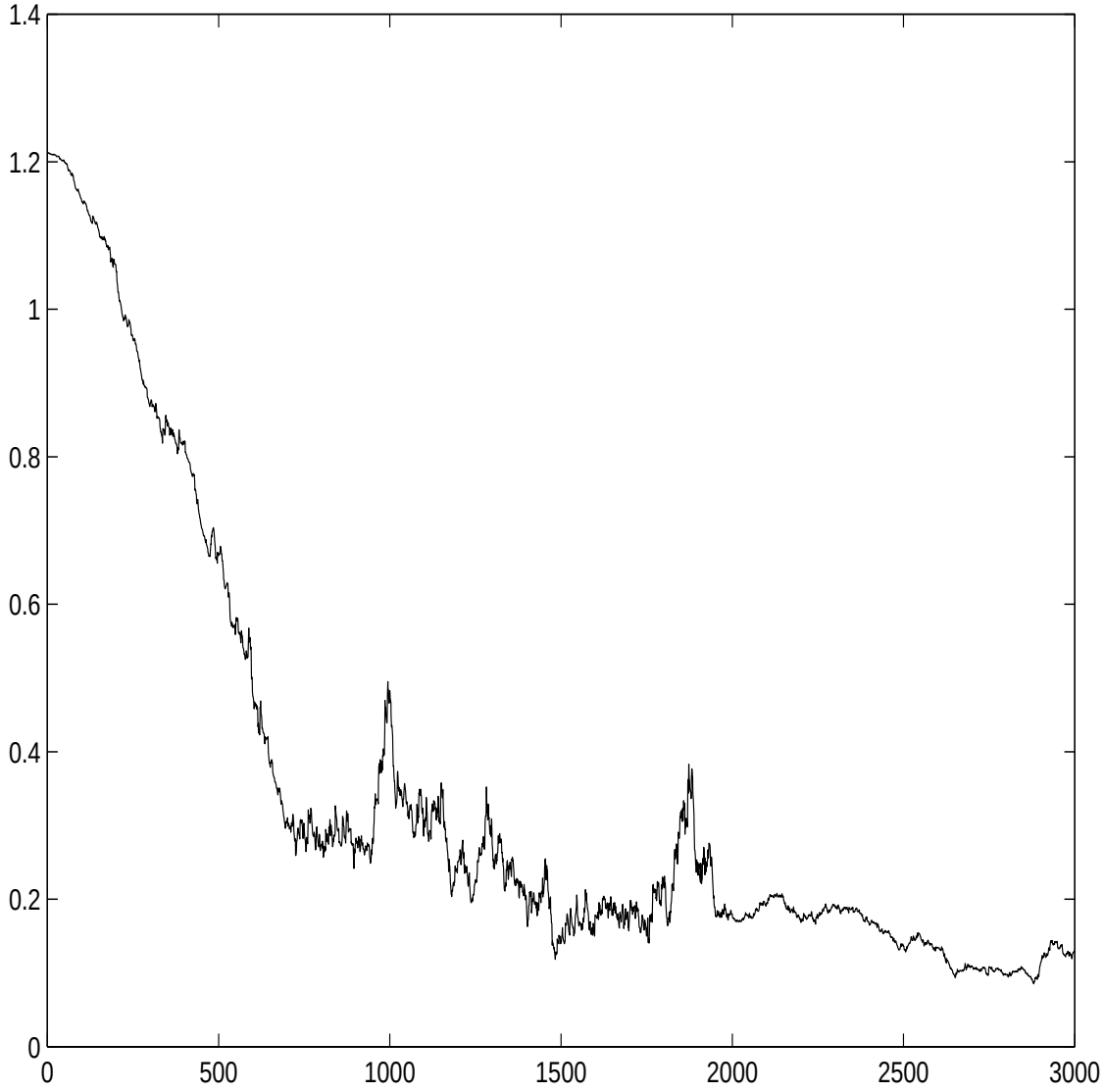

Figure 8: Convergence of the neural learning rule in (30). Approximately 1000 iterations were sufficient for convergence, but this convergence was not very accurate. More accurate results were obtained by decreasing the learning rate after 2000 iterations. The results are averaged over 10 trials.

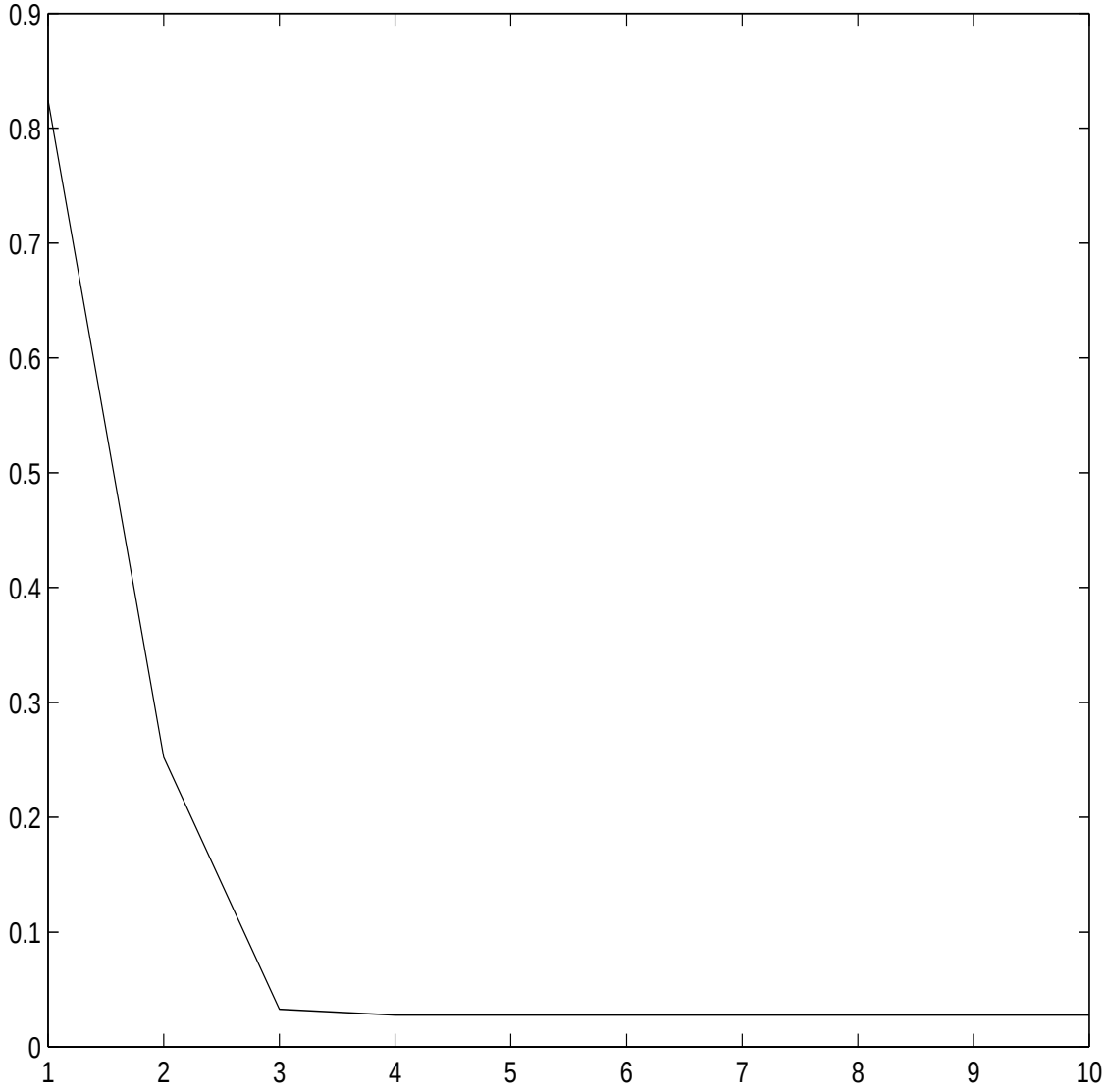

Figure 9: Convergence of the fixed-point algorithm in Section 6.4. Four iterations were enough for convergence, on the average. Also these results are averaged over 10 trials.

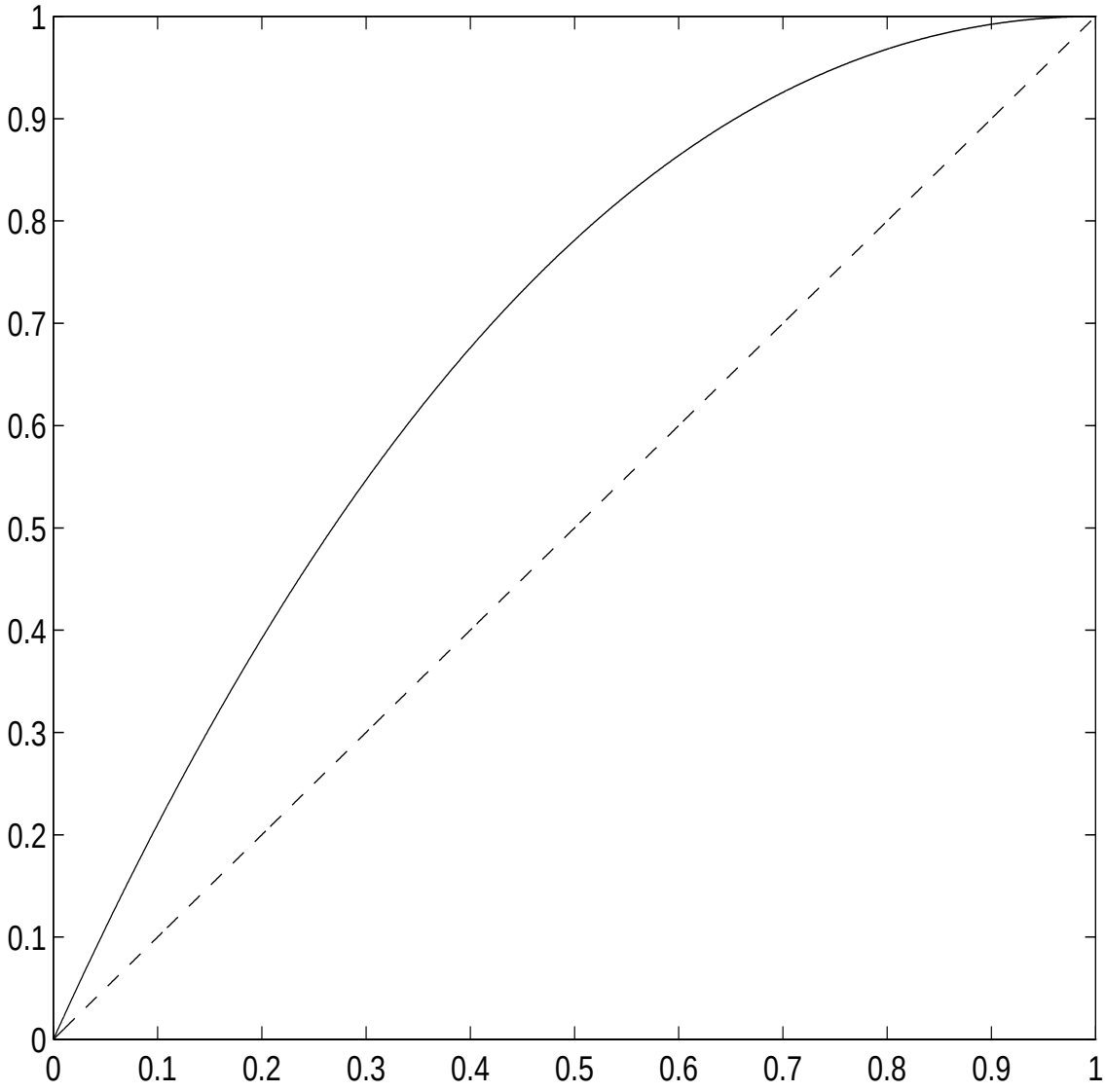

Figure 10: Plot of h in (62). Dashed line is $x = y$. Clearly, iterative application of h leads to convergence to the point 1.