A Comparative Survey on Adaptive Neural Network Algorithms for Independent Component Analysis

Radu Mutihac* and Marc M. Van Hulle

Katholieke Universiteit Leuven, Labo voor Neuro- en Psychofysiologie, Campus Ghastuisberg, B-3000 Leuven, Belgium

{Radu.Mutihac, Marc.VanHulle}@med.kuleuven.ac.be

Abstract. The paper is an overview of the most frequently used neural network algorithms for implementing *Independent Component Analysis* (ICA). The performance of six structurally different algorithms was ranked in blind separation of independent artificially generated signals using the stationary linear ICA model. Ranking of the estimated components was also carried out and compared among different ICA approaches. All algorithms were run with different contrast functions, which were optimally selected on the basis of maximizing the sum of individual negentropies of the network outputs or minimizing their mutual information. Both subgaussian and supergaussian one-dimensional time series were employed throughout the numerical simulations.

Key words: Independent component analysis, entropy, statistical distributions, algorithm.

1 Introduction

In many areas like data analysis, signal processing, and neural networks, a common task is to find an adequate representation of multivariate data for subsequent processing and interpretation. The transformed variables are expected to be the underlying *components* that optimally describe the essential structure of data and to reveal some physical causes involved in the process of data generation. In neural computation this task belongs to unsupervised learning, since the representation is only learned from data without any kind of external control.

Linear transforms are often envisaged to accomplish this task due to their computational and conceptual simplicity. In its full generality, ICA amounts almost to the complex problem of blind model identification on the basis of minimal suppositions. ICA has emerged as an extension of a linear transform called *Principal Component Analysis* (PCA), which has been developed some years ago in context with *Blind Source Separation* (BSS) in *Digital Signal Processing* (DSP) and array processing [1]. The ICA problem is solved on the basis of optimizing certain measures of departure from gaussianity, which leads to a numerical optimization problem. We restricted our study to the

* *Permanent institution*: Faculty of Physics, University of Bucharest, 76900 Romania

linear ICA model, which means that the (unknown) transform from the sources to the observed data and the pseudoinverse transform for estimating the sources from the data are both linear.

2 ICA Model

The statistical principles of the ICA problem are discussed in many seminal papers, but fundamental contributions are due to Comon [1] and Cardoso [2]. Our linear model considered hereafter assumes $\mathbf{x}(t)$, $\mathbf{n}(t) \in \tilde{\mathrm{N}}^N$, and $\mathbf{s}(t) \in \tilde{\mathrm{N}}^M$ three random vectors with zero mean and finite covariance, with the components of $\mathbf{s}(t)$ being statistically independent, whereas **A** is a rectangular constant full column rank $N \times M$ matrix with at least as many rows as columns ($N \geq M$):

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t) = \sum_{i=1}^{M} s_i(t)\,\mathbf{a}_i + \mathbf{n}(t) \tag{1}$$

where $\mathbf{s}(t)$, $\mathbf{x}(t)$, $\mathbf{n}(t)$, and $\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_M\}$ are the sources, the data, the (unknown) noise, and the mixing matrix, respectively (Fig. 1). The sample index (e.g. time or point) *t* is assumed to take discrete values $t = 1,2,...,T$. The variables (time series) are rows in $\mathbf{s}(t)$, $\mathbf{x}(t)$ and $\mathbf{n}(t)$, so that $\mathbf{s}(t)$, $\mathbf{x}(t)$ and $\mathbf{n}(t)$ are column vectors at any *t*. Mixing is supposed to be instantaneous, so there is no time delay between the (latent) source variable $s_i(t)$ mixing into an observable (data) variable $x_j(t)$. Within this framework, the ICA problem can be formulated as follows: given *T* realizations of $\mathbf{x}(t)$, estimate both the matrix **A** and the corresponding realizations of $\mathbf{s}(t)$. In BSS the task is to find the waveforms $\{s_i(t)\}$ of the sources knowing only the mixtures $\{x_j(t)\}$.

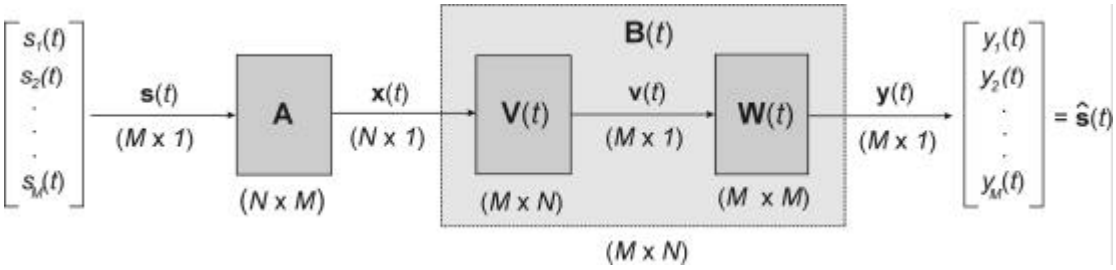

Figure 1. Mixing and separating unknown source signals.

There are several limitations to solving this problem. If no suppositions are made about the noise (which is generally the case), it cannot be introduced in the model but included in the signals, hence the noise-free ICA model can be written in the form:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) = \sum_{i=1}^{M} s_i(t)\,\mathbf{a}_i \tag{2}$$

where the columns $\mathbf{a}_i$, $i = 1,2,...,M$ of the mixing matrix **A** are the basis vectors (mixture coefficients) of ICA. Next limitation refers to the sizes of the different vectors. The size of $\mathbf{s}(t)$ (usually unknown) should not be greater than the size of data $\mathbf{x}(t)$, otherwise the problem becomes underdetermined. If the size of $\mathbf{x}(t)$ is greater than the size of $\mathbf{s}(t)$ (e.g. there are more sensors than sources), the problem is over-determined and the extra data can be used for reducing noise [3]. This is accomplished by projecting the input data $\mathbf{x}(t)$ into its *M*-dimensional signal subspace using for example PCA whitening [4]. In any case, each source signal $s_i(t)$, $i = 1,2,...,M$ is assumed in our model to be a stationary zero-mean stochastic process and only one of them is allowed to have a gaussian distribution.

Adaptive source separation consists in updating a $M \times N$ separating matrix $\mathbf{B}(t)$, without resorting to any information about the spatial mixing matrix **A**, so that the vector

$$\mathbf{y}(t) = \mathbf{B}(t)\mathbf{x}(t) \tag{3}$$

becomes an estimate $\mathbf{y}(t) = \hat{\mathbf{s}}(t)$ of the original independent source signals $\mathbf{s}(t)$; hence the terminology "source separation". This is in contrast with "standard" array processing and beamforming techniques where the columns of **A** or their dependence on the location of the sources is assumed to be known. In neural implementations, $\mathbf{y}(t)$ is the output vector of the network, and the matrix $\mathbf{B}(t)$ is the total weight matrix between the input and the output layers. The estimate $\hat{s}_i(t)$ of the *i*-th source signal may appear in any component $y_j(t)$ of $\mathbf{y}(t)$. The basic indeterminacy of the linear ICA model is due to the fact that any product $\mathbf{A}'\mathbf{s}'(t)$ satisfies the same condition:

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) = \mathbf{A}^{-1}\mathbf{P}\text{? ? }^{-1}\mathbf{P}^T\mathbf{s}(t) = \mathbf{A}'\mathbf{s}'(t) \tag{4}$$

for any permutation matrix **P** and any diagonal scaling matrix **?** , that is, the ICA model can be resolved up to the product of a permutation and a diagonal matrix. This is true for the overall transformation matrix $\mathbf{Q} = \mathbf{B}(t)\mathbf{A}$ as well, because without prior information on the amplitude of the source signals nor on the matrix **A,** the scale of each source signal is unobservable. The permutation indeterminacy stems from the immateriality of labeling the source signals. In order to partly lift up this ambiguity, in several ICA algorithms the amplitude of the estimates $y_j(t)$ are typically centered (zero mean) and scaled to have unit variance (whitened).

2.1 DATA PREPROCESSING

As a prerequisite to apply an ICA method, some algorithms require data preprocessing to some extent. As ICA deals with higher-order statistics it is justified to normalize in some sense the

first- and second-order moments. The effect is that the solution is divided in two parts dealing with dependencies in the first two moments, e.g. the *whitening* or *sphering* matrix $\mathbf{V}(t)$, and the dependencies in higher-order statistics, e.g. the *orthogonal separating matrix* $\mathbf{W}(t)$ in the whitened space. If we assume zero-mean observed data $\mathbf{x}(t)$, then by whitening we get a vector $\mathbf{v}(t)=\mathbf{V}(t)\mathbf{x}(t)$ with decorrelated components, that is, $E\{\mathbf{v}\mathbf{v}^T\}=\mathbf{I}$. The subsequent linear transform defined by $\mathbf{W}(t)$ yields the solution by a rotation $\mathbf{y}(t)=\mathbf{W}(t)\mathbf{v}(t)$, which is the relationship between the whitening and the output layer of the network. The total *separation matrix* between the input and the output layer becomes:

$$\mathbf{B}(t)=\mathbf{W}(t)\mathbf{V}(t) \tag{5}$$

In the standard stationary case, the whitening and the orthogonal separating matrices converge to some constant values during learning. The same model can nevertheless be used in nonstationary situations by keeping these matrices time-varying [5]. Standard PCA is often used for whitening because information can be optimally compressed in the mean-square error sense and filter possible noise out [4]. The PCA whitening matrix is given by

$$\mathbf{V}=\mathbf{D}^{-1/2}\mathbf{E}^T \tag{6}$$

where $\mathbf{E}\mathbf{D}\mathbf{E}^T=E\{\mathbf{x}\mathbf{x}^T\}$ is the eigenvector decomposition of the covariance matrix of the (zero-mean) data **x**, implying that **D** is a $M\times M$ diagonal matrix containing the eigenvalues, and **E** an orthogonal $N\times M$ matrix having the eigenvectors as columns. The appropriate orthogonal transform **W** can be sought by means of (*i*) heuristic independence conditions, (*ii*) optimizing some information-theoretic criteria, or (*iii*) optimizing some suitable contrast functions, in such a way that to come out with as independent as possible outputs.

The core of each algorithm contains the update (learning) rule and its associated optimized criterion. These two items differentiate the neural algorithms, which are actually family of algorithms parameterized by the nonquadratic nonlinearity used. For the large class of ICA adaptive algorithms the update rule is given as $\Delta\mathbf{W}(t)$, which leads to the update equation for $\mathbf{W}(t)$ and/or $\mathbf{B}(t)$:

$$\mathbf{W}(t+1)=\mathbf{W}(t)+\Delta\mathbf{W}(t) \tag{7}$$

As for instance, a simple neural algorithm for adaptive learning of the whitening matrix $\mathbf{V}(t)$ has the form [6]:

$$\mathbf{V}(t+1)=\mathbf{V}(t)+\Delta\mathbf{V}(t)=\mathbf{V}(t)+m(t)\left[\mathbf{I}-\mathbf{v}(t)\mathbf{v}(t)^T\right]\mathbf{V}(t) \tag{8}$$

where the (positive) parameter $m(t)$ sets the learning rate.

There are many methods for estimating the ICA model, which are differentiated by the way the source independence hypothesis is formulated and subsequently exploited. Their common feature is nevertheless some form of higher-order statistics, which specifically means information not contained in the covariance matrix of the observed (zero-mean) data. Therefore, ICA algorithms are linear with respect to data processing but are generally nonlinear in the learning phase. Optimizing some implicit forms of nongaussianity, all ICA approaches come up with different approximations of statistical independence of the estimated components.

3 Neural Network Structures for ICA Algorithms

Feedforward network structures are intensively used in source separation, but recurrent neural network structures have also been considered in solving the BSS problem [7]-[9]. Theoretically, they may exhibit certain advantages in hardware implementation [7]. Some feedforward networks containing several subsequent separation layers have been also reported [10]. It is claimed that they are able to separate sources in difficult cases (weak sources or ill-conditioned problems) provided that the data vectors $\mathbf{x}(t)$ do not contain noise and obey the ICA model above [6].

Let us consider the complete ICA model (1) and rewrite the estimated expansion as follows:

$$\mathbf{x}(t) = \hat{\mathbf{A}}\,\mathbf{y}(t) + \mathbf{n}'(t) = \hat{\mathbf{x}}(t) + \mathbf{n}'(t) \tag{9}$$

where the $N \times M$ matrix $\hat{\mathbf{A}}$ denotes the estimate of the ICA basis matrix **A**, and $\mathbf{n}'(t)$ is the noise term. The *N* inputs of the network (Fig. 2) are the components of the vector $\mathbf{x}(t)$ at discrete values of the sample index *t* and its hidden layer contains *M* units. The ICA model can be implemented in two consecutive stages [4]:

1. Learn a $M \times N$ separating matrix **B** for which the components of $\mathbf{y} = \mathbf{Bx}$ are as independent as possible according to some criterion;
2. Learn a $N \times M$ weight matrix $\hat{\mathbf{A}}$ that minimizes the mean-square error $E\left\{\left\|\mathbf{n}'(t)\right\|^2\right\} = E\left\{\left\|\mathbf{x}(t) - \hat{\mathbf{A}}\mathbf{y}(t)\right\|^2\right\}$ with respect to $\hat{\mathbf{A}}$.

If the network is used for solving the BSS problem, then the basis vectors of ICA are of no interest and the last layer is consequently omitted. If data are sphered, then an extra layer is needed (Fig. 2). The first layer accomplishes $\mathbf{v}(t) = \mathbf{V}\mathbf{x}(t)$, where the $M \times N$ matrix **V** is the whitening matrix. In the case of $N > M$ the matrix **V** simultaneously reduces the dimension of the data vector $\mathbf{x}(t)$. Then the source independent components are separated $\mathbf{y}(t) = \mathbf{W}\mathbf{v}(t)$ by the orthogonal matrix **W** that the network should learn.

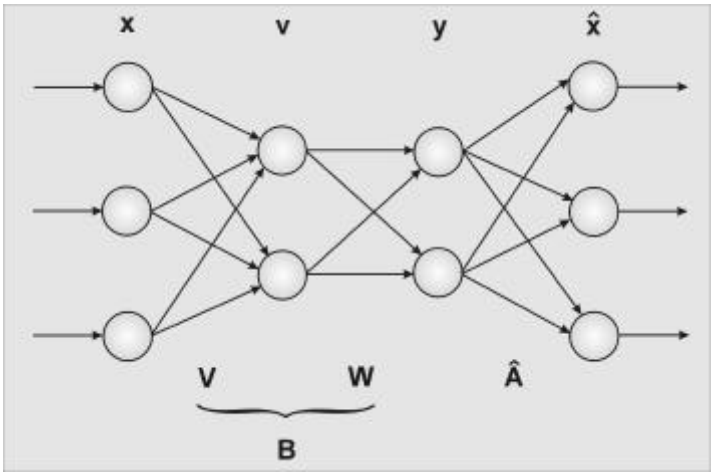

Figure 2. The architecture of a general feedforward neural network performing blind source separation and providing the basis vectors of ICA as columns of the estimated mixing matrix $\hat{\mathbf{A}}$.

Feedback connections are needed during the learning phase, but once the training is over, the network becomes purely feedforward if data are stationary. In most of the cases $M = N$, so that no data compression takes place in the hidden layer. The linear ICA model (1) enforces linear input-output mappings after the learning process, yet nonlinearities must be used during learning the separating matrix **B** or **W** in order to introduce the higher-order statistics into computations.

4 Algorithms for ICA

Apart from the estimation principle of ICA, an algorithm is needed for implementing the necessary computations. Since nonquadratic functions are generally involved by the estimation methods, numerical algorithms are needed, which are quite computationally demanding. The current algorithms for ICA can loosely be classified in two categories. One category contains *adaptive algorithms* generally based on stochastic gradient methods and implemented in neural networks [11], [8], [12], [13], [14], [15], [16]. Adaptive algorithms may also be based either on optimization of cumulant-based contrast functions [8], [12], [13], or on "estimating equations" involving nonlinear distortions of the output $y(t)$ [8], [17]. This class of algorithms exhibits slow convergence and their convergence depends crucially on the correct choice of the learning rate parameters.

The other category relies on *batch computation* minimizing or maximizing some relevant criterion functions [1], [18], [19]. Their main drawback is that generally they imply complex matrix or tensorial operations. Neuromorphic block technique algorithms based on *2*nd- and *4*th-order cumulants were also proposed [20], [21], [22], [23], or likelihood approach [24]. Note that if the

source signals are temporally correlated, then the spatial blind deconvolution (separation) may be based on *2*nd-order statistics only [25], [26], and separation of gaussian sources is possible.

We selected hereafter a number of algorithms aiming to highlight various facets of adaptive neural network approach to solve the ICA problem in the framework of BSS.

4.1 HÉRAULT-JUTTEN (HJ) ALGORITHM

In this pioneering neural separation algorithm [8], the separation matrix **B** is sought in the form

$$\mathbf{B} = (\mathbf{I} + \mathbf{S})^{-1} \tag{10}$$

and the off-diagonal elements of **S** are updated following the rule:

$$\mathbf{S}(t+1) = \mathbf{S}(t) + m(t)\,\mathbf{g}(\mathbf{y}(t))\,\mathbf{h}(\mathbf{y}(t)^T) \tag{11}$$

Here the diagonal elements of **S** are zero, while $\mathbf{g}(\mathbf{y}(t)) = \{g(y_i(t))\}$ is a column vector, $\mathbf{h}(\mathbf{y}(t)^T) = \{h(y_i(t))\}$ is a row vector with $i = 1,2,...,N$, and the learning parameter $m(t) > 0$. Various different odd functions $g(y)$ and $h(y)$ were used, such as y, y^3, $sign(y)$ and $tanh(y)$. The neural implementation of the HJ algorithm may be performed using either feedback or feedforward type architectures [12]. The HJ algorithm initially used for the separation of *2* sources only, was subsequently improved and extended for convolutive mixtures (time delays) [27].

4.2 EASI (PSF) ALGORITHM

The *Parameter Free Separators* (PFS) algorithm was introduced by Laheld and Cardoso [28] and concluded with the *Equivariant Adaptive Separation* via *Independence* (EASI) algorithm [29], which is an example of a nonlinear decorrelation method. The principle behind equivariance emerged from the group theory: when a transformation on the data is equivalent to the transformation of the parameter, the notion of equivariance is of relevance. Since both mixing and separating transforms are invertible linear transforms, the properties of the parameters can be seen through the whole process. Though justified as an adaptive signal processing algorithm, it can be used as a learning algorithm of a nonlinear PCA type network.

The general update rule of the separating matrix equates to:

$$\mathbf{B}(t+1) = \mathbf{B}(t) - l(t)\left[\mathbf{y}(t)\mathbf{y}(t)^T - \mathbf{I} + \mathbf{g}(\mathbf{y}(t))\,\mathbf{h}(\mathbf{y}(t)^T) - \mathbf{h}(\mathbf{y}(t))\,\mathbf{g}(\mathbf{y}(t)^T)\right]\mathbf{B}(t) \tag{12}$$

with $l(t)$ setting the learning rate. If $g(y) = y$ and $h(y) = tanh(y)$ for subgaussian sources, and $g(y) = tanh(y)$ and $h(y) = y$ for supergaussian sources, then separation is accomplished without

extra stabilization. Adding on-line estimation of the kurtosis endows the EASI algorithm with the ability to handle simultaneously both sub- and supergaussian sources. The EISA algorithm provides *uniform performance*, that is, its performance does not depend on the mixing matrix [28].

4.3 THE BIGRADIENT ALGORITHM

The bigradient algorithm was introduced by Wang, Karhunen, and Oja [30]. The update rule for the orthogonal separation matrix **W** is given by:

$$\mathbf{W}(t+1) = \mathbf{W}(t) + m(t)\,\mathbf{v}(t)\,\mathbf{g}\left(\mathbf{y}(t)^T\right) + g(t)\,\mathbf{W}(t)\left[\mathbf{I} - \mathbf{W}(t)^T\,\mathbf{W}\right] \tag{13}$$

The learning parameters are such as $m(t)$ can be either positive or negative and $g(t)$ has practically the values *0.5* or *1*. The first update term $m(t)\,\mathbf{v}(t)\,\mathbf{g}\left(\mathbf{y}(t)^T\right)$ is a nonlinear hebbian term, whereas the second term $g(t)\,\mathbf{W}(t)\left[\mathbf{I} - \mathbf{W}(t)^T\,\mathbf{W}\right]$ keeps the weight matrix $\mathbf{W}(t)$ approximately orthonormal. The main quality of this algorithm is its flexibility, namely, with minor changes can perform separation of both subgaussain and supergaussian sources, but also to standard PCA and MCA (*Minor Component Analysis*). In terms of neural implementation, the update rule (13) can be modified easily so that the weight vectors of the neurons, which are columns in the matrix $\mathbf{W}(t)$, are computed sequentially in a hierarchic order [6].

4.4 FIXED-POINT (FP) ALGORITHMS

Fixed-point algorithms are searching the ICA solution by minimizing mutual information among the estimated components [18]. Mutual information is defined by means of approximating the (differential) entropy using the maximum entropy principle. Ultimately, the task reduces to separately maximizing the negentropy of each component, which is a rather delicate problem. Convergence is at least quadratic.

The FastICA learning rule finds a direction (i.e. a unit vector **w**) in such a way that the projection of $\mathbf{w}^T\mathbf{x}$ maximizes a particular contrast function $J_G(\mathbf{w})$:

$$J_G(\mathbf{w}) = \left[E\left\{G\left(\mathbf{w}^T\mathbf{x}\right)\right\} - E\{G(\mathbf{?})\}\right]^2 \tag{14}$$

where the variance of $\mathbf{w}^T\mathbf{x}$ must here be constrained to unity. The following optimal choices of the function *G* were proposed by Hyvärinen [31]:

$$G_1(u) = (1/a_1)\,log\ cosh\ a_1 u\,, \quad G_2(u) = (-1/a_2)\,exp\left(-a_2\,u^2/2\right), \quad G_3(u) = u^4/4 \tag{15}$$

where $1 \le a_1 \le 2$, $a_2 \cong 1$ are some suitable constants. The expectations are in practice replaced with their sample means, hence the FP algorithm is not a truly neural adaptive algorithm. The learn-

ing rule consists of a decorrelation method, e.g. a deflation scheme based o a Gram-Schmidt-like decorrelation, which amounts to estimating the components one by one. Assume that we have estimated p vectors (independent components) $\mathbf{w}_1,\mathbf{w}_2,...,\mathbf{w}_p$, we run the one-unit FP algorithm for $\mathbf{w}_{p+1}$ and after every iteration step substract from $\mathbf{w}_{p+1}$ the *projections* $\mathbf{w}_{p+1}^T\mathbf{w}_j\mathbf{w}_j$, $j=1,2,...,p$ of the previously estimated p vectors, and the renormalize $\mathbf{w}_{p+1}$:

$$1.\ \mathbf{w}_{p+1} = \mathbf{w}_{p+1} - \sum_{j=1}^{p} \mathbf{w}_{p+1}^T \mathbf{w}_j \mathbf{w}_j \qquad 2.\ \mathbf{w}_{p+1} = \frac{\mathbf{w}_{p+1}}{\sqrt{\mathbf{w}_{p+1}^T \mathbf{w}_{p+1}}} \tag{16}$$

There is available also a *stabilized* version of the FP algorithm developed in order to ameliorate the rather uncertain convergence of the Newton method on which the FP algorithm is based. *Symmetric* decorrelation is also possible when there are no particular directions envisaged. Although the algorithm was motivated as a short-cut method to make neural learning for kurtosis minimization and maximization faster, its convergence was proved independent of the neural algorithm and the well-known results on the connection between ICA and kurtosis [18].

4.5 INFOMAX (BS) ALGORITHM

An important class of algorithms is based on maximization of network entropy (*infomax*) [11], which is, under some conditions, equivalent to the maximum likelihood (ML) approach. The basic idea of *infomax* is to match the slope of the nonlinear transfer function g with the input probability density function, which in one dimensionally may be written as $y = g(x;w) \cong \int_{-\infty}^{x} f_x(x)\,dx$

The Bell Sejnowski (BS) nonlinear information maximization algorithm performs online stochastic gradient ascent in the mutual information between outputs and inputs of a network. By minimizing the mutual information between its outputs, the network factorizes the input into independent components. Considering a network with the input vector $\mathbf{x}$, a weight matrix $\mathbf{W}$, and a monotonically transformed output vector $\mathbf{y} = g(\mathbf{W}\mathbf{x} + \mathbf{w}_0)$, then the resulting learning rule for the weights and bias-weight, respectively, are [4]:

$$\Delta\mathbf{W} \propto \left[\mathbf{W}^T\right]^{-1} + \mathbf{x}(\mathbf{1} - 2\mathbf{y})^T, \qquad \Delta\mathbf{w}_0 \propto \mathbf{1} - 2\mathbf{y} \tag{17}$$

In the case of bounded variables, the interplay between the *antihebbian* term $\mathbf{x}(\mathbf{1}-2\mathbf{y})^T$ and the *antidecay* term $\left[\mathbf{W}^T\right]^{-1}$ produces an output pdf that is close to the flat constant distribution, which corresponds to the maximum entropy distribution.

Originally, the nonlinearity proposed for the algorithm was the typical sigmoidal function used in neural networks:

$$g(y) = \frac{1}{1 + exp(-y)} \tag{18}$$

but later on, in its extended version, a better performing form was introduced:

$$g(y) = y \pm tanh(y) \tag{19}$$

The updated informax ICA algorithm has been refined with the natural gradient feature of Amari, Cichocki & Yang [15], the extended-ICA algorithm of Lee, Girolami & Sejnowski [32], and PCA dimension reduction.

4.6 NATURAL GRADIENT ALGORITHM (ACY ALGORITHM)

Amari, Cichocki and Yang [15] altered the BS algorithm by using the natural gradient instead of the stochastic gradient reducing the complexity of computations and significantly improving the speed of convergence. The on-line ACY algorithm minimizes the statistical dependency among outputs of the network, which is measured by their average mutual information. The Gram-Charlier expansion instead of the Edgeworth expansion was employed in evaluating the marginal entropy for the estimation of the MI. The dependency can be alternatively expressed by the KL divergence between the joint and the product of the marginal distributions of the outputs, which is simply related with the average MI. The KL divergence supplementary has some invariant properties from the differential-geometrical point of view [26]. The minimization of KL divergence comes out with an ICA algorithm for estimating the separating matrix in the whitened space:

$$\Delta \mathbf{W}(t) \propto \left[\mathbf{I} - g(\mathbf{W}(t)\mathbf{x}(t)) \; (\mathbf{W}(t)\mathbf{x}(t))^T \right] \mathbf{W}(t) \tag{20}$$

Two different functions were used instead of the theoretical contrast function proposed by the authors [15] because of too high powers involved that cause in practice a strong dependence on only a few large values:

$$g(y) = \begin{cases} tanh(y), & \text{supergaussian components} \\ y^3, & \text{subgaussian components} \end{cases} \tag{21}$$

The algorithm has the property of equivariance and can be easily implemented on a neural network-like model.

4.7 MAXIMUM LIKELIHOOD (ML)

Maximizing the likelihood can be seen as minimizing the KL divergence between the hypothesized distribution of the sources and the empirical distribution of the output. Two algorithms

performing ICA for source separation, which combine the techniques of fixed nonlinear contrast functions and the ML approach, were included in our tests. They are called the *Pearson-system ICA* algorithm [33] and the *Extended Generalized Lambda Distribution* (EGLD) algorithm [34]. In the ML approach the source distributions are estimated by a parametric model and the score function of the hypothesized source distribution is used as a contrast function which is subject to optimization by a suitable algorithm. Many different approaches to solving the ICA model are essentially equivalent, in the sense that they conclude with the same iterative algorithm [17].

The Pearson system is a parametric family of distributions that may be used to model a wide class of source distributions that may be symmetric or asymmetric, covering a large range of different values of kurtosis and skewness. The benefit of using Pearson system consists in its ability to separate sources with skewed distributions and the same kurtosis as the normal distribution. The score function of the Pearson system is:

$$j(y) = -\frac{y-a}{b_0 + b_1 y + b_2 y} \tag{22}$$

where a, b_0, b_1 and b_2 are the parameters of the distribution, which may be estimated by the method of moments [35]. The algorithm optimizing this criterion could be any suitable ICA algorithm using ML contrasts like the natural gradient [36], or the relative gradient [2] algorithm with the update rule:

$$\mathbf{W}(t+1) = \mathbf{W}(t) + h\left[I - j(\mathbf{y})\mathbf{y}^T\right]\mathbf{W}(t) \tag{23}$$

where $j(\mathbf{y})$ is the vector of score functions and h is the learning rate, or, as in our experiments, the fixed-point algorithm [18] with the update rule in the form:

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \mathbf{D}\left[E\left\{j(\mathbf{y})\mathbf{y}^T\right\} - diagE\left\{j(y_i)y_i\right\}\right]\mathbf{W}(t) \tag{24}$$

where $\mathbf{D} = diag\left[1/\left(E\left\{j(y_i)y_i\right\} - E\left\{j'(y_i)\right\}\right)\right]$.

The extended generalized lambda distribution is used as in Pearson-system based method for modeling the source distributions in a ML approach to ICA. EGLD modeling provides a useful connection between practical estimator and theoretical measure of independence [34]. EGLD is a large family of distributions covering the whole space of the third and fourth moments, having extensive use in fitting empirical data and in artificial generation of various distributions [37], [38]. The latest extension due to Karian and Dudewicz [39] is a combination of *Generalized Lambda Distribution* (GLD) and *Generalized Beta Distribution* (GBD). The GLD is defined by the inverse distribution function:

$$F^{-1}(p) = l_1 + \frac{p^{l_3} - (1-p)^{l_4}}{l_2} \tag{25}$$

where $0 \le p \le 1$ and l_1, l_2, l_3 and l_4 are the parameters of the distribution. In practice, it is not possible to present the density or distribution functions of GLD in an analytical general form; fortunately, observations are easily generated from GLD employing the inverse distribution function [34]. On the other hand, the GBD is characterized by the density function:

$$f(x) = C b_2^{-(b_4+1)} (x - b_1)^{b_3} (b_1 + b_2 - x)^{b_4} \tag{26}$$

on the interval $[b_1, b_1 + b_2]$ and zero elsewhere, whereas *C* is a constant.

The estimation of the parameters for both GLD and GBD is done using the method of moments, where the four first sample moments are calculated from the data. However, due to the intricacy of the computational process, the parameters for both GLD and GBD are tabulated as functions of the third and fourth moments [40], assuming variables of zero mean and unit variance. The underlying source distributions are estimated through the marginal distributions by fitting them to EGLD family using the method of moments. When the EGLD is fitted to data, the choice between GLD and GBD is made based on the values of the third and fourth moments. The score function $j(\mathbf{y})$ of the EGLD stands as an ICA contrast function for maximization. The optimizing algorithm can be any adequate algorithm where ML contrasts are utilized. We preferred the FastICA algorithm with the convergence criterion set to the symmetric mode, e.g. $e = 0.000001$.

5 Basis Vectors of ICA

The basis vectors $\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_M$ (mixture coefficients) of the linear ICA model provide in many cases a more meaningful characterization of data than the classical principal components. If the weight matrix $\mathbf{B}(t)$ converges to a separating solution **B**, then the estimates of the basis vectors $\{\mathbf{a}_i\}$ are the columns of the pseudoinverse $\hat{\mathbf{A}} = \mathbf{B}^T (\mathbf{B}\mathbf{B}^T)^{-1}$, which reduces to $\hat{\mathbf{A}} = \mathbf{B}^{-1}$ if **B** is a square matrix (e.g. $M = N$). This method is nevertheless unsuited for neural implementation due to the required matrix inversion.

A simple stochastic gradient algorithm was developed [41] for neural estimation of basis vectors in independent component analysis:

$$\hat{\mathbf{A}}(t+1) = \hat{\mathbf{A}}(t) + m(t)\left[\mathbf{x}(t) - \hat{\mathbf{A}}(t)\mathbf{y}(t)\right]\mathbf{y}(t)^T \tag{27}$$

based on the minimization of the mean-square representation error $E\left\{\left\|\mathbf{x}(t)-\hat{\mathbf{A}}\mathbf{y}(t)\right\|^2\right\}$ with a positive rate of learning $m(t)>0$. As before, the columns of the matrix $\hat{\mathbf{A}}(t)$ are the estimates of the basis vectors of the linear ICA model after convergence.

6. Assessment of ICA Algorithms

6.1 STATISTICAL INDEPENDENCE

If the multidimensional random variable $\mathbf{x}\in\tilde{\mathrm{N}}^N$ has a probability density function $f_{\mathbf{x}}(\mathbf{x})$, then the independence of the *N* scalar random variables x_i, $i=1,2,...,N$, that is, the components of **x** having the probability density functions $f_{x_i}(x_i)$, respectively, is defined by the factorization:

$$f_{\mathbf{x}}(\mathbf{x})=\prod_{i=1}^{N} f_{x_i}(x_i) \tag{28}$$

The factorization of $f_{\mathbf{x}}(\mathbf{x})$ appears as a way of splitting a multivariate function into univariate functions, which can subsequently be analyzed and processed separately. In many problems the independence is invoked aiming to enforce computational tractability.

A meaningful treatment of the concept of independence relies on information theory, which means deriving the criterion for statistical independence from the statistical properties of data [42]. Entropy is such a criterion based on the amount of information contained in some occurrences of a random variable. Entropy does not exist for continuous random variables, but *differential entropy* does. In the case of a multidimensional continuous random variable **x** with density $f_{\mathbf{x}}(\mathbf{x})$, the differential entropy is defined as:

$$H(\mathbf{x})=-\int f_{\mathbf{x}}(\mathbf{u})\,log\; f_{\mathbf{x}}(\mathbf{u})\,d\mathbf{u} \tag{29}$$

Differential entropy is invariant to orthogonal transforms and it is upper bound, but is no longer invariant to invertible transforms as entropy is in the case of discrete random variables. Therefore, two other concepts are employed as contrast functions that are endowed with the invariance property, namely *negentropy* and *mutual information*.

First, recall that the Kullback-Leibler divergence of two random variables **v** and **z**, with densities $f_{\mathbf{v}}(\mathbf{u})$ and $f_{\mathbf{z}}(\mathbf{u})$, respectively, is given by

$$K(\mathbf{v}\,|\,\mathbf{z})=K\left(f_{\mathbf{v}}(\mathbf{u})\,|\,f_{\mathbf{z}}(\mathbf{u})\right)=\int_{\mathbf{u}} f_{\mathbf{v}}(\mathbf{u})\,log\,\frac{f_{\mathbf{v}}(\mathbf{u})}{f_{\mathbf{z}}(\mathbf{u})}\,d\mathbf{u} \tag{30}$$

Though not really a distance since it is not symmetric, it turns out that the Kullback-Leibler divergence behaves as a statistical measure of the "distance" between two distributions. In this sense, we

will call it KL distance between two random vectors. KL distance is always nonnegative and takes the value *0* iff the distributions are identical.

For a multidimensional continuous random variable **x** with the density $f_{\mathbf{x}}(\mathbf{x})$, to which is associated a gaussian variable $\mathbf{x}_G$ with the same covariance matrix like **x**, the *negentropy* is defined as:

$$J(\mathbf{x}) = H(\mathbf{x}_G) - H(\mathbf{x}) = \int f_{\mathbf{x}}(\mathbf{u}) log \frac{f_{\mathbf{x}}(\mathbf{u})}{f_{\mathbf{x}_G}(\mathbf{u})} d\mathbf{u} = K(\mathbf{x} \mid \mathbf{x}_G) \tag{31}$$

Negentropy is always nonnegative, reaches its minimum for a gaussian random variable, and is invariant to linear invertible transforms. In the above formula, the negentropy appears as a measure of the distance from normality of the density $f_{\mathbf{x}}(\mathbf{x})$, which is expressed in the form of KL distance. Accordingly, the negentropy $J(\mathbf{x})$ can be used as a measure of departure from gaussianity of a multivariate continuous random variable **x**.

The average mutual information is related with entropy, and in the case of two variables it measures the amount of information conveyed by an occurrence of one variable about the other:

$$I(x,y) = H(x) + H(y) - H(x,y) \tag{32}$$

where $H(x,y)$ is the joint entropy of the variables. For the general case of *N* (scalar) random variables x_i, $i = 1,2,...,N$ the mutual information is

$$I(x_1, x_2, ..., x_N) = \sum_i H(x_i) - H(\mathbf{x}) = \int f_{\mathbf{x}}(\mathbf{x}) log \frac{f_{\mathbf{x}}(\mathbf{x})}{\prod f_{\mathbf{x}_i}(x_i)} d\mathbf{x} = K\left(x \mid \prod x_i \right) \tag{33}$$

It comes out that mutual information is symmetric, zero iff the factorization of the joint density $f_{\mathbf{x}}(\mathbf{x})$ holds, that is, when the components are independent, and it is strictly positive otherwise. The independence condition can now be formulated using the information concepts. Comparing the form of negentropy as in (31) and the mutual information (33), it comes out that if a gaussian multivariate is a reasonable approximation to the product of the marginal densities, then negentropy is a means to estimate the mutual information and, implicitly, a measure of independence.

If we consider now the basic linear ICA model $\mathbf{y} = \mathbf{B}\mathbf{x}$, the mutual information between estimated independent components y_i, $i = 1,2,...,N$ becomes:

$$I(y_1, y_2, ..., y_N) = \sum_{i=1}^{N} H(y_i) - H(\mathbf{y}) = \sum_{i=1}^{N} H(y_i) - H(\mathbf{x}) - log|det(\mathbf{B})| \tag{34}$$

If the scalar random variables y_i, $i = 1,2,...,N$ are constrained to be uncorrelated, then $\mathbf{I} = E\{\mathbf{y}\mathbf{y}^T\} = \mathbf{B}E\{\mathbf{x}\mathbf{x}^T\}\mathbf{B}^T$, hence $det\,\mathbf{I} = 1 = (det\,\mathbf{B})(det\,E\{\mathbf{x}\mathbf{x}^T\})(det\,\mathbf{B}^T)$, which implies that $det\,\mathbf{B}$ must be constant.

$$I(y_1, y_2, ..., y_N) = \sum_{i=1}^{N} H(y_i) + constant \tag{35}$$

where the constant that does not depend on **B** [44]. Moreover, since the y_i's are assumed of unit variance, entropy and negentropy differ only by a constant and the sign. It comes out the fundamental relationship between negentropy and mutual information:

$$I(y_1, y_2, ..., y_N) = -\sum_{i=1}^{N} J(y_i) + constant \tag{36}$$

It explicitly means that minimizing pairwise the MI of the random variables $y_1, y_2, ..., y_N$ equates to maximizing the sum of their individual negentropies $H(y_i) = -E\{log\ f_{y_i}(y_i)\}$. But because a gaussian density has maximal (differential) entropy, this means *minimizing the gaussianities* of the random variables y_i, $i = 1,2,...,N$.

Note that, if instead of the *estimated* densities $f_{y_i}(y_i)$ we plug in the *true* densities, say $f^*_{y_i}(y_i)$, of the sources s_i, $i = 1,2,...,N$, then the objective function in (35) may be put in the form:

$$I(y_1, y_2, ..., y_N) = \sum_{i=1}^{N} \left[-E\{log\ f^*_{y_i}(y_i)\} \right] + constant \tag{37}$$

This is the negative of the log-likelihood (for large sample size), thus minimizing MI is equivalent to *Maximum Likelihood*.

The approximations for negentropy introduced by Hyvärinen [31] for a scalar random variable *y* with zero mean, unit variance, and *p* functions G_i are:

$$J(y) \approx \sum_{i=1}^{p} k_i \left[E\{G_i(y)\} - E\{G_i(y_G)\} \right]^2 \tag{38}$$

where G_i are practically any nonquadratic functions, k_i are some positive constants, and y_G is a gaussian variable with zero mean and unit variance as *y*. When using only two nonlinear functions, e.g. an odd one, $G_1 = y\,exp(-y^2/2)$, which measures the asymmetry, and an even one, $G_2 = |y|$, which measures the sparsity/bimodality of an *1D* nongaussian distribution [31], the approximation of entropy becomes simpler:

$$H(y) \approx H(y_G) - \left[k_1 (E\{G_1(y)\})^2 + k_2 (E\{G_2(y)\} - E\{G_2(y_G)\})^2 \right] \tag{39}$$

Based on some heuristic grounds, the simplified forms of two practical implementations of (39) used in our experimental evaluations were the following:

$$\begin{aligned} H_a(y) &= H(y_G) - \left[k_1 \left[E\left\{ y \exp\left(-y^2/2\right) \right\} \right]^2 + k_2^a \left[E\left\{ |y| \right\} - \sqrt{2/p} \right]^2 \right] \\ H_b(y) &= H(y_G) - \left[k_1 \left[E\left\{ y \exp\left(-y^2/2\right) \right\} \right]^2 + k_2^b \left[E\left\{ \exp\left(-y^2/2\right) \right\} - \sqrt{1/2} \right]^2 \right] \end{aligned} \tag{40}$$

with $k_1 = 36/\left(8\sqrt{3} - 9\right)$, $k_2^a = 1/\left(2 - 6/p\right)$, $k_2^b = 24/\left(16\sqrt{3} - 27\right)$ and where the expression $H(y_G) = \frac{1}{2}(1 + log(2p))$ is the entropy of a standardized scalar gaussian variable. We based our ranking on the strict monotonicity of negentropy (38) as function of the gaussian character of the estimated independent components. Though the absolute values of negentropy are eventually questionable and not entirely reliable, the monotonicity of negentropy is able to relatively rank the performance of various algorithms.

6.2 KNOWN SOURCES

When the original source components are known (such as in the case of artificially generated data, the accuracy of separating power of the independent components of an ICA algorithm can be measured by means of various indexes. We will assume hereafter $M = N$. One index used as a global figure of merit for the separation performance may be defined as *signal-to-interference ratio*:

$$SIR = -\frac{1}{N} \sum_{i=1}^{N} 10\, log_{10} \frac{max(Q_i)^2}{Q_i^T Q_i - max(Q_i)^2} \quad \text{(dB)} \tag{41}$$

where $\mathbf{Q} = \mathbf{BA}$ is the overall transforming matrix of the source components, Q_i is the *i*-th column of **Q,** $max(Q_i)$ is the maximum element of Q_i, and *N* is the number of source signals. The higher *SIR* is, the better the separation performance of the algorithm. A second index, CTE, which was used to measure the accuracy of retrieving the independent components, is the distance between the overall transforming matrix **Q** and an ideal permutation matrix, which is interpreted as the *cross-talking error* [45]:

$$CTE = \sum_{i=1}^{N} \left(\sum_{j=1}^{N} \frac{|Q_{ij}|}{max|Q_i|} - 1 \right) + \sum_{j=1}^{N} \left(\sum_{i=1}^{N} \frac{|Q_{ij}|}{max|Q_j|} - 1 \right) \tag{42}$$

Above, Q_{ij} is the *ij*-th element of **Q**, $max|Q_i|$ is the maximum absolute valued element of the row *i* in **Q**, and $max|Q_j|$ is the maximum absolute valued element of the column *j* in **Q**. A permutation matrix is defined so that on each of its rows and columns, only one of the elements equals to unity

while all the other elements are zero. It means that *CTE* attains its minimum value zero for an exact permutation matrix.

6.3 RANKING THE ESTIMATES

Ranking the estimated independent components was another criterion used for assessing the reliability of ICA algorithms. Friedman [46] proposed a robust *structural* measure to arrange the ICA basis vectors. The idea is to first sphere the data and then to map them into the interval $[0,1]$ with the gaussian cumulative density function $\Phi(v)$. For T realizations of a (scalar) random variable $y_k(t)$ the proposed scheme leads to the index for the *k*-th estimated independent component:

$$E_1(y_k) = \sum_{t=1}^{T} \left[y_{s_t} - \left(\frac{2t}{T} - 1 \right) \right]^2 \tag{43}$$

where $\{s_t\}$ are the indexes of the ordered $\{y_k(t)\}$ in such a way that $y_k(i) \le y_k(j)$ iff $s_i \le s_j$. The higher $E_1(y_k)$ is, the more structural information contains the *k*-th estimated independent component

It is meaningful and useful to rank order the components by the extent of their contribution to the original data. The *contribution* of the estimated component $y_k(t)$ may be expressed in two alternative ways [47]. It can be estimated by the root mean square (RMS) of the data set reconstructed solely from this component $\mathbf{x} = \hat{\mathbf{A}}\mathbf{y}$ in which **y** has only one nonzero row corresponding to the appropriate component. Alternatively, it can be regarded as the RMS error introduced per data point when the data **x** are reconstructed without this component (e.g. **y** has only one zero row corresponding to the appropriate component):

$$E_2(y_k) = \frac{1}{T \cdot N} \left[\sum_{j=1}^{N} \sum_{t=1}^{T} \left(C_{jt}^k \right)^2 \right]^{1/2} \tag{44}$$

where C_{jt}^k is the element of an $N \times T$ matrix computed from the outer product of the *k*-th independent component and the *k*-th column of $\hat{\mathbf{A}}$, that is $C_{jt}^k = B_{jk}^{-1} Y_{kt}$. The higher $E_2(y_k)$ is, the higher the contribution of the component $y_k(t)$ to the observed data.

7. Results and Discussion

In our simulations, we used 6 different artificially generated time series of 512 samples each, both subgaussian and supergaussian (Fig. 3). Their analytical expressions in Tables 1 and 2 are in MATLAB language.

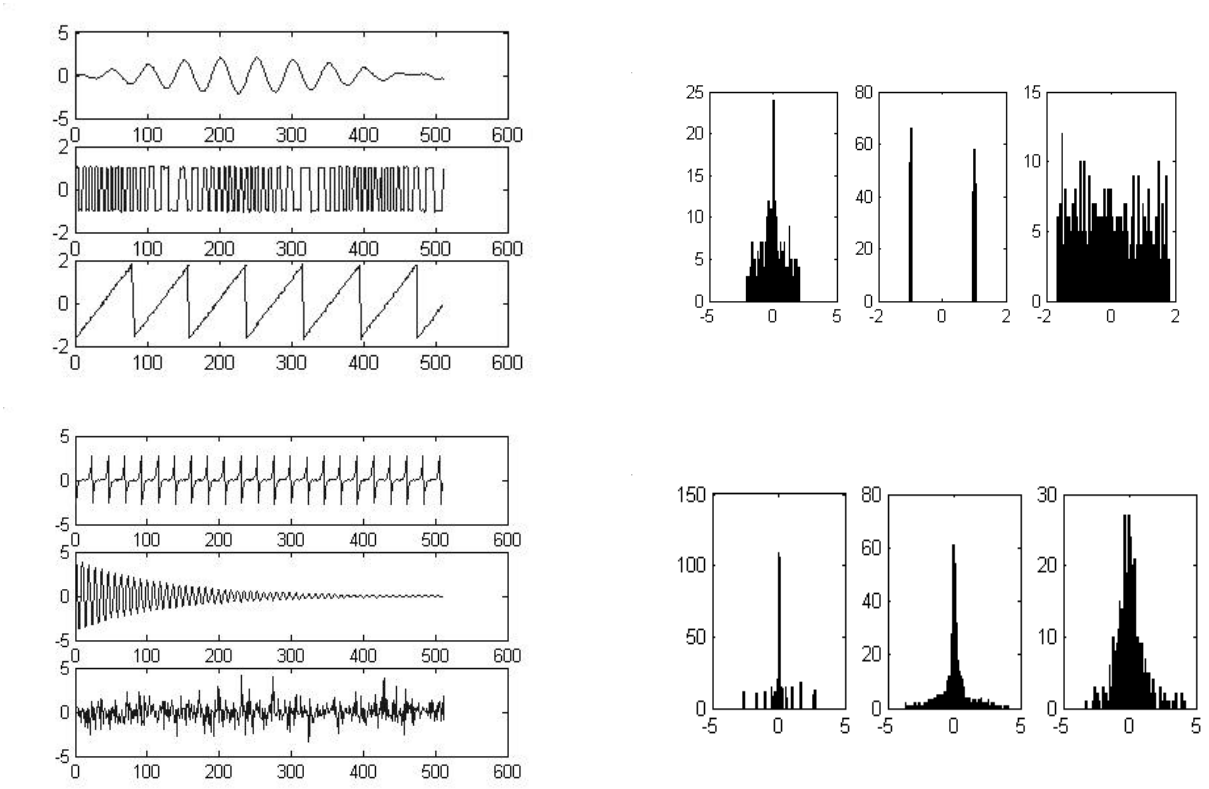

Figure 3. Artificially generated signals (left) and their histograms (right). Both subgaussian (up) and supergaussian (bottom) signal distributions were considered as displayed by their histograms and characterized by their kurtosis.

Table 1. The analytical form and the *3*-rd and *4*-th order cumulants of the subgaussian sources.

Source signal	*Skewness*	*Kurtosis*
Modulated sinusoid: $S(1)=2*sin(t/149)*cos(t/8)$	*0.024637*	*−0.551312*
Square waves: $S(2)=sign(sin(12*t+9*cos(2/29)))$	*0.015638*	*−1.996568*
Saw-tooth: $S(3)=(rem(t,79)-17)/23$	*0.101021*	*−1.191073*

Table 2. The analytical form and the *3*-rd and *4*-th order cumulants of the supergaussian sources.

Source signal	*Skewness*	*Kurtosis*
Impulsive curve: $S(4)=((rem(t,23)-11)/9)^5$	*−0.011980*	*2.353211*
Exponential decay: $S(5)=5*exp(-t/121)*cos(37*t)$	*0.055131*	*3.410776*
Spiky noise: $S(6)=((rand(1,T)<.5)*2-1)*log(rand(1,T))$	*0.464295*	*2.228476*

The signals were mixed by a fixed square matrix **A**, with randomly generated elements within the range $[-1,1]$ for proper scaling, and having $det(\mathbf{A}) > 1$ for nonsingularity:

$$\mathbf{A} = \begin{bmatrix} 0.2658 & -0.2549 & 0.8715 & 0.3988 & 0.4422 & -0.0659 \\ -0.4558 & -0.7068 & -0.6365 & 0.7509 & 0.0538 & 0.7672 \\ 0.6531 & -0.8278 & -0.5370 & -0.8373 & -0.1628 & -0.5906 \\ 0.5700 & -0.6150 & -0.6972 & 0.1253 & -0.6943 & 0.7593 \\ -0.9669 & -0.5057 & -0.2333 & -0.3829 & 0.3270 & -0.6620 \\ 0.6294 & -0.8700 & 0.4146 & 0.8457 & -0.1134 & 0.0999 \end{bmatrix}$$

The algorithms used in our simulations were partly downloaded directly from their original sites (Table 3), some of which were modified, and partly implemented in MATLAB 6.0. All codes for generated data and statistical processing were parts or modified versions of MATLAB codes.

Table 3. The origin of code sources for the neurally implemented ICA algorithms.

Algorithms	*Type*	*Source*
FastICA	original	http://www.cis.hut.fi/projects/ica/fastica/
BS	modified	http://www.cnl.salk.edu/~tewon/ica_cnl.html
ACY	personal	as described in [12]
EASI	modified	http://sig.enst.fr/~cardoso/guidesepsou.html
Pearson-ICA	modified	http://wooster.hut.fi/statsp/papers/Pearson_ICA.zip
EGLD-ICA	modified	http://wooster.hut.fi/statsp/papers/EGLD_ICA.zip

The simulations were carried out in two steps. First, we used *SIR*, *CTE*, and the sum of individual negentropies of the estimated components for each algorithm under test in order to determine its optimal performing nonlinearity and/or contrast function. For the sum of the individual output negentropies we listed both J_a and J_b values corresponding to the different expressions H_a and H_b which approximated the entropy. The results were averaged over *120* full sessions with *5* nonlinearities per session and per algorithm. The best choices are presented in Table 4.

Once the best performing contrast function was determined for each algorithm [48], the separation accuracy of the source components was comparatively measured using *SIR* and *CTE*, whereas the components ranking was carried out on the basis of the indexes E_1 and E_2 among all

algorithms. The results presented in Table 5 are the means and standard deviations corresponding to *500* full runs. The hierarchy corresponds to the decreasing order of overall performance.

Table 4. The optimal performing nonliniarity for each algorithm under study.

Algorithms	*Nonlinearities and score functions*	*Sum of negentropies*	
		J_a	J_b
FastICA	$g(y) = y\,exp\left(-y^2/2\right)$	1.67 ± 0.028	1.25 ± 0.012
BS-Infomax	$g(y) = y \pm tanh(y)$	1.0 ± 0.23	0.8 ± 0.35
ACY	$g(y) = y - tanh(y)$	0.8 ± 0.50	0.6 ± 0.19
EASI	$g(y) = -tanh(y)$	0.5 ± 0.48	0.4 ± 0.35
Pearson-ICA	$j(y) = -\frac{y-a}{b_0 + b_1 y + b_2 y}$	0.96 ± 0.075	0.8 ± 0.66
EGLD-ICA	$F^{-1}(p) = l_1 + \frac{p^{l_3} - (1-p)^{l_4}}{l_2}$	1.02 ± 0.095	0.89 ± 0.23

Table 5. The assessment of ICA algorithms and ranking of the estimated components. Both the mean values and the standard deviations are indicated for the indexes *SIR* and *CTE*. The indexing of the components corresponds to their counterparts in the input sequence.

Algorithms	*SIR* [dB]	*CTE*	E_1	E_2
FastICA	17.1 ± 3.51	0.65 ± 0.12	$y_6, y_3, y_5, y_2, y_1, y_4$	$y_3, y_5, y_2, y_1, y_4, y_6$
BS	12.2 ± 4.72	1.12 ± 0.33	$y_6, y_3, y_5, y_2, y_1, y_4$	$y_3, y_5, y_2, y_1, y_4, y_6$
ACY	13.5 ± 3.86	0.90 ± 0.41	$y_6, y_3, y_5, y_2, y_1, y_4$	$y_3, y_5, y_2, y_1, y_4, y_6$
EASI	7.02 ± 5.30	2.30 ± 1.39	$y_6, y_3, y_5, y_2, y_4, y_1$	$y_3, y_5, y_2, y_1, y_4, y_6$
Pearson-ICA	7.93 ± 2.88	2.14 ± 1.74	$y_6, y_3, y_5, y_2, y_4, y_1$	$y_3, y_5, y_2, y_1, y_4, y_6$
EGLD-ICA	8.21 ± 3.75	2.10 ± 1.56	$y_6, y_3, y_5, y_2, y_1, y_4$	$y_3, y_5, y_2, y_1, y_4, y_6$

6 Conclusions

In recent years, new learning algorithms for solving the ICA problem have been proposed, yet their theoretical properties, range of optimal applicability, and comparative assessment are still largely unexplored. We limited our study to the most popular algorithms for the time being.

To date the best ICA algorithm in terms of convergence, computational requirements, and parameters to be tuned is the FP FastICA with *symmetric* orthogonalization and *exponential* nonlinearity. Its stabilized version converges always to a definite subspace of meaningful components even if the statistical independence is weak. Though the BS and ACY algorithms are optimal in terms of mutual information, their computational cost is higher whereas the results are similar with the FP. Moreover, like all neural unsupervised algorithms, both BS and ACY algorithms are heavily dependent on the learning rates and their convergence is quite slow. The Pearson and EGLD algorithms employing the ML principle separate a relative wide class of nongaussian source signals of large interest, even skewed distributions with zero kurtosis. However, Pearson system's ability is to model distributions that are close to normal distribution constrains its applications since it has no particular advantages for modeling distributions far from normality. As both estimators for parameters and score function are simple rational functions both Pearson-ICA and EGLD algorithms are computationally fast. However, the error margins are sensibly larger than in the case of FP, BS and ACY algorithms.

ICA as a means for performing BSS is a rapidly emerging new application domain of unsupervised neural learning. Theoretically, ICA should be useful in quite all applications as standard PCA and even beyond. First, observe that the assumption of mutual independence between sources is a statistically strong hypothesis but very plausible in real world, particularly in the case of physically separated sources. This makes any ICA performing algorithm extremely attractive for data analysis. Then, the model and the neural algorithms analyzed are referring to stationary linear noiseless ICA model. Challenging directions of further research are at least oriented towards modeling the effect of noise, nonstationarity, and nonlinearity in ICA.

ACKNOWLEDGMENTS

R.M. is supported by a postdoc grant from the European Community, FP5 (QLG3-CT-2000-30161). M.M.V.H. is supported by research grants received from the Fund for Scientific Research (G.0185.96N), the National Lottery (Belgium) (9.0185.96), the Flemish Regional Ministry of Education (Belgium) (GOA 95/99-06; 2000/11), the Flemish Ministry for Science and Technology (VIS/98/012), and the European Community, FP5 (QLG3-CT-2000-30161 and IST-2001-32114).

References

1. Comon, P. (1994) Independent component analysis, A new concept? *Signal Proces*. 36:287-314.
2. Cardoso, J.-F. (1998) Blind signal separation: Statistical principles. *Proc. IEEE* 9(10):2009-2025.
3. Karhunen, J., Cichocki, A., Kasprzak, W., Pajunen, P. (1997) On neural blind separation with noise suppression and redundancy reduction. *Int. Journal of Neural Systems* 8:219-237.
4. Karhunen, J., Oja, E., Wang, L., Joutsensalo, J. (1995) Neural estimation of basis vectors in independent component analysis. In: *Proc. int. Conf. on artificial neural networks (ICANN'95)*. Paris, France, October, pp. 317-322.
5. Karhunen,. J, Pajunen, P. (1997) Blind Source Separation Using Least-Squares Type Adaptive Algorithms. In: *Proc. IEEE 1997 Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'97)*, Munich, Germany, April, pp. 3361-3364.
6. Karhunen, J. (1996) Neural Approaches to Independent Component Analysis and Source Separation. In: *Proc. 4th European Symposium on Artificial Neural Networks (ESANN'96)*, Bruges, Belgium, April, pp. 249-266 (invited paper).
7. Amari, S., Cichocki, A., Yang, H. (1995) Recurrent neural networks for blind separation of sources. In: *Proc. 1995 Int. Symp. on Nonlinear Theory and Applications (NOLTA-95)*. Las Vegas, USA, 1:37-42.
8. Juten, C., Hérault, J. (1991) Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture. *Signal Proces.* 24(1):1-10.
9. Matsuoka, K., Ohya, M., Kawamoto, M. (1995) A neural net for blind separation of nonstationary signals. *Neural Networks* 8(3):411-419.
10. Cichocki A, Kasprzak W, Amari S (1995) Multi-layer neural networks with a local adaptive learning rule for blind separation of source signals. In: *Proc. 1995 Int. Symp. on Nonlinear Theory and Applications (NOLTA-95)*. Las Vegas, USA 1:61-66.
11. Bell, A.J., Sejnowski, T.J. (1995) An information maximization-approach to blind separation and blind deconvolution. *Neural Comput.* 7:1129-1159.
12. Moreau, O., Macchi, O. (1993) New self-adaptive algorithms for source separation based on contrast functions. In: *Proc. IEEE Signal Processing Workshop on Higher Order Statistics.* Lake Tahoe, NV, pp. 215-219.
13. Delfosse, N., Loubaton, P. (1995) Adaptive blind separation of independent sources: A deflation approach. *Signal Proces.* 45:59-83.

14. Oja, E., Karhunen, J. (1995) Signal separation by nonlinear Hebbian learning. In: Palaniswami, M., Attikiouzel, Y., Marks, R., Fogel, D., Fukuda, T. (eds.) Computational Intelligence – A Dynamic System Perspective. IEEE Press, New York, pp. 83-97.
15. Amari, S., Cichocki, A., Yang, H. (1996) A new learning algorithm for blind source separation. In: Advances in Neural Information Processing 8 (*Proc. NIPS'95*). MIT Press, Cambridge, MA.
16. Hyvärinen, A., Oja, E. (1996) A neuron that learns to separate one independent component from linear mixtures. In: Proc. IEEE Int. Conf. on Neural Networks. Washington, DC, pp. 62-67.
17. Cardoso, J.-F., Belouchrani, A., Lahed, B. (1994) A new composite criterion for adaptive and iterative blind source separation. In: *Proc. ICASSP*, 4:273-276.
18. Hyvärinen, A., Oja, E. (1997) A fast fixed-point algorithm for independent component analysis. *Neural Comput.* 9:1483-1492.
19. Cardoso, J.-F. (1992) Iterative techniques for blind source separation using only fourth-order cumulants. In: *Proc. EUSIPCO*, Brussels, pp. 739-742.
20. Gaeta, M., Lacoume, J.-L. (1990) Source separation without a priori knowledge: the maximum likelihood solution. In: *Proc. EUSIPCO*, pp. 621-624.
21. Comon, P. (1991) Independent component analysis. In: *Proc. Int. Workshop on Higher-Order Stat.* Chamrousse, France, pp. 111-120.
22. Giannakis, G., Shamsunder, S. (1991) Modeling of nongaussian array data using cumulants. DOA estimation with less sensors than sources. In : *Proc. Conf. Inf. Sci. Syst,*. Baltimore.
23. Cardoso, J.-F., Souloumiac, A. (1993) Blind beamforming for nongaussian signals. *IEE Proc.-F*, 140(6):362-370.
24. Pham, D.-T., Garrat, P., Jutten, C. (1992) Separation of a mixture of independent sources through a maximum likelihood approach. In: *Proc. EUSIPCO*, pp. 771-774.
25. Tong, L., Liu, R. (1990) Blind estimation of correlated source signals. In: *Proc. Asilomar. Conf*., November.
26. Abed Meraim, K., Belouchrani, Cardoso J.-F., Moulines, E. (1994) Asymptotic performance of second order blind source separation. In: *Proc. ICASSP*., 4:277-280.
27. Nguyen, H.-L., Jutten, C. (1995) Blind source separation for convolutive mixtures. *Signal Proces*. 45:209-229.
28. Laheld, B., Cardoso, J.-F. (1994) Adaptive source separation with uniform performance In *Proc. EUSIPCO*, Edinburgh, September, pp. 183-186.

29. Cardoso, J.-F., Laheld, B.H. (1996) Equivariant adaptive source separation. *IEEE T. Signal Proces*. 44(12):3017-3030.
30. Wang, L., Karhunen, J., Oja, E. (1995) A bigradient optimization approach for robust PCA, MCA, and source separation. In: *Proc. 1995 IEEE Int. Conf. on Neural Networks*, Perth, Australia, November, pp. 1684-1689.
31. Hyvärinen, A. (1998) New approximations of differential entropy for ICA and projection pursuit. In: *Advances in Neural Information Processing Systems (NIPS'97)*. MIT Press 10:273-279.
32. Lee, T.-W., Girolami, M., Bell, A.J., Sejnowski, T.J. (2000) A unifying information-theoretic framework for Independent Component Analysis, *Computers and Mathematics with Applications* 39:1-21.
33. Karvanen, J., Eriksson, J., Koivunen, V. (2000) Pearson System Based Method for Blind Separation. In: *Proc. of 2nd International Workshop on Independent Component Analysis and Blind Signal Separation*, Helsinki, pp. 585-590
34. Eriksson, J., Karvanen, J., Koivunen, V. (2000) Source Distribution Adaptive Maximum Likelihood Estimation of ICA Model. *Proc*. of *2nd International Workshop on Independent Component Analysis and Blind Signal Separation*, Helsinki, pp.227-232
35. Stuart, A., Ord, J.K. (1994) Kendall's Advanced Theory of Statistics: Distribution Theory. 6th edition, Vol. 1, Edward Arnold.
36. Amari, S.-I. (1998) Natural gradient works efficiently in learning. *Neural Comput.* 10:251-276
37. Tukey, J.W. (1960): The practical relationship between the common transformations of percentages of counts and amounts, *Technical Report 36*, Statistical Techniques Research Group, Princeton University.
38. Ramberg, J.S., Dudewicz, E.J., Tadikamalla, P.R., Mykytka, E.F. (1979) A probability distribution and its uses in fitting data. *Technometrics* 21:201-204.
39. Karian, Z.A., Dudewicz, E.J., McDonald, P. (1996) The extended generalized lambda distribution system for fitting distributions to data: history, completion of theory, tables, applications, the "final word" on moment fits. *Comm Stat.: Simulation and Computation* 25(3):611-642.
40. Karian, Z.A., Dudewicz, E.J. (1996) The extended generalized lambda distribution (EGLD) system for fitting distributions to data with moments, II: Tables. *American Journal of Mathematical and Management Sciences*.

41. Karhunen, J., Wang, L., Joutsensalo, J. (1995) Neural estimation of basis vectors in independent component analysis. In: *Proc. int. Conf. on artificial neural networks (ICANN'95)*. Paris, France, October, pp. 317-322.
42. Haykin S (1994) Neural Networks. A Comprehensive Foundation. MacMillan College Publishing, New York.
43. Papoulis A (1991) Probability, *Random Variables, and Stochastic Processes*. 3rd edition, McGraw-Hill.
44. Hyvärinen, A. (1999) Survey on independent component analysis. *Neural Computing Surveys* 2:94-128.
45. Yang, H., Amari S.-I. (1997) Adaptive online learning algorithms for blind separation: maximum entropy and minimum mutual information. *Neural Comput.* 9:1457-1482.
46. Friedman, J.H. (1987) Exploratory projection pursuit. *J. of the American Statistical Association* 82(397):249-266.
47. McKeown, M.J., Makeig, S., Brown, G.G., Jung, T.-P., Kindermann, S.S., Bell, A.J., Sejnowski, T.J. (1998) Analysis of fMRI data by blind separation into independent spatial components. *Human Brain Map*. 6:160-188.
48. Mutihac, R., Van Hulle, M.M. (2002) Assessment of contrast function performance for ICA fixed-point algorithms. Submitted to *European Meeting on Independent Component Analysis*, Vetri Sul Mare, Italy, 2002.