Journal of Machine Learning Research 4 (2003) 1297-1318 Submitted 10/02; Published 12/03

MISEP – Linear and Nonlinear ICA Based on Mutual Information

Luís B. Almeida LUIS.ALMEIDA@INESC-ID.PT
INESC-ID, R. Alves Redol, 9,
1000-029 Lisboa, Portugal
Phone: +351-213100246
Fax: +351-213145843

Editors: Te-Won Lee, Jean-François Cardoso, Erkki Oja and Shun-Ichi Amari

Abstract

Linear Independent Components Analysis (ICA) has become an important signal processing and data analysis technique, the typical application being blind source separation in a wide range of signals, such as biomedical, acoustical and astrophysical ones. Nonlinear ICA is less developed, but has the potential to become at least as powerful.

This paper presents MISEP, an ICA technique for linear and nonlinear mixtures, which is based on the minimization of the mutual information of the estimated components. MISEP is a generalization of the popular INFOMAX technique, which is extended in two ways: (1) to deal with nonlinear mixtures, and (2) to be able to adapt to the actual statistical distributions of the sources, by dynamically estimating the nonlinearities to be used at the outputs. The resulting MISEP method optimizes a network with a specialized architecture, with a single objective function: the output entropy.

The paper also briefly discusses the issue of nonlinear source separation. Examples of linear and nonlinear source separation performed by MISEP are presented.

Keywords: ICA, Blind Source Separation, Nonlinear ICA, Mutual Information

1. Introduction

Linear Independent Components Analysis (ICA) and linear Blind Source Separation (BSS) have become, in the last years, relatively well established signal processing and data analysis techniques (good overviews can be found in work by Lee et al., 1998; Hyvarinen et al., 2001). Nonlinear ICA and nonlinear BSS, on the other hand, are techniques that are still largely under development, and have the potential to become rather powerful tools. Some work on nonlinear ICA has already been published (Burel, 1992; Deco and Brauer, 1995; Marques and Almeida, 1996; Yang et al., 1998; Marques and Almeida, 1999; Palmieri et al., 1999; Valpola, 2000; Almeida, 2000a; Harmeling et al., 2001; Martinez and Bray, 2003).

In this paper we consider ICA as the problem of transforming a set of patterns $\mathbf{o}$ (vectors of size n, often called *observations*), whose components are not statistically independent from one another, into patterns $\mathbf{y} = \mathbf{F}(\mathbf{o})$ whose components are as independent from one another as possible. In linear ICA the transformation $\mathbf{F}$ is restricted to be linear, while in nonlinear ICA there is no such restriction. In blind source separation one further assumes that the observations are the result of a mixture of statistically independent *sources*, s_i, i.e. $\mathbf{o} = \mathbf{M}(\mathbf{s})$, s_i being the components of $\mathbf{s}$. The purpose of BSS is the recovery of the sources from the observations, and ICA is one of the most commonly used techniques for performing this recovery. Once again, one distinguishes linear

BSS, in which the mixture $\mathbf{M}$ is assumed to be linear, and nonlinear BSS, where there is no such assumption. In this paper we deal with linear and nonlinear ICA in the so called *square* case, in which the numbers of components of $\mathbf{s}$, $\mathbf{o}$ and $\mathbf{y}$ are assumed to be the same.

An important ingredient of most ICA methods, both linear and nonlinear, is a measure of the mutual dependence of the extracted components y_i. This measure is sometimes called *contrast function* (Comon, 1994). Many ICA methods are based on the minimization of such a measure. Linear ICA is a relatively constrained problem, and therefore linear ICA methods do not need to be based on strict dependence measures. For example, some of these methods, which give rather good results in appropriate situations, are based only on cumulants up to the fourth order (Cardoso and Souloumiac, 1996; Hyvärinen and Oja, 1997). Nonlinear ICA, on the other hand, is rather unconstrained, and normally demands a good dependence measure. Some of the dependence measures that have been proposed are based on a quadratic "error" between probability densities (Burel, 1992), on moments of all orders (Marques and Almeida, 1996), on Renyi's entropy (Marques and Almeida, 1999) or on the mutual information of the estimated components (e.g. Deco and Brauer, 1995; Taleb and Jutten, 1997; Almeida, 2000a,b). The latter, the mutual information (MI) of the estimated components, is rather appealing as a dependence measure for several reasons. First of all, it is a strict dependence measure: it is always non-negative, and is zero only if the estimated components are statistically independent. We shall outline two other reasons for its appeal ahead.

The mutual information of the components of the vector $\mathbf{y}$ is defined as

$$I(\mathbf{y}) = \sum_i H(y_i) - H(\mathbf{y})$$

where H denotes Shannon's entropy, for discrete variables, or Shannon's differential entropy, $H(\mathbf{y}) = -\int p(\mathbf{y}) \log p(\mathbf{y}) d\mathbf{y}$, for continuous variables, and $p(\mathbf{y})$ denotes the joint probability density of the components of $\mathbf{y}$.[1] This measure has the appealing property of being based on Shannon's entropy, which is the most meaningful entropy measure in a large variety of situations. It also has the property of being insensitive to invertible transformations of the components. More specifically, if we define $z_i = \psi_i(y_i)$, where the ψ_i are invertible, then $I(\mathbf{z}) = I(\mathbf{y})$. This property is intuitively sound, and is of great use in the derivation of algorithms, such as MISEP, based on the minimization of the mutual information, as we shall see ahead. Mutual information has been used as a criterion for ICA in several different ways (for examples of its use in linear ICA see work by Amari et al. 1996; Haykin and Gupta 1999; Almeida 2000b; Taleb and Jutten 1997; for nonlinear ICA examples see work by Yang et al. 1998; Deco and Brauer 1995; Almeida 2000a). This paper's central topic is the method of Almeida (2000a,b).

The use of mutual information as an ICA criterion raises difficulties, that have been circumvented by different authors in different ways. From (1) we see that the computation of the mutual information requires the knowledge of both the joint and the marginal distributions of the estimated sources. In practical situations, however, we usually have access only to a finite set of mixture patterns $\mathbf{o}$ (the training set), from which we can obtain a finite set of vectors of extracted components, $\mathbf{y} = \mathbf{F}(\mathbf{o})$, given some candidate transformation $\mathbf{F}$. The joint and marginal distributions of the components of $\mathbf{y}$ have to be estimated from this finite set.

The need to estimate the joint density $p(\mathbf{y})$ can be circumvented without resorting to approximations, as described ahead. On the other hand, there is no known way of circumventing the need

1. We shall use the same notation, $p()$, to denote the statistical densities of all the random variables dealt with in this paper. The argument used in the function will clarify which random variable is being considered. While this is a slight abuse of notation, it will help to keep expressions simpler and will not originate confusions.

to estimate the marginal densities $p(y_i)$, or some equivalent description of the marginal distributions. One of the main differences among the various MI-based ICA methods is the way in which this estimation is dealt with. For example, Amari et al. (1996); Haykin and Gupta (1999); Deco and Brauer (1995) use truncated series expansions of the densities, estimated from the **y** patterns. The well known INFOMAX method (Bell and Sejnowski, 1995), although originally based on a different reasoning, can be interpreted as assuming some given, a-priori marginal distributions for the y_i, as we shall see ahead. A first extension to INFOMAX (Lee et al., 1999) makes a binary decision on the form of each of these distributions. A more general extension (Taleb and Jutten, 1997) estimates the *score functions* (which can be seen as alternate descriptions of the marginal distributions) by means of multilayer perceptrons, using as optimization criterion the quadratic error between the true and estimated score functions. MISEP (Almeida, 2000a,b), described in this paper, is also based on INFOMAX, but estimates the marginal distributions in a different way, based on a maximum entropy criterion. It has the advantages that (1) both the independent component analysis itself and the estimation of the marginal distributions are performed by the same network, optimized according to a single criterion, and (2) that it is not limited to linear ICA, but can deal with nonlinear mixtures as well.

There is an important difference between linear and nonlinear ICA that we should emphasize before proceeding. Under rather unrestrictive assumptions, linear ICA has essentially a single solution, except for possible permutations and scalings of the components (Comon, 1994)). This makes ICA one of the most important tools for performing linear blind source separation, since it essentially gives a guaranty of recovering the original sources. In the nonlinear case, however, it can be easily shown that ICA has an infinite number of solutions that are not related in any simple way to one another (Darmois, 1953; Hyvarinen and Pajunen, 1999; Marques and Almeida, 1999). In a nonlinear BSS problem, an ICA technique, if used alone, can't give any guaranty of recovering the original sources. This has led some people to think that nonlinear source separation was unsolvable, or at least that it couldn't be solved by means of ICA techniques. This is a wrong view. What we have said means that nonlinear BSS is an ill-posed problem. But many other ill-posed problems exist, with which we deal with relative ease. For example, probability density estimation, the training of classifiers or the estimation of nonlinear regressors are ill-posed problems that we normally don't consider unsolvable. The solution to the ill-posedness is of the same kind in all cases: further knowledge has to be used. Fortunately, this knowledge often exists in practical situations. Frequently, this knowledge takes the form of some regularity assumption about the solution, and is applied to the problem through a suitable form of regularization. The same applies here, and we shall see in this paper several examples of nonlinear source separation performed through ICA.

A complementary note concerns the kinds of indeterminacies that we should expect in nonlinear blind source separation, even if proper regularization is used. In linear BSS these indeterminacies are just permutation and scaling. Here the permutation ambiguity remains, but we should expect the scaling ambiguity, which has the form $y_i = a_i s_i$ with arbitrary scale factors a_i, to be extended to an ambiguity of the form $y_i = f_i(s_i)$, with arbitrary invertible nonlinear functions f_i. In fact, if the sources s_i are independent, so are $f_i(s_i)$, and an ICA-based criterion can't distinguish among them. Prior information may, or may not, help us to avoid these nonlinear transformations of the sources, depending on the specific situation.

The organization of this paper is as follows: Section 2 derives the MISEP method, by extending INFOMAX in the two directions indicated above. Results of linear and nonlinear ICA and BSS are presented in Section 3. Section 4 briefly discusses the issue of separability of nonlinear mixtures, and Section 5 presents the paper's conclusions.

2. The MISEP Method

In this section we start by briefly reviewing INFOMAX, and then proceed to examine the MISEP method, both in its theoretical basis and in its implementation.

2.1 INFOMAX – Brief Review

INFOMAX was originally presented as a maximum information preservation method, but can also be seen as a maximum likelihood one (Cardoso, 1997) or as an MI-based one. It is this MI-based interpretation that interests us in this paper.

In Figure 1 we show the form of the network that is used by INFOMAX. The separation function $\mathbf{F}$, being linear, performs just a product by a matrix. The ψ_i blocks are auxiliary, being used only during training. Each of them outputs a nonlinear, increasing function of its input, with values in $[0,1]$, i.e. $z_i = \psi_i(y_i)$ with $z_i \in [0,1]$. The system is trained by maximizing the output entropy $H(\mathbf{z})$.

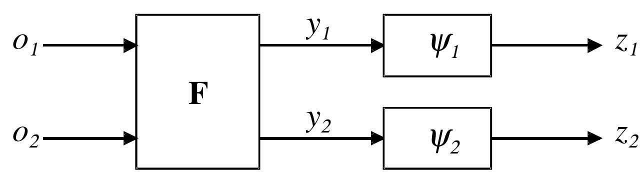

Figure 1: Structure of the ICA systems studied in this paper. In the INFOMAX method the nonlinearities ψ_i are fixed a-priori. In the MISEP method they are adaptive, being implemented by multilayer perceptrons. The figure illustrates the two-component case, but extension to a larger number of components is straightforward.

Since each z_i is related to the corresponding y_i by an invertible transformation, we have $I(\mathbf{y}) = I(\mathbf{z})$. Assume now that we choose for each nonlinearity ψ_i the cumulative probability function (CPF) of the corresponding component y_i. Then z_i will have a uniform distribution in $[0,1]$ and $H(z_i) = 0$. Consequently,

$$\begin{aligned} I(\mathbf{y}) &= I(\mathbf{z}) \\ &= \sum_i H(z_i) - H(\mathbf{z}) \\ &= -H(\mathbf{z}), \end{aligned} \tag{1}$$

Maximization of the output entropy $H(\mathbf{z})$ will therefore be equivalent to the minimization of $I(\mathbf{y})$, the mutual information of the estimated components. INFOMAX can therefore be viewed as minimizing this mutual information, wtih an a-priori choice of the estimated distributions of the components, performed through the choice of the ψ nonlinearities. These should approximate the CPFs of the actual components as closely as possible. However, as said above, linear ICA is a rather constrained problem, and therefore INFOMAX usually performs well even if the output nonlinearities are only crude approximations to these cumulative functions. For example, it is known that logistic sigmoids can be used as nonlinearities for most unskewed, supergaussian distributions (Bell and Sejnowski, 1995).

2.2 MISEP – Theoretical Basis

MISEP uses the same basic network structure as INFOMAX (Figure 1). But since it is also to be applicable to nonlinear mixtures, the separating block **F** shall now be nonlinear, with the capability to implement a relatively wide class of functions. We have often used a multilayer perceptron (MLP) to implement this block, but in some cases we've used a radial basis function network instead.

MISEP should be able to deal with a wide class of statistical distributions of the y_i components. On the other hand, it needs to have good estimates of their CPFs, to be able to perform nonlinear ICA, which is much less constrained than its linear counterpart. We have therefore implemented the ψ nonlinearities by means of MLPs, which adaptively learn the CPFs during the training (again, other kinds of nonlinear blocks could have been used).

The **F** and ψ blocks, taken together, form a nonlinear network with a specialized architecture. The purposes of the training of the two kinds of blocks are very different: We want the **F** block to yield components that are as independent as possible, i.e. to minimize $I(\mathbf{y})$, while each ψ block should approximate the CPF of its input as closely as possible.

We have already seen, in our analysis of INFOMAX, that the minimization of $I(\mathbf{y})$ can be translated into the maximization of the network's output entropy. A key idea in MISEP is understanding that this same criterion will lead the output nonlinearities to approximate the desired CPFs. This is due to the fact that maximizing the output entropy will tend to lead the distribution of each z_i to be uniform in $[0,1]$, since the uniform distribution is the one which has maximum entropy in a finite interval. More specifically, from (1) we can write

$$\sum_i H(z_i) = H(\mathbf{z}) + I(\mathbf{y}).$$

If we assume, for the moment, that the distributions of the y_i are kept fixed, we see that maximizing $H(\mathbf{z})$ will lead to the maximization of each of the marginal entropies $H(z_i)$, since each of them depends on a separate set of parameters (because the ψ_i networks are separate from one another). Maximizing $H(z_i)$ will lead the distribution of z_i to approach the uniform distribution in $[0,1]$, as said above, and will lead ψ_i to approach the CPF of y_i, as desired, if ψ_i is constrained to be an increasing function with values in $[0,1]$ (we shall discuss later how to implement this constraint).

During a training procedure, the distributions of the y_i will not remain fixed. One might wonder whether this would invalidate the reasoning given above. Note, however, that (1) the whole network will be trained by maximization of a single objective function (the output entropy), and therefore there is no danger of instability in the training, assuming that a well designed optimization procedure is used, and (2) when the training procedure approaches a maximum of the entropy and slows down, the statistics of the y_i will change very slowly, and the reasoning above will be valid. Therefore, at convergence, the ψ_i functions will be estimates of the CPFs of the components y_i.

2.3 Implementation

We'll start by discussing how to implement the constraints on the ψ functions, and shall then describe how to train the whole network using the output entropy as objective function.

2.3.1 CONSTRAINING THE ψ MLPS

The MLPs that implement the ψ functions have to be constrained to yield increasing functions with values in $[0,1]$. There are several possibilities for doing this. Here we shall only describe the one that we have found to be most effective (for other possibilities, and for a discussion of their drawbacks, see work by Almeida, 2000a,b, 2001, 2002). To implement these constraints we use, in the ψ MLPs, hidden units with sigmoids which are increasing, with values in $[0,1]$, and we use linear units at the outputs. We normalize the Euclidean norm of the vector of weights leading into each output unit to $1/\sqrt{h}$, h being the number of hidden units connected to that output unit. With non-negative weights, this guarantees that the outputs will be in $[0,1]$. If we use non-negative weights throughout these networks, they will also be guaranteed to yield non-decreasing functions. In practice we have found that instead of strictly enforcing non-negativity of the weights, it is preferable to enforce it in a soft way: we initialize all weights with positive values, and the training procedure by itself tends to keep them all positive, because a negative weight, among positive ones, would decrease the output entropy. We have occasionally encountered negative weights during the training, but these normally revert to positive values by themselves in a few iterations.

In actual implementations we have used, in the hidden layer, sigmoids with values in $[-1,1]$. This yields ψ functions with values in $[-1,1]$, which are estimates of the CPFs re-scaled to this interval. This still performs minimization of $I(\mathbf{y})$, as can easily be checked. The use of these sigmoids has the advantage of resulting in a faster training.

2.3.2 MAXIMUM ENTROPY TRAINING

The whole network of Figure 1 is to be trained through maximization of the output entropy. This is the same criterion that is used in INFOMAX, and the first steps in the derivation of our training procedure closely follow those of INFOMAX. We use gradient-based optimization. The output entropy can be written as

$$H(\mathbf{z}) = H(\mathbf{o}) + \langle \log |\det \mathbf{J}| \rangle$$

where $\mathbf{J} = \partial \mathbf{z}/\partial \mathbf{o}$ is the Jacobian of the transformation performed by the network, and the angle brackets denote expectation. The term $H(\mathbf{o})$ doesn't depend on the network's parameters, and can be omitted from the optimization. The remaining term, which is a statistical mean, will be approximated by the empirical mean, i.e. by the mean computed on the training set,

$$\langle \log |\det \mathbf{J}| \rangle \approx \frac{1}{K} \sum_{k=1}^{K} \log \left|\det \mathbf{J}^k\right| = E,$$

where $\mathbf{J}^k$ denotes the value of $\mathbf{J}$ for the k-th training pattern, and K is the number of training patterns. E will be our objective function.

Here we have to depart from the INFOMAX derivation, because our network is more general than the one used there. We want to use a gradient method to maximize E, which is a function of the Jacobians $\mathbf{J}^k$. Direct computation of the components of the gradient is very cumbersome and inefficient. However, from the theory of neural networks we know that, for any network, back-propagation is a simple and efficient method to compute the gradient of a function of its outputs relative to its weights (see, for example, work by Almeida, 1997, especially Section C.1.2.3.1). The network of Figure 1 doesn't output the Jacobians, which are what our objective function depends on. Therefore, to be able to efficiently compute the gradient of E, we need to first find a network that computes $\mathbf{J}^k$, and then backpropagate through that network.

The network that computes the Jacobians is essentially a linearized version of the network of Figure 1. To illustrate how to obtain such a network, we shall assume specific structures for the **F** and ψ blocks. We'll assume that the **F** block has a single hidden layer of sigmoidal units, linear output units, and no direct connections between input and output units. We'll assume a similar structure for each of the ψ blocks: a single hidden layer of sigmoidal units, a single linear output unit, and no direct connections between input and output units.

A network for computing $\mathbf{J}^k$, assuming this structure, is shown in Figure 2. The upper part of the figure shows the network of Figure 1, drawn in a different way. The **A** block represents the weight matrix of the hidden layer of **F**, and its output is the vector **Ao** (we'll denote both the block and the corresponding matrix by the same letter, since this does not cause any confusion; we'll also assume that **o** is augmented with a component $o_0 = 1$, and that the matrix **A** includes a corresponding column with the bias terms of the hidden layer units; the same is assumed for vector **y** and matrix **C**, which appear later). The leftmost Φ block applies the hidden layer's sigmoids to each of the components of **Ao**. Its outputs are the activations of the units of the hidden layer of **F**. Block **B** corresponds to the weight matrix of the output units of **F**, and its output is **y**. The ψ_i blocks, taken together, form an MLP with a single hidden layer and with linear output units. This MLP is special, in that the weights corresponding to connections between units of different ψ_i blocks are always zero, but otherwise it is similar to **F** in structure. It is represented, in Figure 2 by the upper **C**, rightmost Φ, and **D** blocks.

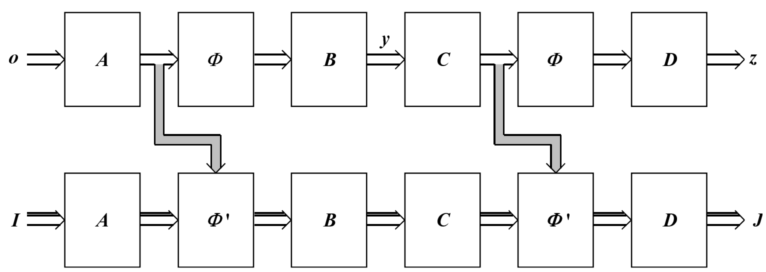

Figure 2: Network for computing the Jacobian.

The lower part of the figure is the one that computes the Jacobian itself, and is essentially a linearized version of the network of the upper part. An important difference, however, is that it propagates matrices, instead of vectors (this is depicted in the figure by the 3-D arrows). The network's input is the $n \times n$ identity matrix **I**, where n is, as above, the number of components of **o**, and also the number of independent components to be estimated. The output of the lower **A** block is $\mathbf{AI} = \mathbf{A}$.[2] This product of the identity by **A** might seem useless, but is useful later, in the backpropagation phase, for computing the partial derivatives of the objective function relative to the elements of **A**. The leftmost Φ' block performs a product by a diagonal matrix whose diagonal elements are the derivatives of the sigmoids of the corresponding units in the upper Φ block. Blocks **B**, **C** and **D** perform products by weight matrices, like **A**, and the rightmost Φ' block is similar to the leftmost one, corresponding to a diagonal matrix with the derivatives of the sigmoids of the upper,

2. The matrices **A** and **C**, in the lower part of the network and in the equations describing it, are stripped of the bias terms. Once this is noted, using the same letters for the upper- and lower-part matrices should cause no confusion.

rightmost Φ block. To compute the sigmoid derivatives, the two Φ' blocks need to receive the input activations of the corresponding hidden units from the upper part. This information is transferred through the gray arrows.

The output of the lower part of the network is the Jacobian of the transformation performed by the upper part (for the specific observation pattern being input at $\mathbf{o}$), and is given by

$$\mathbf{J} = \mathbf{D}\Phi_r'\mathbf{C}\mathbf{B}\Phi_l'\mathbf{A},$$

where Φ_r' and Φ_l' denote, respectively, the rightmost and leftmost Φ' diagonal matrices of Figure 2.

Once we have a network that outputs the Jacobian, the computation of the derivatives of the objective function relative to the network's weights essentially amounts to a backpropagation through this network. There are still a few details that are worth emphasizing, however.

The input to the backpropagation is made into the lower part of the network, and consists of

$$\frac{\partial E}{\partial \mathbf{J}} = \left(\mathbf{J}^{-1}\right)^T.$$

Nothing is input into the upper part, because E doesn't depend on $\mathbf{z}$, i.e. $\partial E/\partial \mathbf{z} = 0$.

The backpropagation must be performed along all of the network's paths. This means that there will be backpropagation along the gray arrows into the upper part, and this propagation will proceed backward through the upper part. Backpropagation through most blocks is rather straightforward, but the Φ' ones are somewhat unusual. Figure 3-a) shows a unit of one of these blocks, propagating in the forward direction. It is governed by

$$h_{ij} = \phi'(s_i)g_{ij},$$

where g_{ij} denotes a generic input into the block from the left arrow, s_i is the corresponding input from the gray arrow, and h_{ij} is the corresponding output towards the right arrow. The backward propagation is governed by the partial derivatives

$$\frac{\partial h_{ij}}{\partial g_{ij}} = \phi'(s_i)$$
$$\frac{\partial h_{ij}}{\partial s_i} = \phi''(s_i)g_{ij}.$$

The backpropagation unit is therefore as depicted in Figure 3-b), where each box denotes a product by the indicated value. Note that since the forward unit has two inputs, the backward unit has two outputs, one leading left in the lower part of Figure 2 and the other leading upward along the gray arrow.

All the network's weights, except the hidden units' biases, are shared by the upper and lower parts of the network. Since the lower part is linear and propagates matrices, it can be seen as n identical networks, each one propagating one of the columns of the identity matrix. Therefore the lower part's weights can also be seen as being shared among these n networks. The normal procedure for handling shared weights should be used: the partial derivatives relative to all the weight's instances are added, and the sum constitutes the partial derivative relative to the shared weight.

We should note that the method for computing the gradient of E that we have presented, despite having been described in detail for a specific network structure, is in fact rather general, being

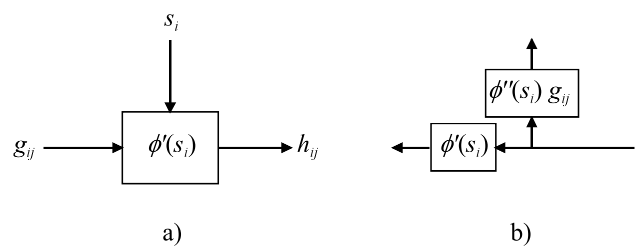

Figure 3: a) A unit of a Φ' block. b) The corresponding backpropagation unit.

applicable to nonlinear networks of any kind. In the examples presented in Section 3, the **F** block that was used had a slightly more complex structure than what has been assumed above, having also direct connections between its input and output layers. We have also made tests (Almeida, 2003a,b) where the **F** block was implemented by means of a radial basis function network.

Another important remark is that, according to our experience, the magnitudes of the partial derivatives involved in the gradient of E vary widely during the training. It is therefore essential to use a training procedure that dynamically adapts to the magnitudes of the gradient components. We have used an adaptive step sizes technique with error control (Almeida, 1997) with very good results. Conjugate gradient techniques would probably also be able to deal with such a situation in an efficient way.

Matlab-compatible code implementing MISEP is available at

`http://neural.inesc-id.pt/~lba/ICA/MIToolbox.html`.

3. Experimental Results

In this section we describe several experiments that were made to confirm the validity of the MISEP method. These experiments were mainly aimed at assessing the method's ability to perform ICA (extraction of independent components), and not blind source separation (recovery of the original sources). However, as shown by the results reported in Section 3.2, the method was able to recover the sources from nonlinear mixtures that involved relatively smooth nonlinearities.[3]

3.1 Linear ICA

There exist nowadays several different methods for performing linear ICA. In this context, MISEP is an extension of INFOMAX that has the advantage of learning the output nonlinearities during the training. It is therefore adaptive to the actual statistical distributions of the sources. These distributions don't have to be assumed a-priori, or estimated by some separate method. Our tests of linear ICA were mainly aimed at showing this adaptability of the method to different source distributions. To show this, the network that was used was the same in all tests: The **F** block was linear, yielding simply a product by a matrix. Each ψ block had a single hidden layer with four arctangent units, and a linear output unit. Each training set had 100 mixture vectors.

Figure 4 shows the separation of two supergaussian signals (speech and a supergaussian noise). The separation is virtually perfect. Figure 5 shows scatter plots of the original signals and of the

3. In this paper we use the term "smooth" with the meaning of "not deviating too much from linear." The examples that are given in Sections 3 and 4 further clarify that meaning.

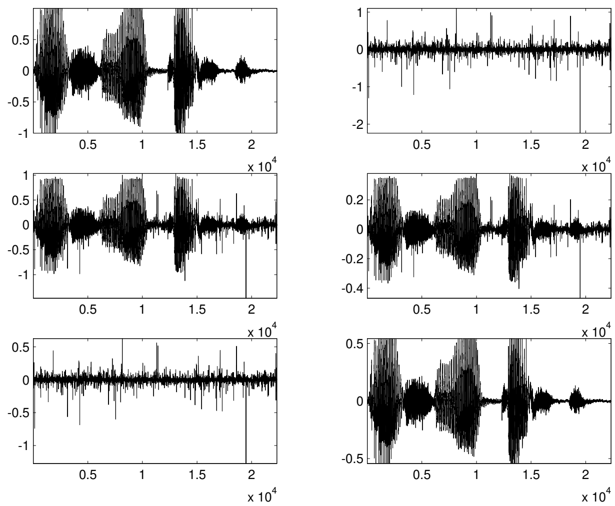

Figure 4: Separation of supergaussian signals. Top: source signals. Middle: mixtures. Bottom: separated signals.

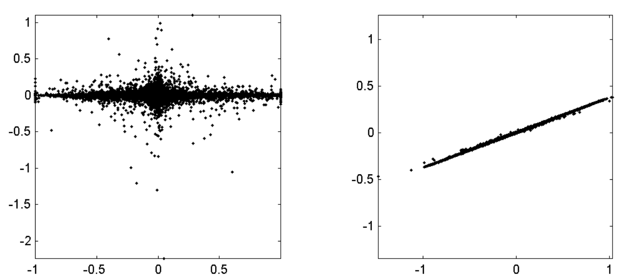

Figure 5: Scatter plots of the separation of supergaussian signals. Left: source signals; speech (horizontal) and noise (vertical). Right: mixtures.

mixture. The mixture can be seen to be almost singular. No prewhitening (also called sphering) was used, although it could have been used, if desired, and would probably have led to a faster training.

Figure 6 shows details of the network's operation. The scatter plot of **z** shows that the network's output approximated a uniform distribution rather well. Figure 7 shows the CPFs estimated by the ψ blocks. They agree well with the source distributions. The speech signal that we used had a somewhat skewed distribution, which is quite apparent in the estimated cumulative probability function.

Figures 8-11 illustrate a separation of speech and a strongly subgaussian, bimodal noise. Note the rather uniform distribution of **z**, and the estimation of the CPFs performed by the ψ blocks.

With two mildly subgaussian (uniformly distributed) sources, the system was also able to perform a good separation (these results are not shown here for brevity). With two strongly subgaussian, bimodal distributions the system sometimes yielded a good solution, corresponding to the absolute minimum of the mutual information (Figure 12). Sometimes it converged to a local min-

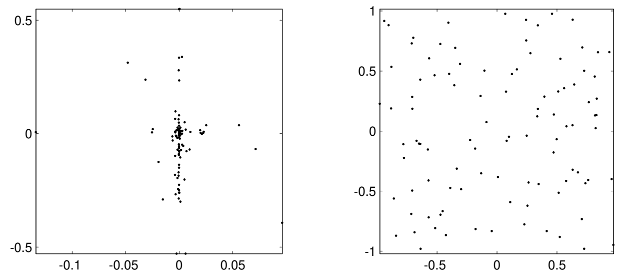

Figure 6: Scatter plots of the separation of supergaussian signals. Left: separated signals; speech (vertical) and noise (horizontal). Right: signals at the outputs of the ψ nets (note the uniform distribution). These plots show only the 100 patterns of the training set.

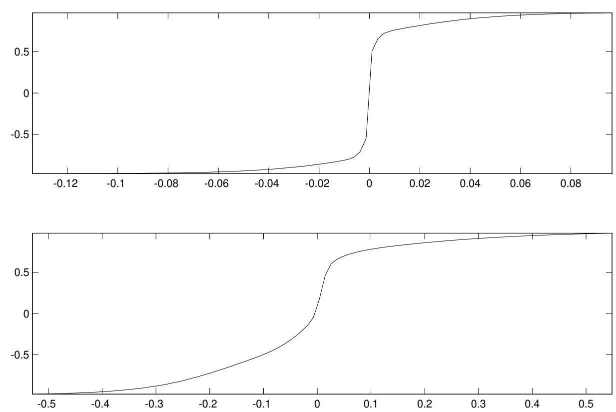

Figure 7: Separation of supergaussian signals – nonlinearities estimated by the ψ nets. Top: noise. Bottom: speech. These nonlinearities are estimates of the CPFs, apart from a re-scaling of the range to $[-1,1]$.

imum of the mutual information, in which one of the sources was well separated (Figure 13), and sometimes to another minimum in which none of the sources was separated (Figure 14). Local optima are a characteristic of mutual information, and of several other dependence measures, when there is more than one source that is multimodal. Several of the known linear ICA methods can converge to these local optima.

3.2 Nonlinear ICA

This section gives examples of nonlinear ICA tests. We first present several examples using two-component mixtures, and then a test with a four-component mixture. To illustrate the versatility of the MISEP method, the same network was used in all of the two-component cases. The **F** block had 20 arctangent hidden units, 10 of which were connected to each of the block's output units. It also had direct connections between input and output units, to be able to perfectly implement linear

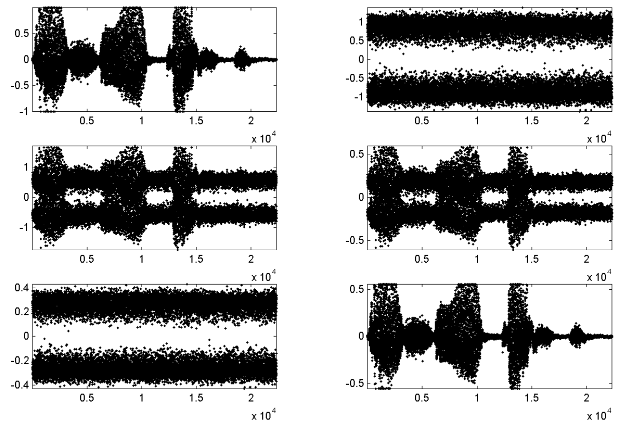

Figure 8: Separation of a supergaussian and a subgaussian signal. Top: source signals. Middle: mixtures. Bottom: separated signals. Samples are shown as unconnected dots for better visibility of the bimodal character of the noise.

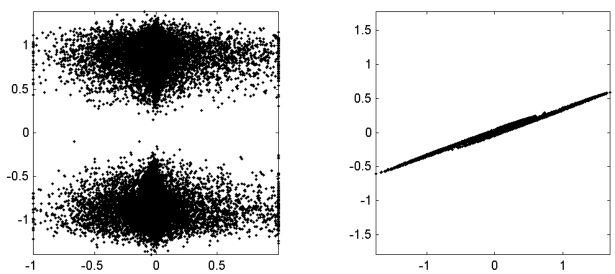

Figure 9: Scatter plots of the separation of a supergaussian and a subgaussian signal. Left: source signals. Right: mixtures.

separation, if necessary. Each ψ block had a single hidden layer with two arctangent units, and a linear output unit. Each training set had 1000 mixture vectors.

Figure 15 shows the separation of a nonlinear mixture of two speech signals, which are supergaussian. The mixture was of the form

$$o_1 = s_1 + a(s_2)^2 \tag{2}$$

$$o_2 = s_2 + a(s_1)^2 \tag{3}$$

With the value of a that was used, the signal to noise ratio (SNR) of o_1 relative to s_1 was 7.8 dB, and the SNR of o_2 relative to s_2 was 10.4 dB. After nonlinear separation, the SNR of y_1 relative to s_1 became 16.4 dB and the SNR of y_2 relative to s_2 was 17.4 dB. The average improvement was of 7.8 dB. Linear ICA, on the other hand, did not yield any improvement in the components of $\mathbf{y}$, relative to those of $\mathbf{o}$. This was expected: the specific mixture that was used had no "linear part", as can be seen both from the mixture equations and from the scatter plot, Figure 15-a). This kind of mixture was chosen specifically to evaluate the nonlinear capabilities of the method, since the linear part of the separation was known to be relatively easy to handle.

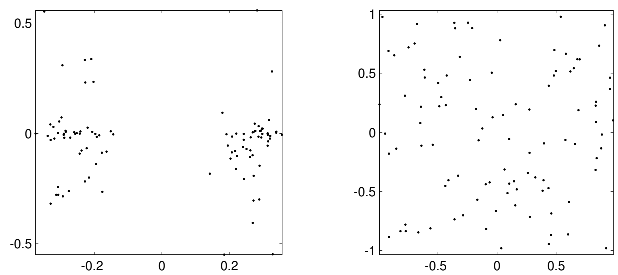

Figure 10: Scatter plots of the separation of a supergaussian and a subgaussian signal. Left: separated signals. Right: signals at the outputs of the ψ nets (note the uniform distribution). These plots show only the 100 patterns of the training set.

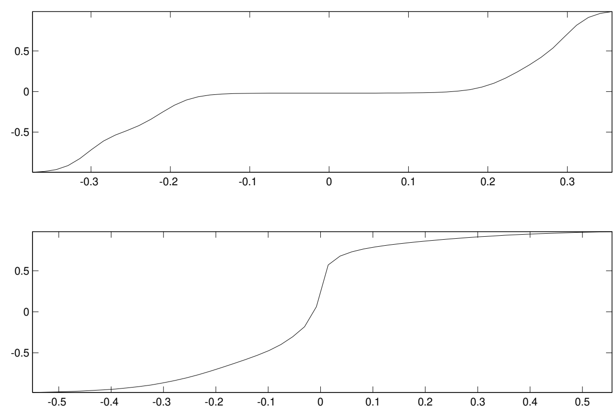

Figure 11: Separation of a supergaussian and a subgaussian signal: nonlinearities estimated by the ψ nets. Top: noise. Bottom: speech. These nonlinearities are estimates of the CPFs, apart from a re-scaling of the range to $[-1,1]$.

Figures 16 and 17 show the separation of a nonlinear mixture of a supergaussian and a subgaussian, and of two subgaussian sources, respectively. As in the linear case, if more than one source is multimodal, there are local optima of the mutual information, in which the optimization may get trapped. Examples of such cases are not shown here, for brevity.

In a larger test we used four sources, two subgaussian (sources 1 and 2) and two supergaussian (sources 3 and 4). Figure 18 shows scatter plots of pairs of sources. The mixture was of the form

$$o_i = s_i + a_i \sum_{j \neq i} [s_i s_j + (s_j)^2].$$

Figure 19 gives examples of scatter plots of the mixture components. Note that in a 4-dimensional distribution, pairwise scatter plots don't always give a full idea of the distribution. For example, in the center and right-hand plots of Figure 19 the supergaussian components appear somewhat "fuzzy" because these are projections from a 4-dimensional space into a 2-dimensional one. In the 4-dimensional space the supergaussian components remain as sharp as the corresponding sources.

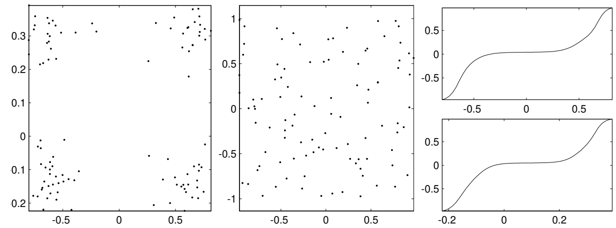

Figure 12: Separation of two subgaussian, bimodal signals at an absolute minimum of the mutual information. Left: scatter plot of the separated signals. Center: scatter plot of outputs of the ψ nets. Right: nonlinearities estimated by the ψ nets.

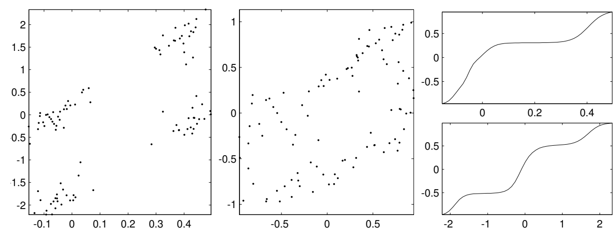

Figure 13: Separation of two subgaussian, bimodal signals at a local minimum of the mutual information. Left: scatter plot of the separated signals. Center: scatter plot of outputs of the ψ nets (note the non-uniform distribution). Right: nonlinearities estimated by the ψ nets.

The a_i coefficients in (4) were chosen so as to yield a degree of nonlinearity that was clearly visible (and that is best illustrated by the scatter plots of Figure 19), but small enough for the method to still be able to approximately separate the original sources. This gives an idea of the degree of nonlinearity that MISEP is able to handle, if the purpose is source separation and not simply ICA.

The network that we used in this case had a structure that was very similar to the one used in the previous tests. Of course, the **F** block had four inputs and four outputs. It now had 40 hidden units, divided into four sets of 10, each set connecting to one of the four output units. It also had direct connections between input and output units. Each of the four ψ blocks had two hidden units. Figure 20 shows scatter plots of the extracted components. We see that the system was able to recover the sources quite well, although not perfectly.

Regarding convergence speed, the two-source nonlinear ICA tests, with batch-mode training and with training sets of 1000 patterns, normally converged in less than 400 epochs. On a 400 MHz Pentium processor running a Matlab implementation of the method, these 400 epochs took less than 4 minutes. The four-source results were obtained in 1000 epochs, also with a training set of 1000 patterns. These 1000 epochs took less than 20 minutes on the same processor.

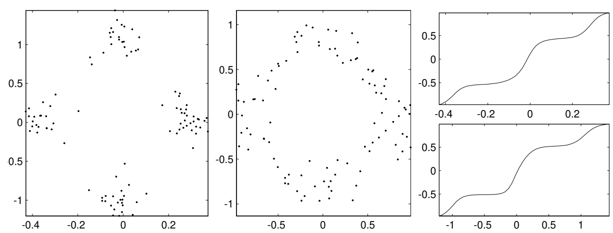

Figure 14: Separation of two subgaussian, bimodal signals at another local minimum of the mutual information. Left: scatter plot of the separated signals. Center: scatter plot of the outputs of the ψ nets (note the non-uniform distribution). Right: nonlinearities estimated by the ψ nets.

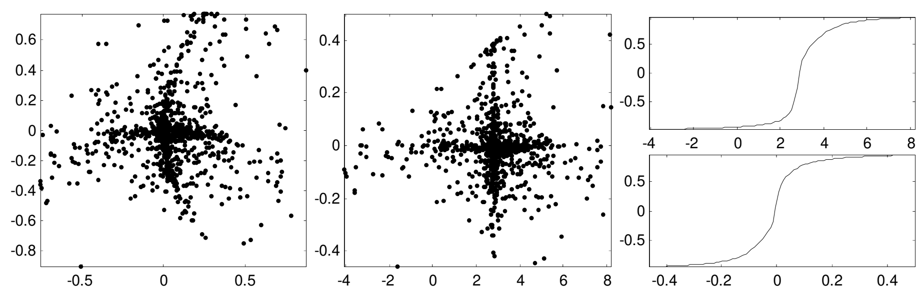

Figure 15: Separation of a nonlinear mixture of two speech signals. Left: scatter plot of the mixed signals. Center: scatter plot of the separated signals. Right: CPFs learned by the system.

4. Some Comments on Source Separability

In all the examples of nonlinear mixtures that were given above, the MISEP method was able to perform not only ICA, but source separation as well, despite the fact that nonlinear BSS is an ill-posed problem. The tests that we described used only the regularization inherently performed by MLPs with small initial weights and with a relatively small number of hidden units.[4] No explicit regularization was used in our experiments, although it could have been easily incorporated, if necessary (e.g. through weight decay in the **F** block). The nonlinear mixtures that we used could not be inverted, in an exact way, by the MLP-based **F** block, which means that the unmixing block was not matched to the mixtures. This block had, therefore, to estimate an approximation to the true unmixing transformations.

We do not argue that the kind of regularization that we used will be the best in all nonlinear BSS situations (or even that it was the best for the specific mixtures that we've tested). Other alternatives

4. These MLPs implement functions that are approximately linear and thus rather smooth: On the one hand, gradient-based optimization, being local, tends to find weights that are relatively close to the initial ones, and therefore are also small. Initializing the **F** MLP with small weights therefore tends to lead to ICA solutions in which the MLP implements a smooth transformation. On the other hand, MLPs with a small number of hidden units have a limited capability to yield unsmooth transformations. Both of these are forms of implicit regularization.

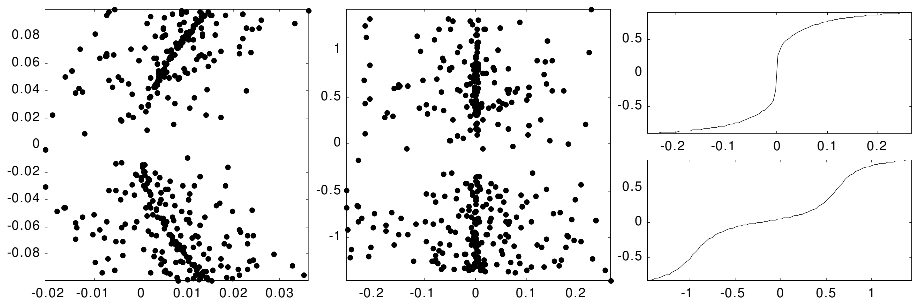

Figure 16: Separation of a nonlinear mixture of a supergaussian and a subgaussian signal. Left: scatter plot of the mixed signals. Center: scatter plot of the separated signals. Right: CPFs learned by the system.

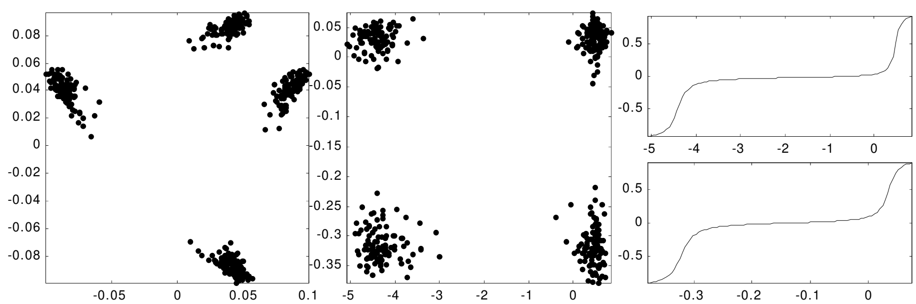

Figure 17: Separation of a nonlinear mixture of two subgaussian signals. Left: scatter plot of the mixed signals. Center: scatter plot of the separated signals. Right: CPFs learned by the system.

are, for example, the use of an explicit regularization term in MISEP, or the use of prior distributions within a Bayesian setting (as in ensemble learning, Valpola, 2000), for example.

A question that has often been asked to the author is whether it is possible to characterize the kinds of nonlinear mixtures for which MISEP will be able to recover the original sources (or, more generally, the cases in which nonlinear ICA will be able to do so). The answer, at this stage, has to be mostly negative: it is not yet possible to make such characterizations in any precise way. While linear mixtures are rather constrained, nonlinear mixtures have a very wide range of variability. If source recovery is to be achieved from a nonlinear mixture, some prior information has to be used. The possible kinds of prior information also have a wide range of variability (e.g. smoothness of the mixture, strict constraints on the form of the mixture, knowledge about the sources' distributions...). The study of conditions for nonlinear source separability is quite an extensive topic in itself, and will probably take some time to be dealt with in a rigorous way. The author's experience shows that, the more prior information we incorporate into the separating method, the better will be the resulting separation, as one would expect.

Given the impossibility of giving a precise, or even an approximate characterization of source separability conditions, we'll limit ourselves to giving an example of a situation that is unseparable

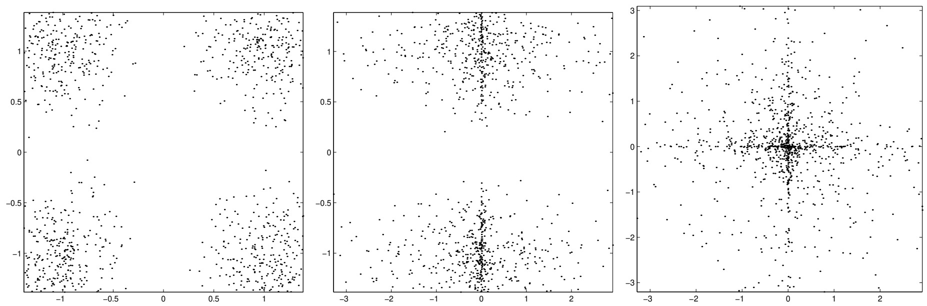

Figure 18: Separation of a nonlinear mixture of four signals, two subgaussian and two supergaussian. Left: scatter plot of the two subgaussian sources (# 1 and 2). Center: scatter plot of a subgaussian and a supergaussian source (# 1 and 3). Right: scatter plot of the two supergaussian sources (# 3 and 4).

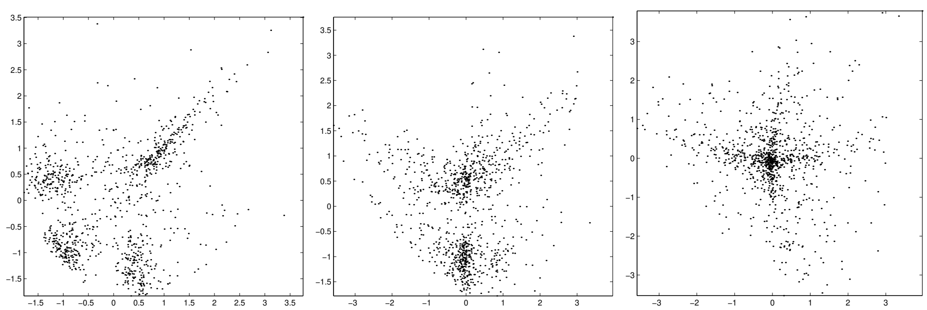

Figure 19: Separation of a nonlinear mixture of four signals, two subgaussian and two supergaussian. Left: scatter plot of mixture components 1 and 2. Center: scatter plot of mixture components 1 and 3. Right: scatter plot of mixture components 3 and 4.

with the present form of MISEP (although it may become separable with future improvements). In this unseparable example the two sources are supergaussian, and the mixture is quadratic, as in (2) and (3), but the a coefficient is given a large value. The scatter plot of the mixture, as well as the scatter plots of the extracted components at various stages along the optimization, are shown in Figure 21. In somewhat loose terms, we can say that the system wrongly aligned the two outer "half-branches" of the sources with each other, these having become the horizontal extracted component. The two inner half-branches ended up being almost merged together, and essentially formed the vertical extracted component. Although the original sources are strongly mixed in the extracted components, these components are almost independent from each other after the 2000 epochs shown (and the objective function was still slowly improving at that point).

This an example of a case in which ICA was approximately achieved, but source separation was not. It is a case where the mixture was, in our terms, too unsmooth, deviating too much from a linear one. With this mixture, MISEP consistently failed to separate the sources. On the other

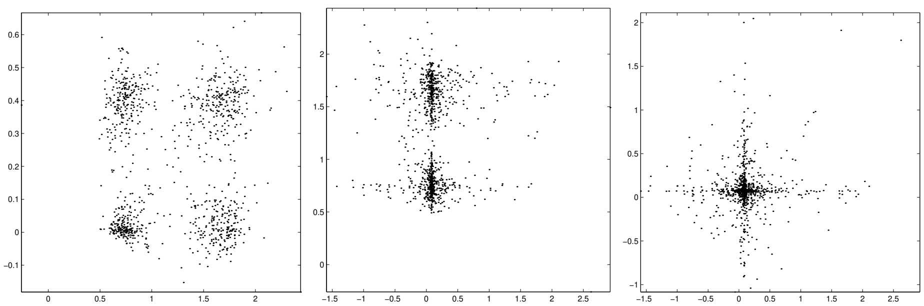

Figure 20: Separation of a nonlinear mixture of four signals, two subgaussian and two supergaussian. Left: scatter plot of extracted components 1 and 2. Center: scatter plot of extracted components 1 and 3. Right: scatter plot of extracted components 3 and 4.

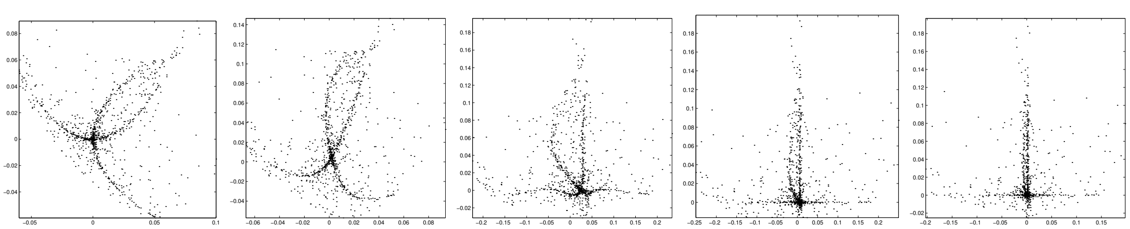

Figure 21: A case in which ICA was achieved, but source separation was not. The leftmost scatter plot shows the mixture components. Then, from left to right, the scatter plots show the extracted components after 45, 150, 330 and 2000 training epochs, respectively.

hand, in the cases reported in Section 3, the method consistently was able to perform separation,[5] when tried with different initializations of the MLP's weights and, in the cases where sources were randomly generated, also when tried with different realizations of the sources, obeying the same source distributions.

A question that may asked regarding the structure of Figure 1 (and one that was actually asked by one of our anonymous reviewers), is why the ψ blocks are needed. It is true that, in principle, one could simply use the **F** block, with its outputs bounded within a hypercube, and maximize its output entropy. A uniform distribution of $\mathbf{y}$ in that hypercube would result, if the **F** block were flexible enough, and therefore the y_i would be independent from one another.

This issue is related to the smoothness one, and that is why it is discussed here. The key phrase in the above reasoning is "if the **F** block were flexible enough." Excluding situations in which all the sources have close-to-uniform distributions, the **F** block will have to perform a rather unsmooth transformation to fit the extracted components uniformly into a hypercube. Given the above discussion on the need of the smoothness constraint for achieving source separation, we see that we can't expect this unsmooth **F** to be able to recover the original sources in such a situation,

5. In the situations involving two bimodal sources, separation was always achieved when the system converged to the absolute minimum of the mutual information. However, the system sometimes converged to local minima, as previously mentioned.

even though it will yield independent components. By separating the unmixing **F** block from the ψ ones (which have the purpose of turning each component into a uniformly distributed one), we can keep a smooth **F** (even applying explicit regularization to it if appropriate), while allowing the ψ blocks to perform the transformation of the y_i into uniformly distributed variables, these blocks being allowed to be rather unsmooth if necessary. The structuring of the network into separate **F** and ψ blocks thus gives it a much greater ability to perform nonlinear source separation.

5. Conclusions

We have presented MISEP, a method for performing ICA by minimizing the mutual information of the estimated components. Some of the features of the method are:

- It is able to perform both linear and nonlinear ICA.
- It adapts to the statistical distributions of the estimated components. It can therefore deal with a wide range of source distributions.
- It uses a single network to perform both the ICA operation and the estimation of the distributions of the sources. This network is optimized according to a single objective function, the output entropy.

We have presented experimental results that show the capability of MISEP to perform both linear and nonlinear ICA. We have also shown examples in which blind source separation was performed on relatively smooth nonlinear mixtures, using this smoothness as an assumption to handle the ill-posedness of nonlinear source separation. The smoothness regularization that we used in the experiments presented in this paper was only the one implicitly performed by MLPs with small initial weights and with relatively few hidden units.

MISEP is not the only currently available alternative for nonlinear ICA/BSS. Some other methods that deserve mention are those described by Yang et al. (1998); Marques and Almeida (1999); Valpola (2000); Harmeling et al. (2001); Martinez and Bray (2003). We shall not make detailed comparisons with these methods here. But we wish to emphasize that we believe that, among the currently available nonlinear ICA/BSS methods, MISEP is the one of the very few that simultaneously have the following qualities:

- having a relatively simple implementation,
- being quite flexible in terms of the kinds of nonlinear separating networks that it can use,
- being able to handle a large variety of source distributions,
- not needing to rely on temporal structure of the sources,
- being able to easily incorporate various forms of regularization,
- being relatively efficient in computational terms.

On the other hand, it is clear from the discussion and the examples presented in this paper that MISEP, in its present form, can only perform source separation when the nonlinearities involved in the mixture are not too strong. In this respect, especially the method of Harmeling et al. (2001) seems to be more powerful (although it may have drawbacks in some other aspects).

Many issues remain open, regarding the MISEP method, and will be addressed in future work. Some of them are:

- To further study the method in blind and semi-blind source separation settings, clarifying the kinds of mixtures that can be separated, the kinds of prior information that can be used, and the role of regularization.
- Finding ways to measure the quality of the separation that is obtained. The measures currently used for linear ICA will probably not be appropriate, due to the possibility that the separated components are nonlinearly transformed relative to the original sources, as discussed in the end of Section 1.
- To make the method able to deal with stronger nonlinearities.
- To study the behavior of the method with larger numbers of sources and with noisy observations. It can be mentioned that preliminary results with up to 10 sources have already been obtained. These will be reported in a forthcoming paper (Almeida, 2003b).
- To study the extension of the method to under- and over-determined situations, to non-stationary and non-instantaneous mixtures, etc.
- The application of the method to real-life problems. An application to a real-life nonlinear image separation problem has already started to show promising results, although these are still in too early a stage to be reported here.

Acknowledgments

The author wishes to acknowledge the anonymous reviewers for their comments, which helped to significantly improve the quality of this paper. This work was partially supported by Praxis project P/EEI/14091/1998 and by the European IST project BLISS.

References

L. B. Almeida. Multilayer perceptrons. In E. Fiesler and R. Beale, editors, *Handbook of Neural Computation*. Institute of Physics, Oxford University Press, 1997. Available electronically at `http://neural.inesc-id.pt/~lba/papers/AlmeidaHNNC.ps.zip`.

L. B. Almeida. Linear and nonlinear ICA based on mutual information. In *Proc. Symp. 2000 on Adapt. Sys. for Sig. Proc., Commun. and Control*, Lake Louise, Alberta, Canada, 2000a.

L. B. Almeida. Simultaneous MI-based estimation of independent components and of their distributions. In *Proc. Second Int. Worksh. Independent Component Analysis and Blind Signal Separation*, pages 169–174, Helsinki, Finland, 2000b.

L. B. Almeida. ICA of linear and nonlinear mixtures based on mutual information. In *Proc. 2001 Int. Joint Conf. on Neural Networks*, Washington, D.C., 2001.

L. B. Almeida. MISEP – an ICA method for linear and nonlinear mixtures, based on mutual information. In *Proc. 2002 Int. Joint Conf. on Neural Networks*, Honolulu, Hawaii, 2002.

L. B. Almeida. Faster training in nonlinear ICA using MISEP. In *Proc. Int. Worksh. Independent Component Analysis and Blind Signal Separation*, Nara, Japan, 2003a.

L. B. Almeida. Linear and nonlinear ICA based on mutual information – the MISEP method. *Signal Processing*, 2003b. To appear.

S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. In *NIPS 95*, pages 882–893. MIT Press, 1996.

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

G. Burel. Blind separation of sources: A nonlinear neural algorithm. *Neural Networks*, 5(6):937–947, 1992.

J.-F. Cardoso. Infomax and maximum likelihood for source separation. *IEEE Letters on Signal Processing*, 4:112–114, 1997.

J.-F. Cardoso and A. Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM Journal of Matrix Analysis and Applications*, 17(1), 1996.

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

G. Darmois. Analyse générale des liaisons stochastiques. *Rev. Inst. Internat. Stat.*, 21:2–8, 1953.

G. Deco and W. Brauer. Nonlinear higher-order statistical decorrelation by volume-conserving neural architectures. *Neural Networks*, 8:525–535, 1995.

S. Harmeling, A. Ziehe, M. Kawanabe, B. Blankertz, and K. Mueller. Nonlinear blind source separation using kernel feature spaces. In T-W. Lee, editor, *Proc. Int. Worksh. Independent Component Analysis and Blind Signal Separation*, 2001.

S. Haykin and P. Gupta. A new activation function for blind signal separation. ASL Technical Report 1, McMaster University, Hamilton, Ontario, Canada, 1999.

A. Hyvarinen, J. Karhunen, and E. Oja. *Independent component analysis*. J. Wiley, 2001.

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

A. Hyvarinen and P. Pajunen. Nonlinear independent component analysis: Existence and uniqueness results. *Neural Networks*, 12(3):429–439, 1999.

T.-W. Lee, M. Girolami, A. Bell, and T. Sejnowski. An unifying information-theoretic framework for independent component analysis. *International Journal on Mathematical and Computer Modeling*, 1998.

T.-W. Lee, M. Girolami, and T. Sejnowski. Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources. *Neural Computation*, 11: 417–441, 1999.

G. C. Marques and L. B. Almeida. An objective function for independence. In *Proc. International Conference on Neural Networks*, pages 453–457, Washington DC, 1996.

G. C. Marques and L. B. Almeida. Separation of nonlinear mixtures using pattern repulsion. In J. F. Cardoso, C. Jutten, and P. Loubaton, editors, *Proc. First Int. Worksh. Independent Component Analysis and Signal Separation*, pages 277–282, Aussois, France, 1999.

D. Martinez and A. Bray. Nonlinear blind source separation using kernels. *IEEE Trans. on Neural Networks*, 14(1), Jan. 2003.

F. Palmieri, D. Mattera, and A. Budillon. Multi-layer independent component analysis (MLICA). In J. F. Cardoso, C. Jutten, and P. Loubaton, editors, *Proc. First Int. Worksh. Independent Component Analysis and Signal Separation*, pages 93–97, Aussois, France, 1999.

A. Taleb and C. Jutten. Entropy optimization - application to blind separation of sources. In *Proc. ICANN'97*, Lausanne, Switzerland, 1997.

H. Valpola. Nonlinear independent component analysis using ensemble learning: Theory. In *Proc. Second Int. Worksh. Independent Component Analysis and Blind Signal Separation*, pages 251–256, Helsinki, Finland, 2000.

H. Yang, S. Amari, and A. Cichocki. Information-theoretic approach to blind separation of sources in nonlinear mixture. *Signal Processing*, 64(3):291–300, February 1998. Available electronically at `http://citeseer.nj.nec.com/article/yang98informationtheoretic.html`.