An Experimental Comparison of Neural Algorithms for Independent Component Analysis and Blind Separation

Xavier Giannakopoulos[1,2], **Juha Karhunen**[1], **and Erkki Oja**[1]

[1]Helsinki University of Technology, Laboratory of Computer and Information Science
P.O.Box 5400, FIN-02015 HUT, Espoo, FINLAND
[2]IDSIA, Corso Elvezia 36, Lugano, Switzerland

email: `xavier@idsia.ch, Juha.Karhunen@hut.fi, Erkki.Oja@hut.fi`
Fax: +358-9-451 3277; http://www.cis.hut.fi

June 2, 1999

Abstract

In this paper, we compare the performance of five prominent neural or adaptive algorithms designed for Independent Component Analysis (ICA) and blind source separation (BSS). In the first part of the study, we use artificial data for comparing the accuracy, convergence speed, computational load, and other relevant properties of the algorithms. In the second part, the algorithms are applied to three different real-world data sets. The task is either blind source separation or findind interesting directions in the data for visualisation purposes. We develop criteria for selecting the most meaningful basis vectors of ICA and measuring the quality of the results. The comparison reveals characteristic differences between the studied ICA algorithms. The most important conclusions of our comparison are robustness of the ICA algorithms with respect to modest modeling imperfections, and the superiority of fixed-point algorithms with respect to the computational load.

Keywords: Comparison study, independent component analysis, blind separation, learning algorithms, neural networks.

1 Introduction

Independent Component Analysis (ICA) [8, 16] is an unsupervised technique which tries to represent a set of input data in terms of statistically independent variables. ICA has lately drawn a lot of attention both in unsupervised neural learning and statistical signal processing. In particular, efficient new neural learning algorithms [1, 2, 7, 13, 18, 20, 21, 26, 31, 32] have been introduced for ICA and the closely related blind source separation (BSS) problem. In basic blind source separation, the goal is to recover mutually independent but otherwise unknown original source signals from their linear instantaneous mixtures without knowing the mixing coefficients. ICA and BSS have many applications especially in signal and image processing [19, 23].

In developing applications, it is important to know the computational and other properties of available ICA algorithms. This calls for an experimental comparison. In this paper, we compare five neural or semi-neural ICA algorithms using both artificially generated and real-world data. Artificial data is needed in examining the accuracy and convergence speed, because only then the correct results are known. We also apply the same ICA algorithms to three real-world data sets. In this case, the correct or theoretically best results are unknown, and it is possible to compare the algorithms only mutually or with respect to their performance in the application at hand.

The rest of the paper is organized as follows. In the next section, the basic data model used in ICA and BSS is discussed. In section 3, the algorithms included in this comparison are briefly reviewed. Section 4 contains the experimental results for artificial data. In section 5, the comparison methods used for real-world data are introduced, and section 6 gives the results. The final section contains the main conclusions of our study.

2 Problem formulation

We consider the standard linear data model used in independent component analysis and blind source separation [4, 8, 16, 20]:

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

Here the column vector $\mathbf{s}(t)$ is called the source vector or independent component vector. Its m components $s_i(t)$, $i = 1, \ldots, m$, are the m unknown, mutually statistically independent components or source signals at time or index value t. For simplicity, they are assumed to be zero mean and stationary. The components of the n-dimensional data or mixture vector $\mathbf{x}(t)$ are some linear mixtures of these independent components or sources. The $n \times m$ mixing matrix $\mathbf{A}$ is an unknown full rank constant matrix with $n \geq m$. The column vectors $\mathbf{a}_i$, $i = 1, \ldots, m$, of the mixing matrix $\mathbf{A}$ are the basis vectors of ICA. The full rank assumption guarantees that the basis vectors of ICA are linearly independent, and the assumption $n \geq m$ says that there are at least as many mixtures as source signals. At most one of the independent components or source signals $s_i(t)$ is allowed to be Gaussian.

In neural and adaptive ICA and BSS methods, an $m \times n$ inverse or separating matrix $\mathbf{B}(t)$ is updated for learning the ICA expansion (1) so that the m-vector

$$\mathbf{y}(t) = \mathbf{B}(t)\mathbf{x}(t) \tag{2}$$

becomes an estimate $\mathbf{y}(t) = \hat{\mathbf{s}}(t)$ of the independent components. The found independent components or source signals are unordered, so that the estimate $\hat{s}_i(t)$ of the i:th independent component may appear in any component $y_j(t)$ of $\mathbf{y}(t)$. The amplitudes $y_j(t)$ are usually scaled to have a unit variance. The basis vectors $\mathbf{a}_i$ of ICA can be estimated as column vectors of the pseudoinverse $\mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}$ of $\mathbf{B}$ if necessary [20]. If the separating matrix $\mathbf{B}$ is square ($m = n$), this expression reduces to the standard inverse $\mathbf{B}^{-1}$.

The assumptions made on the model (1) are discussed in more detail in several papers, for example in [8, 20, 23]. It should be noted that in practical situations some of the assumptions may not hold exactly. For example, the actual number m of independent components is often unknown and may sometimes be larger than the number n of available mixtures, or the source signals can be more or less nonstationary. Noise (typically Gaussian) may also be present, but the additive noise term $\mathbf{n}(t)$ is usually omitted from the model (1), because it is impossible to separate the noise from the sources without additional information [18]. In this paper, we study also the robustness of ICA against validity of the model assumptions.

In their basic form, ICA and BSS problems are much the same, utilizing the same data model (1) with the same assumptions. However, the viewpoint is somewhat different, so that in blind source separation the main interest lies in recovering the source signals $s_i(t)$. On the other hand, in several applications of ICA the basis vectors $\mathbf{a}_i$ of ICA are of primary interest, the independent components $s_i(t)$ being auxiliary variables only. It should also be noted that if the data model is more complicated, for example nonlinear, the ICA and BSS problems may significantly differ from each other.

3 Neural ICA and BSS algorithms

In this section, we briefly describe the five algorithms included in our study. More details can be found in the given references. These five algorithms were chosen because they are well-known or were considered to be prominent representatives of their algorithmic group. Quite recently, many new neural algorithms have been proposed for ICA and BSS, but it was not possible to include them all in our comparison study. Furthermore, in most cases these new algorithms are fairly close to the existing ones at least in their basic principles. In the comparisons, we have used the original MATLAB codes provided by the authors whenever possible. The web addresses of these codes and the real-world data sets used in the second part of our study are given in the Appendix.

In several neural ICA or BSS algorithms, the data vectors $\mathbf{x}(t)$ are preprocessed by whitening (sphering) them:

$$\mathbf{v}(t) = \mathbf{V}(t)\mathbf{x}(t). \tag{3}$$

Here $\mathbf{v}(t)$ denotes the t:th whitened vector, and $\mathbf{V}(t)$ is an $m \times n$ whitening matrix. After whitening, the components of the vectors $\mathbf{v}(t)$ are uncorrelated with unit variances. Mathematically, they satisfy the condition $\mathrm{E}\{\mathbf{v}(t)\mathbf{v}(t)^T\} = \mathbf{I}$, where $\mathbf{I}$ is the unit matrix and E denotes expectation value.

Whitening can be done in many ways [20]. Standard principal component analysis is often applied because of its optimal data compression properties. At this stage, the dimensionality of the data vectors may be dropped from n to m if necessary. After prewhitening the subsequent $m \times m$ separating matrix $\mathbf{W}(t)$ can be taken orthogonal: $\mathbf{W}^T\mathbf{W} = \mathbf{I}$, which usually improves convergence of the separation algorithms and makes the separation task easier. Thus in whitening approaches the total separating matrix is $\mathbf{B}(t) = \mathbf{W}(t)\mathbf{V}(t)$.

In other algorithms, the separating matrix $\mathbf{B}(t)$ is estimated directly, which may be advantageous if some of the sources are weak or the mixing matrix is close to singularity [4]. However, in large-scale problems one must usually resort to prewhitening for getting the sources separated.

In most neural separation algorithms, different type of nonlinearities must be applied to sub-Gaussian and super-Gaussian sources. For sub-Gaussian sources $s_i(t)$, the simple fourth-order statistics kurtosis defined by

$$\kappa[s_i(t)] = \mathrm{E}\{s_i(t)^4\} - 3[\mathrm{E}\{s_i(t)^2\}]^2 \tag{4}$$

is negative, while it is positive for super-Gaussian sources. The kurtosis of Gaussian sources is zero. Otherwise, the nonlinearity may be chosen fairly freely [5], though it affects the final accuracy of separation.

3.1 Fixed-point (FP) algorithms

One iteration of the generalised fixed-point algorithm for finding a row vector $\mathbf{w}_i^T$ of the orthogonal separating matrix $\mathbf{W}$ is [26, 14]

$$\begin{aligned} \mathbf{w}_i^* &= \mathrm{E}\{\mathbf{v}g(\mathbf{w}_i^T\mathbf{v})\} - \mathrm{E}\{g'(\mathbf{w}_i^T\mathbf{v})\}\mathbf{w}_i \\ \mathbf{w}_i &= \mathbf{w}_i^*/\|\mathbf{w}_i^*\|. \end{aligned} \tag{5}$$

Here $g(y)$ is a suitable nonlinearity, typically $g(y) = y^3$ or $g(y) = \tanh(y)$, and $g'(y)$ is its derivative with respect to y. The expectations are in practice replaced by their sample means. Hence the fixed-point algorithm is not truly a neural adaptive algorithm. The algorithm requires prewhitening of the data. The vectors $\mathbf{w}_i$ must be orthogonalised against each other; this can be done either sequentially or symmetrically [14, 10]. Usually the algorithm (5) converges after 5-20 iterations. The basic sequential version with tanh nonlinearity is denoted by **FPth**, the respective symmetric version by **FPsymth**.

The original fixed-point algorithm (**FP**) [15] uses the cubic nonlinearity $g(y) = y^3$. Then $\mathrm{E}\{g'(\mathbf{w}_i^T\mathbf{v})\} = 3 \parallel \mathbf{w} \parallel^2$, making the algorithm simpler with a fast cubic convergence. However, this version is less accurate.

3.2 Natural gradient algorithm (ACY)

Originally proposed on heuristic grounds (see [7]), this popular and simple neural gradient algorithm [1, 32] was derived by applying the natural gradient concept to an earlier algorithm introduced by Bell and Sejnowski [2]. It is often called the Amari-Cichocki-Yang (**ACY**) algorithm. The update rule for the separating matrix $\mathbf{B}$ is

$$\Delta \mathbf{B} = \mu[\mathbf{I} - \mathbf{g}(\mathbf{y})\mathbf{y}^T]\mathbf{B}. \tag{6}$$

We have omitted the time index t from all the quantities in (6) for simplicity. Thus the complete updating formula is $\mathbf{B}(t+1) = \mathbf{B}(t) + \Delta\mathbf{B}(t)$. The notation $\mathbf{g}(\mathbf{y})$ means that the nonlinearity $g(y)$ is applied to all the component of the vector $\mathbf{y} = \mathbf{Bx}$. In principle, it would be possible to apply a specific nonlinearity $g_i(y_i)$ to each component y_i of $\mathbf{y}$ separately in all the algorithms if necessary. The learning parameter μ is usually a small constant.

The code used for the basic natural gradient algorithm (6) actually computes a batch gradient for stability reasons. In spite of this, this algorithm may suffer from convergence problems in large-scale problems. Therefore whitening is often applied as a preprocessing step for improving the convergence properties. The update rule (6) takes then the form

$$\Delta \mathbf{W} = \mu[\mathbf{I} - \mathbf{g}(\mathbf{y})\mathbf{y}^T]\mathbf{W} \tag{7}$$

where now $\mathbf{y} = \mathbf{Wv}$. This version is denoted by **WACY**, and it is using the tanh nonlinearity. - The algorithm **BS** appearing in some figures is quite similar to WACY, but it uses the standard sigmoid $g(y) = e^y(1+e^y)^{-1}$.

The main advantages of the algorithms (6) and (7) are that they are simple, truly neural, and have been derived by minimizing the mutual information which is an information-theoretic measure for statistical independence.

3.3 Extended Bell-Sejnowski algorithm (ExtBS)

This algorithm is also based on the update rule (7), but the following features have been added. First, kurtosis is estimated on-line [11] for handling both super-Gaussian and sub-Gaussian sources simultaneously. This can be done by using the nonlinearity

$$\mathbf{g}(\mathbf{y}) = \mathrm{sign}[\mathbf{k}(\mathbf{y})]\tanh(\mathbf{y}) + \mathbf{y}, \tag{8}$$

where $\mathbf{k}(\mathbf{y})$ denotes the vector that contains the estimated kurtosis values $\kappa[y_i(t)]$ of the components of $\mathbf{y}$ [11]. The vector multiplication in (8) is defined as a Kronecker product (the corresponding elements are multiplied). Thus the sign of the tanh term is either negative or positive depending on the kurtosis of the respective component of the output vector $\mathbf{y}$. Second, the learning parameter μ is not a constant, but is optimised using simulated annealing.

When reducing the dimension from n to m by using PCA (Principal Component Analysis) in context with prewhitening, the resulting algorithm is denoted by **ExtBSpc**, when reducing the dimension by using rectangular ICA, it is denoted by **ExtBSic**.

There exists several other more advanced but also complicated versions of the basic natural gradient algorithm (6). For example in [34], the probability densities of the source signals are modeled and estimated by using mixtures of densities.

3.4 EASI algorithm

The **EASI** algorithm is introduced as an adaptive signal processing algorithm in [4], but it can as well be used in the neural network area for blind source separation and ICA. The general update formula for the total separating matrix $\mathbf{B}$ is in its normalized form [4]

$$\Delta\mathbf{B} = \mu \left[\frac{\mathbf{I} - \mathbf{y}\mathbf{y}^T}{1 + \mu\mathbf{y}^T\mathbf{y}} + \frac{\mathbf{y}\mathbf{g}(\mathbf{y}^T) - \mathbf{g}(\mathbf{y})\mathbf{y}^T}{1 + \mu|\mathbf{y}^T\mathbf{g}(\mathbf{y})|} \right] \mathbf{B} \tag{9}$$

The normalization terms in the denominator are needed for keeping the algorithm stable especially if the nonlinearity $\mathbf{g}(\mathbf{y})$ is growing fast, for example when $\mathbf{g}(\mathbf{y}) = \mathbf{y}^3$. Otherwise, these terms may often be dropped out. On-line estimation of the kurtosis was added to the initial code for handling super-Gaussian and sub-Gaussian sources at the same time when we studied real-world data.

3.5 RLS algorithm for a nonlinear PCA criterion (NPCA-RLS)

This recursive least-squares (RLS) algorithm [21, 27] for a nonlinear PCA criterion (abbreviated here **NPCA-RLS**) is an improved version of the nonlinear PCA algorithm for BSS problems, which in turn is discussed and analyzed in [20, 25]. Originally, the nonlinear PCA criterion and algorithm were introduced and analyzed already several years earlier [33, 17], but their proper use in blind source separation was not understood at that time.

The basic symmetric version, adapted for the BSS problem using prewhitened data vectors

$\mathbf{v}(t)$, is [21]

$$\begin{aligned}
\mathbf{z}(t) &= \mathbf{g}(\mathbf{W}(t-1)\mathbf{v}(t)) = \mathbf{g}(\mathbf{y}(t)), \\
\mathbf{h}(t) &= \mathbf{P}(t-1)\mathbf{z}(t), \\
\mathbf{m}(t) &= \mathbf{h}(t)/(\beta + \mathbf{z}^T(t)\mathbf{h}(t)), \\
\mathbf{P}(t) &= \frac{1}{\beta}\mathrm{Tri}\left[\mathbf{P}(t-1) - \mathbf{m}(t)\mathbf{h}^T(t)\right], \\
\mathbf{e}(t) &= \mathbf{v}(t) - \mathbf{W}^T(t-1)\mathbf{z}(t), \\
\mathbf{W}(t) &= \mathbf{W}(t-1) + \mathbf{m}(t)\mathbf{e}(t)^T. \qquad (10)
\end{aligned}$$

The forgetting constant $0 < \beta \leq 1$ should be close to unity. The notation Tri means that only the upper triangular part of the argument is computed and its transpose is copied to the lower triangular part. For clarity, we have used the time index t in the formulae (10), because quantities on the right hand side may depend either on the index value $t-1$ or t. However, also the NPCA-RLS rule is an adaptive one-step updating algorithm.

The main advantage of the NPCA-RLS algorithm (10) compared to the original nonlinear PCA algorithm is that the learning parameters are determined automatically from the input data so that they become roughly optimal [21, 27]. On-line estimation of kurtosis was added also to this algorithm for real-world data. The sequential version of the algorithm (10) is simpler but less accurate [27]. The NPCA-RLS type algorithms are discussed in more detail in [21, 22, 27].

Before proceeding, we point out that there exist various connections between the algorithms included in our comparison and also with other approaches, such as the maximum likelihood method for blind source separation or Bussgang methods for blind deconvolution; see [6, 22, 23] for further information.

4 Artificially generated data

[Figure 1 about here.]

4.1 Experimental arrangements

We began the comparison with artificial data generated from known sources, because only then the accuracy and convergence speed of the algorithms can be measured reliably. The experimental setup was the same for each algorithm in order to make the comparison as fair as possible. For the ICA/BSS algorithms, only the data vectors $\mathbf{x}(t)$ and the number m of sources were given as the known input data. Both sub-Gaussian and super-Gaussian sources were generated, and the mixing matrix $\mathbf{A}$ used in our simulations consisted of uniformly distributed random numbers. The general test arrangements are described in more detail in [10].

The accuracy was measured using two performance indexes. The first one, E_1, is defined by [1, 32]

$$E_1 = \sum_{i=1}^{m}(\sum_{j=1}^{m} \frac{|p_{ij}|}{max_k|p_{ik}|} - 1) + \sum_{j=1}^{m}(\sum_{i=1}^{m} \frac{|p_{ij}|}{max_k|p_{kj}|} - 1) \qquad (11)$$

where p_{ij} is the ij-th element of the $m \times m$ matrix $\mathbf{P} = \mathbf{BA}$. If the sources have been separated perfectly, $\mathbf{P}$ becomes a permutation matrix. 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. Clearly, the index (11) attains its minimum value zero for an ideal permutation matrix. The larger the value E_1 is, the poorer is the performance of a separation algorithm.

The second performance index E_2 is otherwise similar to E_1 and behaves fairly similarly, but it uses squared values of the elements of the matrix $\mathbf{P}$ instead of the absolute values in the formula (11).

[Figure 2 about here.]

4.2 Results

In the following, we report the results of our study using figures describing important characteristics of the algorithms.

Accuracy

First, we conducted a basic experiment measuring the accuracy of the tested algorithms. The number of sources (independent components) was in this series of experiments 10. All the sources were chosen super-Gaussian, because for this source type all the algorithms worked, including the Bell-Sejnowski (BS) algorithm. The sources used in this as well as in other experiments are given together with other details in [10]. For achieving statistical reliability, the experiment was repeated over 100 different realizations of the input data.

Figure 1 depicts the accuracy using statistical boxplots. For each of the 100 realizations, the accuracy was measured using the error index E_1, and the values of the index were arranged according to their goodness. The rectangular boxes correspond to the second and third quartile of the arranged index values. Thus these rectangles correspond to typical medium accuracy results occurring in 50% of the realizations. The horizontal line roughly in the middle of each rectangle is the median accuracy separating the second and third quartile for the corresponding algorithm. The larger interval described by two dashed vectical lines outside each box in figure 1 is regarded as the normal range of the error index. It was defined to be 1.5 times the height of the regtangle of each algorithm. The values outside this normal range are considered to be outliers, and are denoted by small crosses in figure 1.

Figure 1 shows that the natural gradient algorithm (ACY, WACY) achieves the best accuracy in this problem, but the Bell-Sejnowski algorithm (BS, ExtBS), symmetric fixed-point rule using the sigmoidal nonlinearity, and the NPCA-RLS algorithm are only slightly inferior. However, NPCA-RLS has more outliers, yielding occasionally poorer accuracy. The standard fixed-point algorithm (FP) has the poorest accuracy, though even the separation results given by it are usually adequate. This poorer accuracy is due to the cubic nonlinearity which weights heavily large values, and to the sequential nature of the algorithm, typically leading to propagation and growth of errors through the estimation sequence.

Computational requirements

Next, we studied the computational load of the chosen algorithms, measured in floating point operations. The stopping criterion for convergence is included in the program packages used for ExtBS and fixed-point algorithms. Roughly speaking, it is defined so that the change of the update term $\Delta\mathbf{W}$ must be below a predefined threshold. For the other algorithms, the required amount of computation for convergence was deduced from their learning curves. For example

Figure 3 shows that EASI has converged at about $1.5 \cdot 10^8$ Flops, ACY at about $2 \cdot 10^8$ Flops, and so on.

The experiments were similar as in studying the accuracy, so that the number of sources (independent components) was 10 and 100 realizations of the input data were used. Figure 2 shows a schematic diagram of the computational load vs. the accuracy of the outcome. The boxes typically contain 80% of the 100 trials, representing thus standard outcomes of a single experiment. Occasionally, poorer or better values may appear (cf. fig. 1).

Clearly, fixed-point algorithms require the smallest amount of computation. Of the adaptive algorithms, NPCA-RLS converges fastest, probably due to its roughly optimal determination of learning parameters. For the other adaptive algorithms, the learning parameter was a constant. Its value was determined by making some preliminary experiments so that a value providing good convergence was found. The natural gradient algorithm (ACY) and extended Bell-Sejnowski algorithm achieve a good final accuracy, but in the studied case of 10 sources, their computational load is about 20-50 times larger than for the fixed-point algorithms. The EASI algorithm is not among the best ones in any respect, though not the poorest on the other hand.

[Figure 3 about here.]

Convergence speed

The convergence speeds of the adaptive algorithms are compared in figure 3. Fixed-point algorithms do not appear in this comparison. The results shown are averages of 10 trials for 10 super-Gaussian source signals (for which all the algorithms worked without on-line estimation of kurtosis). The main observation is that the recursive least-squares version of the nonlinear PCA algorithm (NPCA-RLS) converges clearly fastest of the adaptive algorithms. The difference between NPCA-RLS and the other algorithms could probably be reduced by using simulated annealing or other more sophisticated technique for determining the learning parameters.

For sub-Gaussian sources, the results were qualitatively similar to those in figure 3, except that sometimes the EASI algorithm may converge even quicker than NPCA-RLS. However, sometimes its convergence speed was the poorest among the compared algorithms. Generally, a weakness of adaptive algorithms using stochastic gradients is that they are fairly sensitive to the choice of the learning parameters. A more detailed description of the behavior of EASI and the other algorithms can be found in [10].

Error for increasing number of sources

In figure 4, the error (square root of the error index E_2) is plotted as the function of the number of super-Gaussian sources. The results are median values over 50 different realizations of the input data. For more than 5 sources, the number of data samples was increased so that it was proportional to the square of the number of sources. Generally, the natural gradient algorithm (ACY) and its extended version (ExtBS) achieve the best accuracy, behaving very similarly as expected. The basic fixed-point algorithm (FP) using a cubic nonlinearity has the poorest accuracy, but its error increases only slightly after 7 sources. On the other hand, the version of the fixed-point algorithm which uses symmetric orthogonalization and tanh nonlinearity (FPsymth) performs as well as the natural gradient algorithm. For an unknown reason, the errors of the EASI and NPCA-RLS algorithms have a peak around 5-6 sources. For a larger number of sources, the accuracy of the NPCA-RLS algorithm is close to the best algorithms, while the error of EASI increases linearly with the number of source signals. However, the error of all the algorithms is tolerable for most practical purposes.

[Figure 4 about here.]

In experiments where the number of sub-Gaussian sources was increased, it was necessary to replace the cubic nonlinearity $g(t) = t^3$ with the more stable $g(t) =$ -tanh(t) in the EASI algorithm and with $g(t) =$ t - tanh(t) in the ACY algorithm to make them converge with more than 10 sources.

Effect of noise

We studied the effect of noise on performance of separation algorithms, too. To this end, we used the noisy ICA/BSS model

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \alpha\mathbf{n}(t), \tag{12}$$

where $\mathbf{n}(t)$ is a vector of Gaussian noise having the same variance as the source vector $\mathbf{s}(t)$, and the parameter α controlling the proportion of noise varied in the interval $[0.01, 0.2]$. The first conclusion was that the separation results degraded fairly smoothly until the noise power increased up to -20dB of the signal power. If the amount of noise is increased even more, it may happen that the algorithms studied are not able to separate all the sources. It should be noted that because of the blindness of the BSS problem, it is impossible to separate the Gaussian noise term $\mathbf{n}(t)$ from the sources [18]. In practice, noise smears the separated sources, turning the separation results almost useless if there is a lot of noise present.

Another observation was that once there is even a little noise present in the data, the error strongly depends on the condition number of the mixing matrix $\mathbf{A}$. This holds both for equivariant algorithms [4] which compute the separating matrix $\mathbf{B}$ directly and for algorithms employing prewhitening, except for NPCA-RLS which behaves differently. The condition number of a matrix describes how close to singularity it is. Figure 5 shows the correlation between the condition number of the mixing matrix and the quality of separation results as a function of the parameter α. It can be seen that when even a little of noise is added to the mixtures, the separation quality depends strongly on the quality of the mixing matrix, while there is no such dependence in the noiseless case $\alpha = 0$. The correlation in Figure 5 decreases when the amount of noise increases, because separation of sources becomes increasingly difficult.

[Figure 5 about here.]

5 Comparison methods for real-world data

For comparing the performance and properties of the studied ICA or BSS algorithms in more practical circumstances, we made experiments with three different real-world data sets. These data sets, namely crab data, satellite data, and MEG artefact data, will be briefly discussed in context with the respective results in the next section.

Because we are now dealing with real-world data, the assumptions made on the ICA model (1) may not hold, or hold only approximately. Furthermore, our goal for the first two data sets is to extract meaningful features from the data. This problem is different from blind source separation, and is very similar to projection pursuit.

In projection pursuit [9], the goal is to find one-dimensional projections of multidimensional data containing as much "interesting" structural information as possible for visualisation purposes. Usually directions onto which the projections of the data are as non-Gaussian as possible are considered interesting, because in a wide majority of directions the projected data tends to have almost Gaussian distribution. An information-theoretic argument for searching most non-Gaussian projections is that entropy is maximized by Gaussian distribution (for a fixed variance). Entropy measures the randomness and unpredictability of the data. Thus in the directions associated with the most non-Gaussian projections the data are least random, containing deterministic structural information. Therefore, we have earlier suggested the application of ICA to projection pursuit in [18, 20].

The outcome of an algorithm (for example the separating matrix $\mathbf{B}$) often provides interesting results for a subset of independent components only. The remaining independent components may be more or less useless. Thus we have to choose a method indicating which of the estimated independent components are the best ones for representing meaningful information. In the following we propose two different indices for ranking the independent components.

An index for projection pursuit

A first solution is to base the index on information-theoretic concepts, using the data only. For choosing the most non-Gaussian basis vectors of ICA, negentropy would be an appropriate classical measure [8]. However, its direct estimation turned out to be too sensitive to noise and to the tails of the distributions of the independent components, whereas the interesting structure lies mainly in the central part of the distribution. Therefore, we used an index developed from a more robust structural measure proposed by Friedman in [9].

This index is the estimated squared distance between the experimental cumulative distribution and Gaussian cumulative distribution, given by [10]

$$I_1(j) = \sum_{i=1}^{N} \left(q_j(i) - \frac{i}{N} \right)^2 . \tag{13}$$

The numbers $q_j(i)$ consist of the values $r_j(t)$, $t = 1, \ldots, N$, sorted in increasing order with the index value i. The values $r_j(t)$ are in turn defined by

$$r_j(t) = \Phi \left(\frac{y_j(t) - \bar{y}_j}{\sigma_{y_j}} \right) . \tag{14}$$

Here $y_j(t)$ is the estimated value of the j:th unordered independent component corresponding to the t:th data vector $\mathbf{x}(t)$, as explained in Section 2. The number $r_j(t)$ is the normalised value obtained from the Gaussian cumulative distribution function Φ, and $\bar{y}_j$ and σ_{y_j} are respectively the mean and the standard deviation of y_j. The larger the index $I_1(j)$ is, the more the respective estimated independent component y_j should contain structural information on the data. This index has yielded good results in selecting appropriate basis vectors of ICA for projection pursuit tasks in our experiments.

A separability index for classification

If the classification of the data vectors $\mathbf{x}(t)$ is known (for example for data sets designed for testing pattern classification methods), this information can be used for defining another performance index. A simple measure based on known classification is the ratio

$$I_2(\{c_i\}) = \frac{V_{intra}}{V_{inter}} \tag{15}$$

between the intra-class variance

$$V_{intra} = \sum_{i=1}^{C} \left(\frac{1}{N_i} \sum_{\mathbf{x} \in c_i} \|\mathbf{x} - \mathbf{m}_i\|^2 \right) \tag{16}$$

and inter-class variance

$$V_{inter} = \sum_{i=1}^{C} \|\mathbf{m}_i - \mathbf{m}\|^2. \tag{17}$$

In these formulas, C denotes the number of classes, and c_i is the set of the N_i data vectors $\mathbf{x}$ belonging to the i:th class. The mean of the vectors in the i:th class (in c_i) is denoted by $\mathbf{m}_i$, and $\mathbf{m}$ is the total mean of all the data vectors $\mathbf{x}$. The smaller the index I_2 is, the better the performance. A small value of I_2 implies that the data vectors are concentrated on tight clusters corresponding to the classes, and the distance between these clusters is relatively larger.

Still another way to compare the algorithms directly is to study their ability to find relevant independent components. Assume that we can associate a class with an independent component. The class c_i is said to be found, if after learning by some ICA algorithm one of the independent components is classified to c_i. Repeating the experiment with each algorithm sufficiently many times, one can estimate the probability of finding a class. This estimate is the ratio of the number of successful trials where the class is found over the total number of trials. This comparison method is less sensitive to the precision of the found components, and gives insight into the searching power of each ICA algorithm.

6 Results for real-world data

6.1 Crab data

The crab data set is taken from [28]; see also the Appendix. It consists of measurements taken from two species of crabs (for instance the body depth). The goal is to predict the species and if possible also the sex of an individual. The data contains 200 sample vectors, 50 belonging to each species and sex. A data vector contains five measurements from each single crab in the data set. The correct classification is available for checking purposes.

Characteristic to this data set is that it is possible to provide a good separation using a linear projection, but this projection is not easily found because of the general shape of the data. Therefore for example standard PCA fails in finding good separating projections. A drawback of the data set is that the number of available sample vectors, 200, is fairly small with respect to the statistical reliability of the assumptions and results.

[Figure 6 about here.]

After estimating the 5 independent components, the index I_1 was computed for each algorithm, and the two best independent components were chosen for visualising the crab data. Figure 6 shows the projections given by various ICA algorithms and also by the standard PCA for comparison purposes. All the ICA algorithms performed fairly well after some tuning of the nonlinearities [10]. Separation of the two species would be possible using a simple linear classifier. Note that the lines separating the species and somehow (depending on the algorithm) also the sexes would be in vertical and horizontal directions, because each independent component converges to its own separating projection. This is not the case with PCA.

Table 1 shows the numerical index values. Here, the two best independent components with respect to the index I_1 are denoted by IC_1 and IC_2. The index I_2 was computed for these components, too. It is interesting to note that although PCA yields a good, low value of the clustering index I_2, its results for the structural index I_1 are very poor (recall that for this index large values are better). This clearly indicates that PCA does not pay attention to the structure of the projection.

The ranking of the algorithms differs with indices I_1 and I_2, but is very similar to the one obtained with artificial data. EASI yields the poorest index values, maybe because it tries to do everything in one step, and confuses the sexes of the blue crab species. In this example, prewhitening using PCA provides a good basis for subsequent separation. It is noteworthy that the index values in Table 1 and the visual quality of the results in Figure 6 sometimes differ from each other, for example for the whitened natural gradient algorithm (WACY).

[Table 1 about here.]

6.2 Satellite data

The satellite image data is taken from the UCI Machine Learning Repository [24]. Each data point is a pixel from a Landsat image showing a part of Australia. There are four spectral bands, and each pixel is stored with its eight nearest neighbors. Thus the data vectors $\mathbf{x}(t)$ are $4 \times 9 = 36$ dimensional, and their total number is 6435. A classification in 6 soil types (red soil, cotton, grey soil, damp grey soil, very damp grey soil, and vegetation stubble) is provided, with the smallest class representing approximately 10% of the data.

First we computed the index I_2 for each independent component separately. The three best components were then selected, and the index I_2 (15) was computed for the 3-dimensional space spanned by these components. The dimension 3 was chosen after visual inspection of the data. Figure 7 shows the projection onto the 2-dimensional subspace separating the two first classes from the rest. The continuum of grey soil, damp grey soil, and very damp grey soil is visible but not separated. It is separated on a third component, though the boundaries between these somewhat vague classes are of course not strict. The ACY algorithm did not achieve good results, and as the ExtBS is in fact the same algorithm with whitening and some improvements, it was used instead.

[Figure 7 about here.]

The results shown in Table 2 indicate that the ExtBS algorithm performs best. The FPth algorithm achieves almost as good values, while NPCA-RLS is the most remote. A value above 1 in the ratio I_2 means that the average intra-class variance is bigger than the average inter-class variance. On the other hand, the version of ExtBS where the dimension reduction is performed using a rectangular ICA matrix fails completely, whereas EASI algorithm using a similar method yields acceptable results.

[Table 2 about here.]

One should note that in this problem standard PCA performs best using the criterion I_2. This is due to the shape of the raw data, where the classes are already clustered along the maximum variance directions. Hence ICA analysis of this data set gives only little new insights.

6.3 MEG artefact data

Magnetoencephalography (MEG) studies the brain using the magnetic field generated by populations of neurons. The data were collected from a 122-channel Neuromag device equipped with special sensors for the magnetic field. As this field is very weak, extra sources corresponding to various artefacts badly degrade the brain signal. It has turned out that ICA can separate artefacts from the actual brain signals [30], which is very difficult using other methods.

[Figure 8 about here.]

The data set [30] covered one second of recording at 297 Hz sampling frequency, and was filtered using a $0.3 - 45$ Hz cutoff frequency and then downsampled by a factor two. The data were then compressed using PCA to a dimension ranging from 7 to 12, and 7 to 11 independent components were retrieved. Figure 8 shows typical results given by the standard fixed-point algorithm, revealing some typical artefacts like eye movements, muscle activity, heartbeat, and a disturbance caused by a digital watch. Except for the watch signal which should be a square wave, all artefacts are retrieved with good precision.

From the different conditions and trials, a set of about 2400 separating vectors (row vectors of the separating matrix $\mathbf{B}$) was formed and labelled by hand. Then the probability of retrieval for a given class was computed for each algorithm. In Table 3, the results are summarised for each algorithm and artefact class. The abbreviations of the classes are explained in the caption of Figure 8; avg. denotes the average probability of retrieval.

The MEG data proved to be more difficult for the ICA algorithms, and some of them could not retrieve all the independent components in the data. The probabilities of standard fixed-point (FP) and NPCA-RLS algorithms are underestimated, because results with some suboptimal numbers of independent components are also taken into account in counting the probability. The FPsymth and ExtBS algorithms yield rather similar results, indicating again that they do much the same thing. The deflation version of the fixed-point algorithm has problems with the watch signal. The cubic nonlinearity used retrieves the sub-Gaussian signals last, and only by retrieving more than 9 independent components is it possible to see that signal. The noisy watch signal is lost more easily also due to orthogonalisation errors. It can be noted that none of the ICA algorithms is best for all the classes.

[Table 3 about here.]

7 Conclusions

In this paper, we have experimentally compared five prominent neural or semi-neural learning algorithms introduced for independent component analysis or blind source separation. To our knowledge, this is the first large-scale comparison of such algorithms. It should provide useful information especially to people interested in developing applications of ICA and BSS.

From the experiments made for studying various aspects of neural ICA algorithms, the following general conclusions can be made. The first conclusion is robustness of ICA. Even though the assumption of statistical independence is not strictly fulfilled, the algorithms converge towards a clear set of components (MEG data), or a subspace of components whose dimension is much smaller than the dimension of the problem (satellite data). This is a good characteristic encouraging the use of ICA as a general data analysis tool.

The second conclusion is that currently the best trade-off for estimating ICA is the fixed-point algorithm with symmetric orthogonalisation and tanh nonlinearity. This is because it provides similar results as the Bell-Sejnowski/natural gradient algorithm family, which is optimal with respect to minimising the mutual information, while the computational load is clearly smaller.

A general conclusion on the experiments with artificial data is that for designing an efficient ICA algorithm, one should split the problem into different parts. These include at least the following choices: the algorithm; the nonlinearity; and the control structure. Of course, the dependencies between these constituent parts must be taken into account. Such a design allows one to make a good practical compromise between efficiency, robustness, precision, and other relevant requirements of the problem at hand.

The experiments with real-world data, especially with the crab data, demonstrate that ICA is a useful tool in projection pursuit and feature extraction. In a recent paper by Girolami [12], an interesting comparison is made between ICA and generative topographic mapping (GTM) in projection pursuit. GTM [3] is a new, theoretically justified nonlinear mapping method resembling closely the self-organizing map, which in turn is a well-known and widely used nonlinear visualisation method. The results show that sometimes ICA performs clearly better than GTM. This is especially remarkable because ICA uses linear mapping while in GTM the mapping is nonlinear.

Appendix: Programs and data sets

The following list contains addresses of the data sets and programs or software packages used in this paper that are currently available in public domain on the web.

- **Crab data:** http://markov.stats.ox.ac.uk.
- **Satellite data:** See under Statlog in [24].
- **Extended BS algorithm:** http://www.cnl.salk.edu/~tewon/ica_cnl.html.
- **Fixed-point algorithm:** FastICA at http://www.cis.hut.fi/projects/ica/fastica/.
- **EASI algorithm:** http://sig.enst.fi/~cardoso/guidesepsou.html.

References

[1] S. Amari, A. Cichocki, and H. Yang, "A new learning algorithm for blind signal separation," in *Advances in Neural Information Processing Systems 8*, MIT Press, Cambridge, MA, 1996, pp. 757-763.

[2] A. Bell and T. Sejnowski, "An information-maximisation approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, pp. 1129-1159, 1995.

[3] C. Bishop, M. Svensen, and C. Williams, "GTM: The generative topographic mapping," *Neural Computation*, vol. 10, pp. 215-234, 1998.

[4] J.-F. Cardoso and B. Laheld, "Equivariant adaptive source separation," *IEEE Trans. on Signal Processing*, vol. 44, pp. 3017-3030, December 1996.

[5] J.-F. Cardoso, "Infomax and maximum likelihood for blind source separation," *IEEE Signal Processing Letters*, vol. 4, pp. 112-114, April 1997.

[6] J.-F. Cardoso, "Entropic contrasts for source separation," to appear as Chapter 2 in S. Haykin (ed.), *Adaptive Unsupervised Learning.*

[7] A. Cichocki and R. Unbehauen, "Robust neural networks with on-line learning for blind identification and blind separation of sources," *IEEE Trans. on Circuits and Systems-1*, vol. 43, pp. 894-906, November 1996.

[8] P. Comon, "Independent component analysis - a new concept?," *Signal Processing*, vol. 36, pp. 287-314, 1994.

[9] J. Friedman, "Exploratory projection pursuit," *J. of American Statistical Assoc.*, vol. 82, no. 397, March 1987, pp. 249-266.

[10] X. Giannakopoulos, "Comparison of adaptive independent component analysis algorithms," Dipl.Eng thesis made for EPFL, Switzerland, at Helsinki Univ. of Technology, Finland, 58 p. Available at `http://www.cis.hut.fi/~xgiannak/`.

[11] M. Girolami and C. Fyfe, "Generalized independent component analysis through unsupervised learning with emergent bussgang properties," *Proc. 1997 IEEE Int. Conf. on Neural Networks (ICNN'97)*, Houston, Texas, June 1997, pp.2147-2152.

[12] M. Girolami, "The latent variable data model for exploratory data analysis and visualisation," *Neural Processing Letters,* vol 8, 1998, pp. 27-39.

[13] A. Hyvärinen and E. Oja, "Simple neuron models for independent component analysis," *Int. J. Neural Systems*, vol. 7, 1996, pp. 671-687.

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

[15] A. Hyvärinen and E. Oja, "A fast fixed-point algorithm for independent component analysis," *Neural Computation*, vol. 9, 1997, pp. 1483-1492.

[16] C. Jutten and J. Herault, "Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1-10, July 1991.

[17] J. Karhunen and J. Joutsensalo, "Representation and separation of signals using nonlinear PCA type learning," *Neural Networks*, vol. 7, no. 1, pp. 113-127, 1994.

[18] J. Karhunen, "Neural approaches to independent component analysis and source separation," in *Proc. 4th European Symp. on Artificial Neural Networks (ESANN'96)*, Bruges, Belgium, April 1996, pp. 249-266.

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

[20] J. Karhunen, E. Oja, L. Wang, R. Vigario, and J. Joutsensalo, "A class of neural networks for independent component analysis," *IEEE Trans. on Neural Networks*, vol. 8, pp. 486-504, May 1997.

[21] J. Karhunen and P. Pajunen, “Blind source separation and tracking using nonlinear PCA criterion: a least-squares approach,” in *Proc. 1997 Int. Conf. on Neural Networks (ICNN’97)*, Houston, Texas, June 1997, pp. 2147-2152.

[22] J. Karhunen, P. Pajunen, and E. Oja, “The nonlinear PCA criterion in blind source separation: Relations with other approaches,” *Neurocomputing*, vol. 22, pp. 5-20, 1998.

[23] T.-W. Lee, M. Girolami, A. Bell, and T. Sejnowski, “A unifying information-theoretic framework for independent component analysis,” to appear in *Int. J. on Mathematical and Computer Modeling* (in press).

[24] C. Merz and P. Murphy, “UCI repository of machine learning databases, 1998”. Available at http://www.ics.uci.edu/~mlearn/MLRepository.html.

[25] E. Oja, “The Nonlinear PCA learning rule and signal separation – mathematical analysis,” *Neurocomputing,* vol. 17, pp. 25-45, September 1997.

[26] E. Oja, J. Karhunen, A. Hyvärinen, R. Vigario, J. Hurri, “Neural independent component analysis - approaches and applications,” in S. Amari and N. Kasabov (Eds.), *Brain-Like Computing and Intelligent Information Systems*, Springer-Verlag, Singapore, 1998, pp. 167-188.

[27] P. Pajunen and J. Karhunen, “Least-squares methods for blind source separation based on nonlinear PCA,” *Int. J. Neural Systems*, vol. 8, nos. 5 & 6, pp. 601-612, 1997.

[28] B. Ripley, *Pattern Recognition and Neural Networks.* Cambridge Univ. Press, Cambridge, UK, 1996.

[29] R. Vigário, “Extraction of ocular artifacts from EEG using Independent Component Analysis,” *Electroenceph. clin. Neurophysiol.*, vol. 103, 1997, pp. 395-404.

[30] R. Vigário, V. Jousmäki, M. Hämäläinen, R. Hari, and E. Oja, “Independent component analysis for identification of artifacts in magnetoencephalographic recordings,” in *Advances in Neural Information Processing Systems 10*, MIT Press, Cambridge, MA, 1998, pp. 229-235.

[31] L. Wang and J. Karhunen, "A unified neural bigradient algorithm for robust PCA and MCA," *Int. Journal of Neural Systems,* vol. 7, March 1996, pp. 53-67.

[32] H. Yang and S.-I. Amari, "Adaptive online learning algorithms for blind separation: maximum entropy and minimum mutual information," *Neural Computation*, vol. 9, pp. 1457-1482, 1997.

[33] L. Xu, "Least mean square error reconstruction principle for self-organizing neural-nets," *Neural Networks*, vol. 6, pp. 627-648, 1993.

[34] L. Xu, C. Cheung, H. Yang, and S. Amari, "Independent component analysis by the information-theoretic approach with mixture of densities," in *Proc. 1997 Int. Conf. on Neural Networks (ICNN'97)*, Houston, Texas, June 1997, pp. 1821-1826.

List of Figures

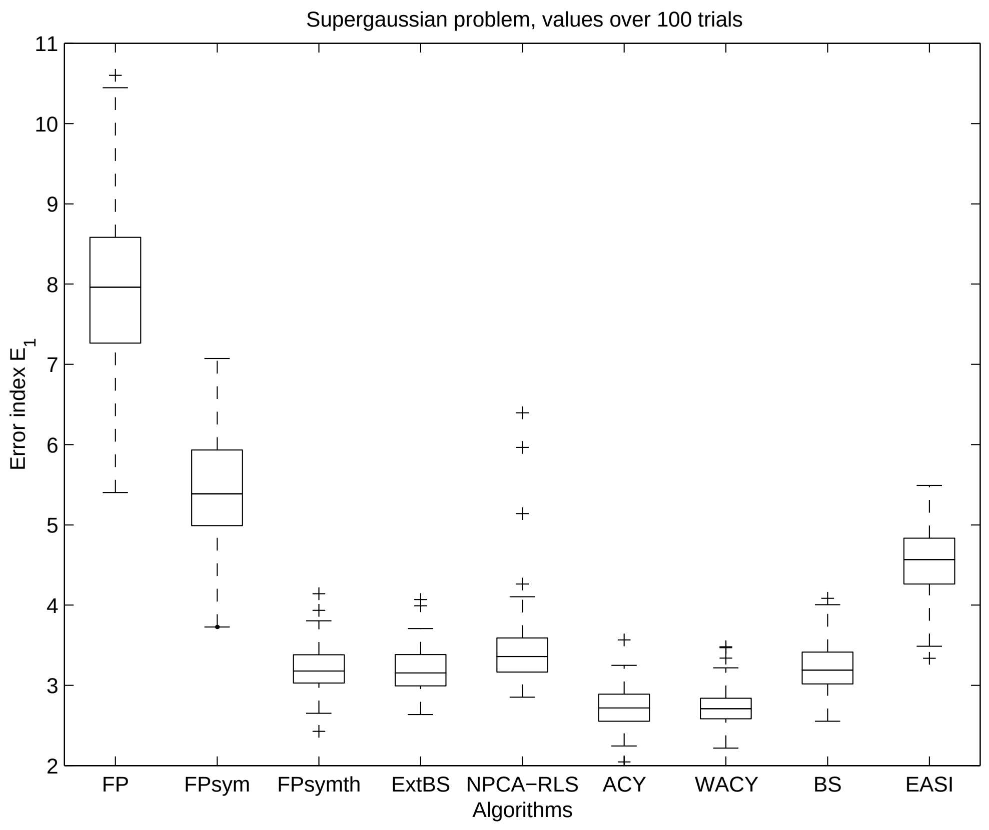

Figure 1: Accuracy of the algorithms in a problem of 10 super-Gaussian sources.

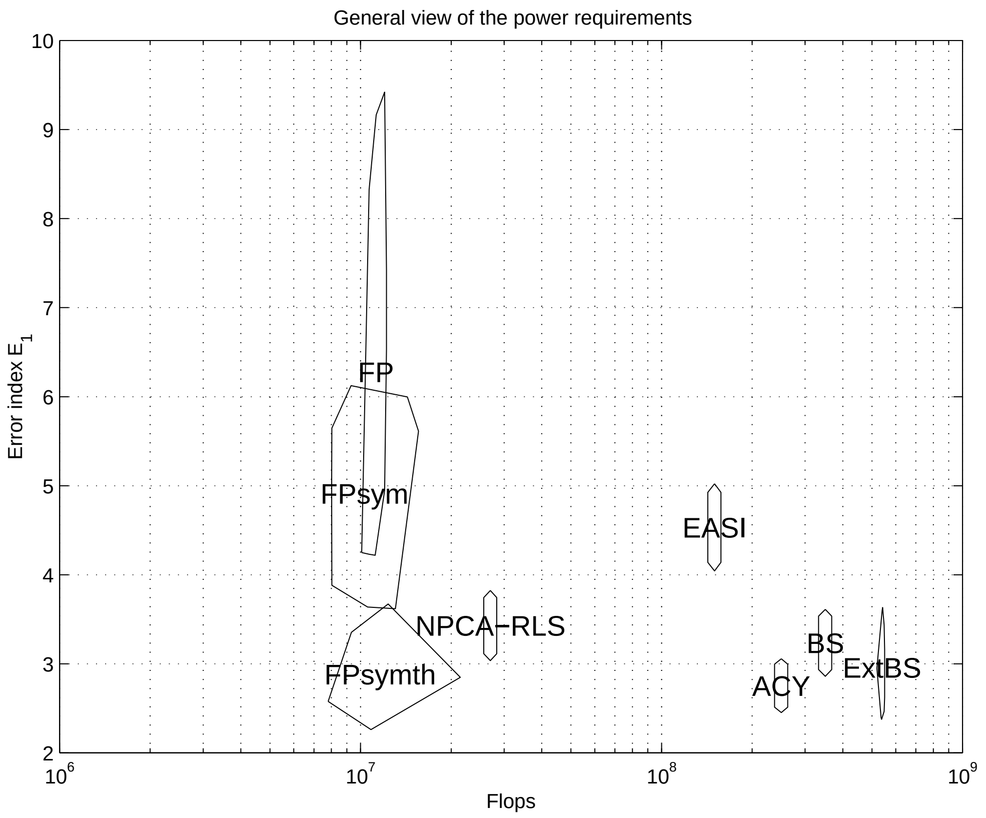

Figure 2: Computational requirements in flops versus the error index E_1.

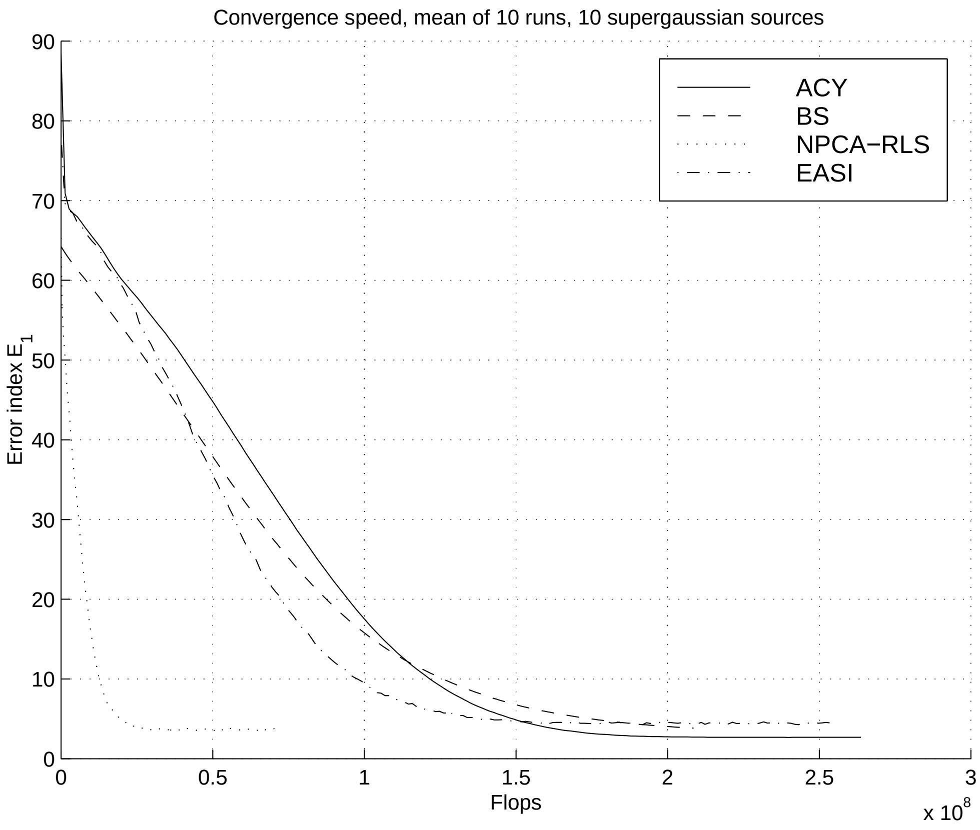

Figure 3: Convergence speed of adaptive BSS/ICA algorithms as a function of required floating-point operations for 10 super-Gaussian sources.

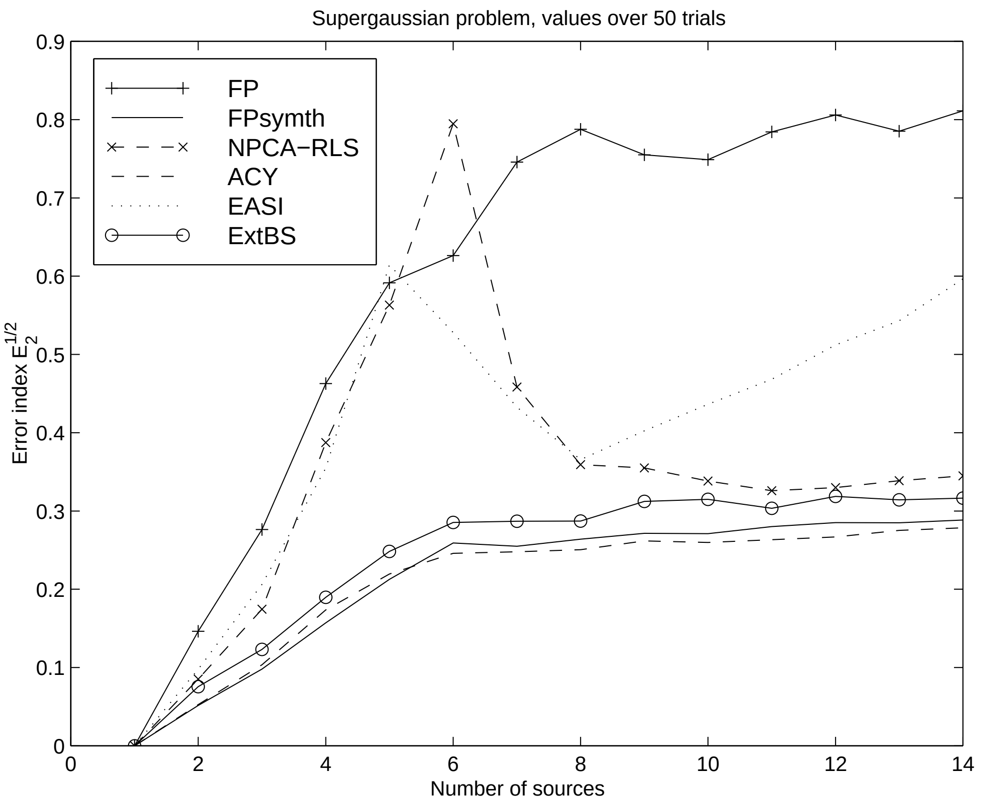

Figure 4: Error as a function of the number of sources.

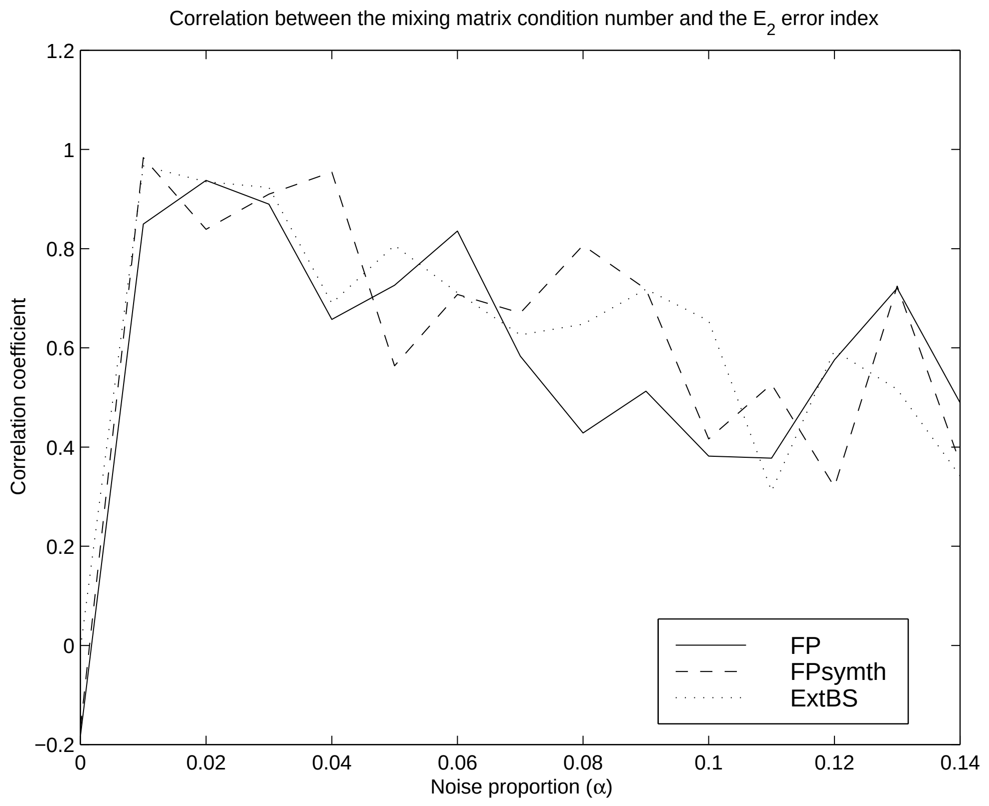

Figure 5: Correlation between the condition number of the mixing matrix and the quality of separation results as a function of noise.

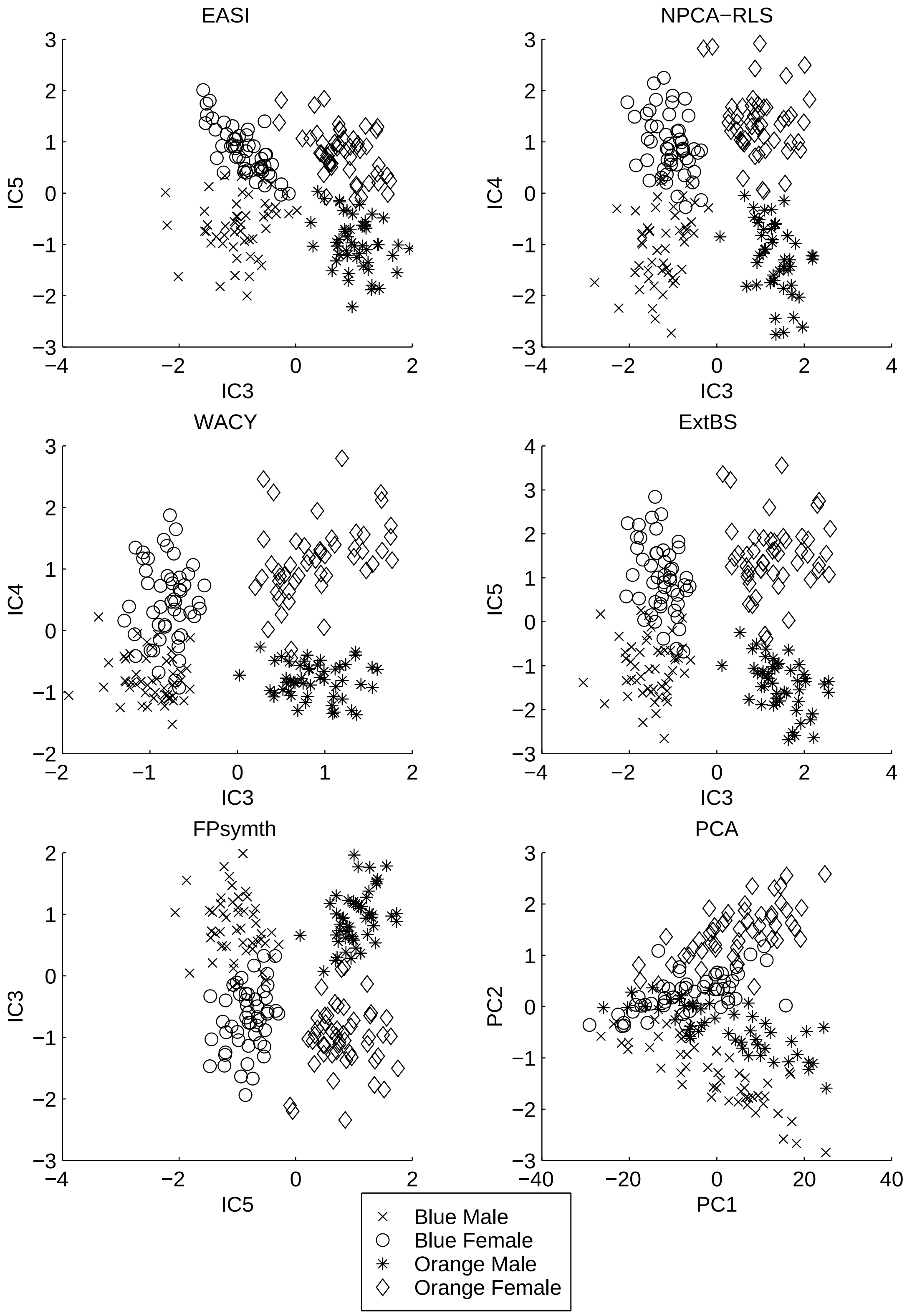

Figure 6: Separation of crab species and sexes using the best two ICA projections or standard PCA.

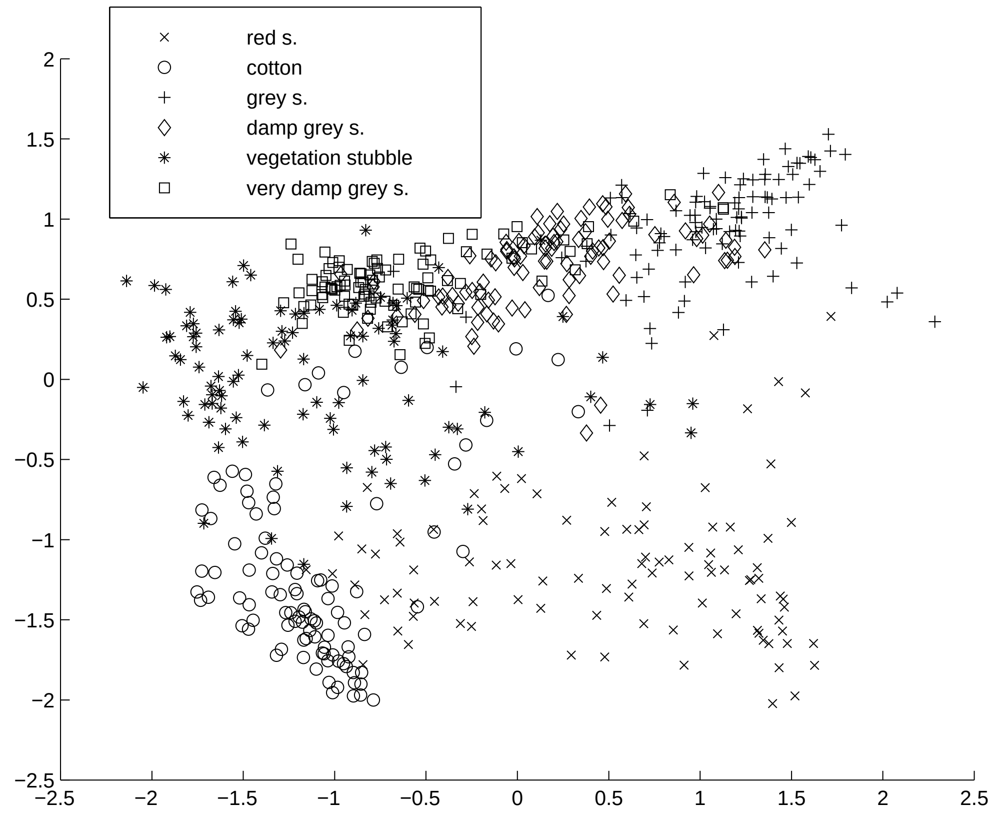

Figure 7: Two best independent components for satellite data given by the FPsymth algorithm. 100 elements from each class were taken at random.

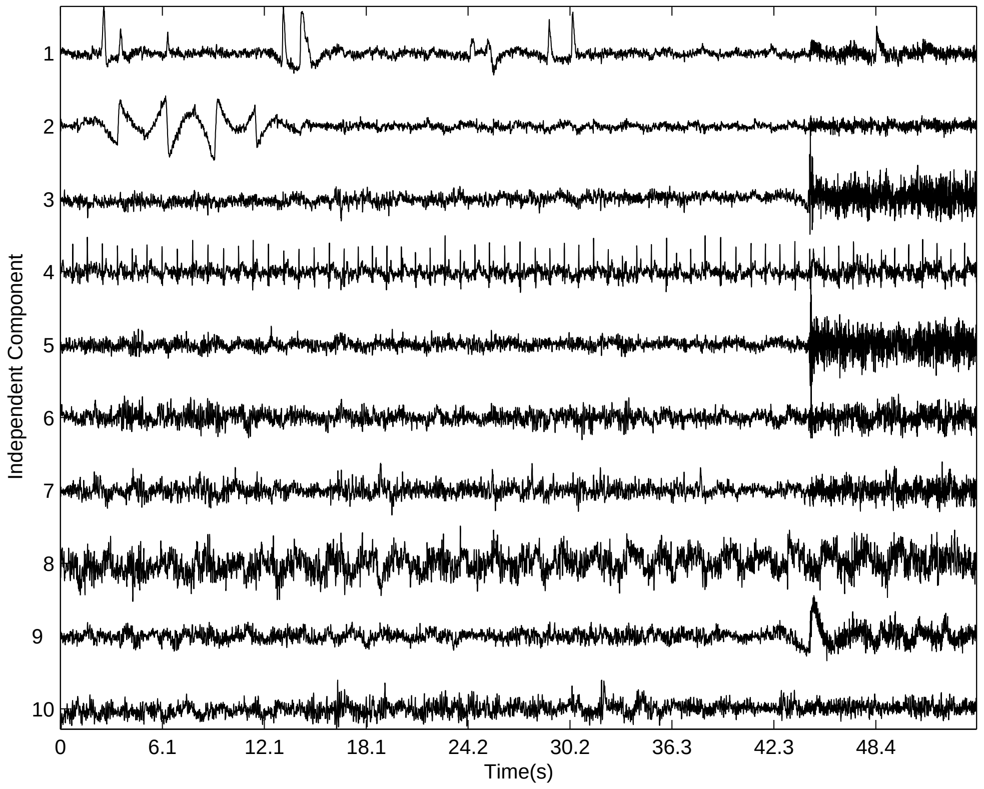

Figure 8: Independent components corresponding to various artefacts from MEG data. Results given by the FP algorithm. Explanations: 1. eye blinks (veog), 2. horizontal ocular movement (heog), 3. 5. and 9. muscular biting activity (musc), 4. heartbeat (card), 8. digital watch (watch), 10. visual activity (vis).

List of Tables

Algorithm	$I_1(IC_1)$	$I_1(IC_2)$	I_2
EASI	2.0229	0.91603	0.5717
NPCA-RLS	4.9095	1.588	0.29535
WACY	6.1064	3.7669	0.31682
ExtBS	5.9895	2.337	0.28358
FPsymth	5.218	1.8307	0.28988
PCA	0.86174	0.26039	0.34734

Table 1: Typical index values for the crab data.

Algorithm	I_2
EASI	1.1841
ExtBSic	5.008
ExtBSpc	0.48916
FPth	0.81401
NPCA-RLS	1.9435
PCA	0.47808

Table 2: Clustering values for the space spanned by the three best ICA vectors. Median value of several trials.

Algorithm	card	heog	veog	watch	musc1	musc2	musc3	vis	avg.
ExtBS	1	1	1	0.76	0.98	0.56	1	0.38	0.84
FP	1	1	1	0.2	1	0.68	0.98	0.35	0.77
FPsymth	1	1	1	0.78	1	0.56	1	0.42	0.85
NPCA-RLS	0.73	0.1	0.48	0.2	0.65	0.9	0.95	0.48	0.56

Table 3: Probability of retrieval of artefact classes. The value one means that there is at least one component that represents this class for every run of the algorithm.