A COMPARISON OF NEURAL ICA ALGORITHMS USING REAL-WORLD DATA

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

[1]IDSIA, Corso Elvezia 36, Lugano, Switzerland
[2]Laboratory of Computer and Information Science, Helsinki University of Technology
P.O. Box 2200, 02015 HUT, Espoo, Finland
Email: xavier@idsia.ch, Juha.Karhunen@hut.fi, Erkki.Oja@hut.fi

ABSTRACT

In this paper, we compare the performance of five prominent neural or semi-neural algorithms designed for Independent Component Analysis (ICA) using three different real-world data sets. The task is either to find interesting directions in the data for visualisation purposes or blind source separation. We develop criteria for selecting the most meaningful basis vectors of ICA or measuring the goodness of results. The comparison reveals characteristic differences between the studied neural ICA algorithms, complementing the results obtained earlier for artificially generated data.

1. INTRODUCTION

Independent Component Analysis (ICA) [4, 10] is an unsupervised technique which tries to represent the 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, 3, 11, 12, 16] have been introduced for ICA and applied to the closely related blind source separation (BSS) and other problems.

In developing applications, it is important to know the computational properties of available algorithms. This calls for an experimental comparison of the ICA algorithms. In a companion paper [7], we have presented a first comparison of neural ICA algorithms using artificially generated data. This is necessary, because only then the correct results are known. In this paper we apply the same ICA algorithms to three real-world data sets. Then 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.

2. ICA FORMULATION

We consider the standard linear data model used in ICA and BSS [10, 11, 2, 4]:

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

Here the m components $s_i(t)$, $i = 1, \ldots, m$, of the column vector $\mathbf{s}(t)$ 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 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 $n \geq m$. Its columns $\mathbf{a}_i$ are the basis vectors of ICA. At most one of the independent components $s_i(t)$ is allowed to be Gaussian.

For learning the ICA expansion (1), an $m \times n$ inverse or separating matrix $\mathbf{B}(t)$ is updated 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 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 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 [11].

3. NEURAL ICA OR BSS ALGORITHMS

In several neural ICA or BSS algorithms, the data vectors $\mathbf{x}(t)$ are preprocessed by whitening (sphering) them: $\mathbf{v}(t) = \mathbf{V}(t)\mathbf{x}(t)$. Here $\mathbf{v}(t)$ denotes the t:th whitened vector satisfying $\mathrm{E}[\mathbf{v}(t)\mathbf{v}(t)^T] = \mathbf{I}$, where $\mathbf{I}$ is the unit matrix, and $\mathbf{V}(t)$ is an $m \times n$ whitening matrix. Whitening can be done in many ways [11]. After prewhitening the subsequent $m \times m$ separating matrix $\mathbf{W}(t)$ can be taken orthogonal, which often improves the convergence. Thus in whitening approaches the total separating matrix is $\mathbf{B}(t) = \mathbf{W}(t)\mathbf{V}(t)$.

In this paper we concentrate on experimental results, describing the algorithms included in our study only briefly. For more details, see the references and [6].

Fixed-point (FP) algorithms. One iteration of the generalised fixed-point algorithm for finding a row vector $\mathbf{w}_i^T$ of $\mathbf{W}$ is [9]

$$\begin{aligned} \mathbf{w}_i^* &= \mathrm{E}\{\mathbf{v}g(\mathbf{w}_i^T\mathbf{v})\} - \mathrm{E}\{g'(\mathbf{w}_i^T\mathbf{v})\}\mathbf{w}_i \\ \mathbf{w}_i &= \mathbf{w}_i^*/\|\mathbf{w}_i^*\|. \end{aligned} \tag{3}$$

Here $g(t)$ is a suitable nonlinearity, typically $g(t) = t^3$ or $g(t) = \tanh(t)$, and $g'(t)$ is its derivative. The expectations are in practice replaced by their sample means. Hence the fixed-point algorithm is not a truly 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 [9, 6]. Usually the algorithm (3) converges after 5-20 iterations. The basic sequential version with tanh nonlinearity is denoted by **FPth**, the respective symmetric version by **FPsymth**.

Natural gradient algorithm (ACY). Originally proposed on heuristic grounds (see [3]), this popular and simple neural gradient algorithm was later on derived from information-theoretic criteria [16]. It is often called the Amari-Cichocki-Yang (ACY) algorithm. The update rule for the separating matrix $\mathbf{B}$ is

$$\Delta\mathbf{B} = \mu_k[\mathbf{I} - \mathbf{g}(\mathbf{y})\mathbf{y}^T]\mathbf{B}. \tag{4}$$

The notation $\mathbf{g}(\mathbf{y})$ means that the nonlinearity $g(t)$ is applied to each component of the vector $\mathbf{y} = \mathbf{Bx}$. The learning parameter μ_k is usually a small constant.

The basic algorithm (4) does not use prewhitening, which leads in many cases to a poor convergence. Therefore whitening is often applied to improve the convergence properties. The update rule (4) takes then the form

$$\Delta\mathbf{W} = \mu_k[\mathbf{I} - \mathbf{g}(\mathbf{y})\mathbf{y}^T]\mathbf{W} \tag{5}$$

where now $\mathbf{y} = \mathbf{Wv}$. This version is denoted by **WACY**.

Extended Bell-Sejnowski algorithm (ExtBS). This algorithm is also based on the update rule (5) but the following features have been added. First, kurtosis is estimated on-line for handling both super-Gaussian and sub-Gaussian sources simultaneously. This can be done by using the nonlinearity

$$\mathbf{g}(\mathbf{y}) = \mathrm{sign}(\mathbf{K}_4)\tanh(\mathbf{y}) + \mathbf{y} \tag{6}$$

where $\mathbf{K}_4$ denotes the kurtosis [8]. Thus the sign of the tanh term may vary depending on the kurtosis of the respective component of the output vector $\mathbf{y}$. Second, the learning parameter μ_k 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**.

EASI algorithm. Introduced as an adaptive signal processing algorithm in [2], EASI can as well be used in the neural network area for computing ICA. The general update formula for the total separating matrix $\mathbf{B}$ is in its normalized form [2]

$$\Delta\mathbf{B} = \mu_k \left[\frac{\mathbf{I} - \mathbf{y}\mathbf{y}^T}{1 + \mu_k \mathbf{y}^T\mathbf{y}} + \frac{\mathbf{y}\mathbf{g}(\mathbf{y}^T) - \mathbf{g}(\mathbf{y})\mathbf{y}^T}{1 + \mu_k|\mathbf{y}^T\mathbf{g}(\mathbf{y})|}\right]\mathbf{B} \tag{7}$$

On-line estimation of the kurtosis was added to the initial code for handling super-Gaussian and sub-Gaussian sources at the same time.

RLS algorithm for a nonlinear PCA criterion (NPCA-RLS). The basic symmetric version, adapted for the BSS problem using prewhitened data vectors $\mathbf{v}(t)$, is [12]

$$\begin{aligned} \mathbf{z}(t) &= \mathbf{g}(\mathbf{W}(t-1)\mathbf{v}(t)) = \mathbf{g}(\mathbf{y}(t)), \\ \mathbf{h}(t) &= \mathbf{P}(t-1)\mathbf{z}(t), \\ \mathbf{m}(t) &= \mathbf{h}(t)/(\beta + \mathbf{z}^T(t)\mathbf{h}(t)), \\ \mathbf{P}(t) &= \frac{1}{\beta}\mathrm{Tri}\left[\mathbf{P}(t-1) - \mathbf{m}(t)\mathbf{h}^T(t)\right], \\ \mathbf{e}(t) &= \mathbf{v}(t) - \mathbf{W}^T(t-1)\mathbf{z}(t), \\ \mathbf{W}(t) &= \mathbf{W}(t-1) + \mathbf{m}(t)\mathbf{e}(t)^T. \end{aligned} \tag{8}$$

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. The NPCA-RLS algorithm (8) is a recursive least-squares version of the nonlinear PCA algorithm [11]. The learning parameter is determined so that it becomes roughly optimal [12]. On-line estimation of kurtosis was added also to this algorithm.

4. COMPARISON METHODS

We have made experiments with crab data, satellite data, and MEG artefact data. These data sets will be briefly discussed in context with experimental results. 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 [5], 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. Therefore, we have earlier suggested the use of ICA for projection pursuit in [11].

Projection pursuit is a suitable task for comparison purposes, even if no ideal independent components exist. This is because we measure performance between the algorithms, and not between each algorithm and a perfect solution, which was possible with artificial mixtures. If some classification is available for the data, we use it for measuring the quality of the independent components, but no classification system has been appended to the ICA algorithms.

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 components are the best ones for representing meaningful information. In the following we propose two different indices for ranking the independent components.

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. However, its direct estimation turned out to be too sensitive to noise and to the tails of the distributions of 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 [5] to arrange the ICA basis vectors.

After some manipulations, this leads to the index [6]

$$I_1(j) = \sum_{i=1}^{N} \left(q_j(i) - \frac{i}{N} \right)^2 , \tag{9}$$

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

$$r_j(t) = \Phi \left(\frac{y_j(t) - \bar{y}_j}{\sigma_{y_j}} \right) . \tag{10}$$

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 contains structural information on the data. This index yielded good results in our projection pursuit experiments [6].

If the classification of the data vectors is known for example in pattern recognition tasks, this can be used for defining another performance index. A simple measure based on known classification is the ratio between intra-class variance and inter-class variance defined by

$$I_2(\{c_i\}) = \frac{\sum_{i=1}^{C} \left(\frac{1}{|c_i|} \sum_{\mathbf{x} \in c_i} \|\mathbf{x} - \mathbf{m_i}\|^2 \right)}{\sum_{i=1}^{C} \|\mathbf{m_i} - \mathbf{m}\|^2} \tag{11}$$

Here C is the number of classes, c_i is the set of data vectors belonging to the i:th class, $\mathbf{m_i}$ is the mean of the set c_i (data vectors of the i:th class), and $\mathbf{m}$ is the mean of all the data vectors. The smaller the index I_2 is, the better the performance.

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 in a run of some ICA algorithm one of the independent components gets classified to c_i. Running each algorithm sufficiently many times one can estimate the probability of finding a class. This estimate is the ratio of the number of runs that found the class over total number of runs. This comparison method is less sensitive to the precision of the found components and gives insight into the searching power of each algorithm.

5. RESULTS

5.1. Crab data

The crab data set is taken from [14]. It consists of measurements taken on the body of two species of crabs (for instance the body depth). The goal is to predict the species and eventually the sex of an individual. There are 5 measurements, and 200 sample vectors, 50 of each species and each sex. The correct classification is available for checking.

The advantage of 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 instance standard PCA fails in finding good separating projections.

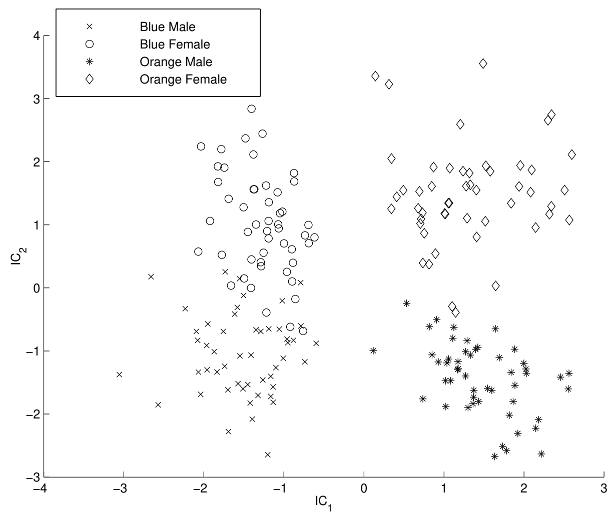

Figure 1: Separation of crab species and sex using the best two projections. Results given by the ExtBS algorithm.

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.

A drawback is that the number of available data vectors, 200, is fairly small, and statistical assumptions may not hold.

After estimating the 5 independent components the index I_1 was computed for each algorithm, and the two best components were chosen for visualisation (IC_1 and IC_2). The index I_2 was computed for these components, too. Figure 1 shows how the IC projections separate the classes. Separation of species would be possible through a linear classifier. Note that the separating lines would be in vertical and horizontal directions, because each component converges to its own separating projection. This is not the case with PCA.

Table 1 shows the results for each algorithm. It is interesting to note that although PCA yields good results with the clustering index I_2, the results for the structural index I_1 are very poor, showing clearly 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 in [7]. EASI yields the poorest results, maybe because it tries to do everything in one step. Prewhitening using PCA provides a good basis for subsequent separation.

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 value for the space spanned by the three best ICA vectors. Median value.

5.2. Satellite data

The satellite image data is taken from the UCI Machine Learning Repository [13]. It has been used in the Statlog project. 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 number is 6435. A classification in 6 soil types is provided, with the smallest class representing approximately 10% of the data.

First we computed the index (11) for each independent component separately. The three best components were then selected, and the index (11) was computed for the 3-dimensional space spanned by these components. The dimension 3 was chosen after visual inspection of the data. Figure 2 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. 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.

The results shown in Table 2 indicate that the ExtBS algorithm performed best, with FPth having close values, while NPCA-RLS is the most remote (a value above 1 in the ratio means that the average intraclass 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 the EASI algorithm using a similar method yields acceptable results.

One should note that in this problem standard PCA performs best using the criterion I_2. This is due to the same problem as with the crab data set, and also to

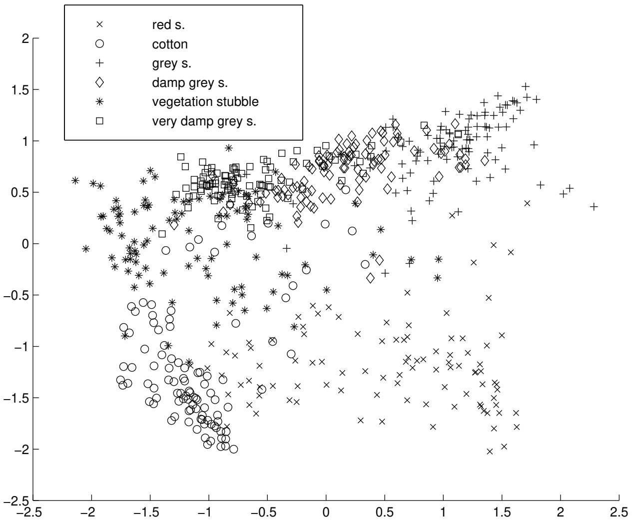

Figure 2: Two best independent components for satellite data. 100 elements from each class are taken at random.

the shape of the raw data, where the classes are already clustered along the maximum variance directions. This means also that ICA analysis of this data set gives only little new insights.

5.3. 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 SQUID sensors for magnetic field. As this field is very weak, extra sources corresponding to various artefacts degrade badly the brain signal. It has turned out that ICA can separate artefacts from the actual brain signals [15], which is very difficult using other methods.

We used the same data set as in [15]. This data set 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 compressed using PCA to a dimension ranging from 7 to 12, and 7 to 11 independent components were retrieved. Figure 3 shows typical results given by an ICA 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.

Out of 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.

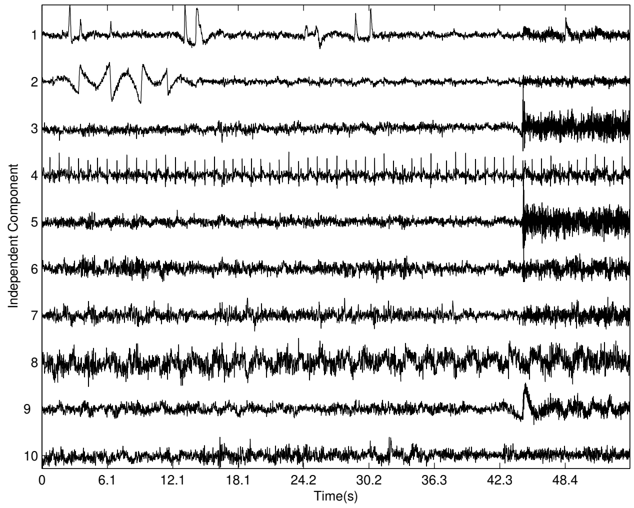

Figure 3: Independent components extracting artefacts from MEG data. Explanations: 1. eye blinks (veog), 2. horizontal ocular movement (heog), 3. 5. and 9. muscular biting activity (musc), 4. heartbeat (card), 8. digital watch (watch), 10. visual activity (vis).

The MEG data proved to be more difficult for the ICA algorithms, and some of them could not retrieve all the components in the data. The EASI algorithm had greatest difficulties, because did not have the stability reached via PCA compression. Its retrieval probability was in practice zero. The probabilities of 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

Algorithm	card	heog	veog	watch	
ExtBS	1	1	1	0.76	
FP	1	1	1	0.2	
FPsymth	1	1	1	0.78	
NPCA-RLS	0.73	0.1	0.48	0.2	
Algorithm	musc	mu.2	mu.3	vis	avg.
ExtBS	0.98	0.56	1	0.38	0.84
FP	1	0.68	0.98	0.35	0.77
FPsymth	1	0.56	1	0.42	0.85
NPCA-RLS	0.65	0.9	0.95	0.48	0.56

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

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 also lost more easily due to orthogonalisation errors. It can be noted that none of the ICA algorithms is best for all the classes.

6. CONCLUSIONS

From the experiments made for revealing various aspects of neural ICA algorithms, the following general conclusions can be made. The first conclusion is the stability of ICA. Even though the assumption of statistical independence is not 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 BS/ACY-family, which is optimal in terms of mutual information, while its computational load is clearly smaller [7].

7. APPENDIX: PROGRAMS AND DATA SETS

Except for the MEG data set that is not available in public domain, everything used in this paper is public and many parts are accessible on the web at the addresses given below.
- For the crab data, see http://markov.stats.ox.ac.uk.
- For the satellite data, see under Statlog in [13].
- For the Extended BS algorithm, see http://www.cnl.salk.edu/~tewon/ica_cnl.html.
- For the FP algorithm, see the FastICA package at http://www.cis.hut.fi/projects/ica/fastica/.
- For the EASI algorithm, see http://sig.enst.fi/~cardoso/guidesepsou.html.

8. REFERENCES

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

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

[3] 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*, 43(11):894–906, 1996.

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

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

[6] X. Giannakopoulos. Comparison of adaptive independent component analysis algorithms. Master's thesis, EPFL, March 1998. Made at Helsinki Univ. of Technology, Finland.

[7] X. Giannakopoulos, J. Karhunen, and E. Oja. An experimental comparison of neural ICA algorithms. In *Proc. Int. Conf. on Artificial Neural Networks*, Skövde, Sweden, 1998.

[8] M. Girolami and C. Fyfe. Generalised independent component analysis through unsupervised learning with emergent bussgang properties. In *Proc. IEEE Int. Conf. on Neural Networks*, pages 1788–1791, Houston, Texas, 1997.

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

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

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

[12] J. Karhunen and P. Pajunen. Blind source separation and tracking using nonlinear PCA criterion: A least-squares approach. In *Proc. IEEE Int. Conf. on Neural Networks*, pages 2147–2152, Houston, Texas, 1997.

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

[14] B. D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, UK, 1996.

[15] R. Vigário, V. Jousmäki, M. Hämäläinen, R. Hari, and E. Oja. Independent component analysis for identification of artifacts in magnetoencephalographic recordings. In *Neural Information Processing Systems 10.* MIT Press, 1998.

[16] H. Yang and S.-I. Amari. Adaptive online learning algorithms for blind separation: Maximum entropy and minimum mutual information. *Neural Computation*, 9(7):1457–1482, October 1997.