PERGAMON

Neural Networks 13 (2000) 597–611

Neural Networks

www.elsevier.com/locate/neunet

Blind signal processing by the adaptive activation function neurons

S. Fiori

Neural Networks and Adaptive System Research Group, Department of Industrial Engineering (DIE), University of Perugia, Perugia, Italy

Received 4 August 1999; accepted 5 April 2000

Abstract

The aim of this paper is to study an Information Theory based learning theory for neural units endowed with adaptive activation functions. The learning theory has the target to force the neuron to approximate the input–output transference that makes it flat (uniform) the probability density function of its output or, equivalently, that maximizes the entropy of the neuron response. Then, a network of adaptive activation function neurons is studied, and the effectiveness of the new structure is tested on Independent Component Analysis (ICA) problems. The new ICA neural algorithm is compared with the closely related 'Mixture of Densities' (MOD) technique by Xu et al.. Both simulation results and structural comparison show the new method is effective and more efficient in computational complexity.

Keywords: Adaptive signal processing; Adaptive activation function; Independent component analysis

1. Introduction

Unsupervised learning models based on Information Theory have become an important research field in the area of neural networks. Over the recent years, applications to optimal information transmission and preservation (Linsker, 1992; Plumbley, 1993) have been investigated, and problems like unsupervised probability density function (pdf) shaping and approximation (Linsker, 1989; Miller & Horn, 1998; Roth & Baram, 1996; Vapnik, 1997) with applications to linear estimation and time-series prediction have been solved by means of neural networks. Recently, density shaping by neural units has been studied in relation with Independent Component Analysis (ICA) (Bell & Sejnowski, 1996; Obradovic & Deco, 1997; Taleb & Jutten, 1997; Xu, Cheung & Amari, 1997; Xu, Cheung, Ruan & Amari, 1998a; Xu, Cheung, Yang & Amari, 1998b), Blind System Deconvolution (Bell & Sejnowski, 1996; Bellini, 1986; Fiori, Uncini & Piazza, 1998b), and Uniform Hashing (Alon & Orlitsky, 1996; Fiori, Bucciarelli & Piazza, 1998a; Majewski, Wormald, Havas & Czech, 1996).

In this paper, we deal with the problem of searching for a non-linear transformation $h(\cdot)$ between a random scalar process $x(t)$ and a transformed (*warped*) one, so that the pdf of the warped process becomes flat (uniform), by means of a neural system endowed with an information theory-based learning rule. In order to provide a suitable representation for $h(\cdot)$, we use here *adaptive activation function neural units*. These have been introduced by Chen and Chang (1996), Piazza, Uncini and Zenobi (1992), Piazza, Uncini and Zenobi (1993) and Vecci, Piazza and Uncini (1997) as they were noted to guarantee a sufficiently high degree of flexibility in some signal processing applications and reduced-complexity neural structures (as well as closely related structures like functional-link neural units (Mel, 1994; Pao, 1989; Rumelhart & McClellend, 1986; Zurada, 1992)) although their use was limited to supervised tasks, while preliminary studies about unsupervised learning in adaptive activation functions networks have recently appeared in Fiori (1999) and Fiori and Piazza (1998).

Here we deal with the problem of searching for the transformation $h(\cdot)$ when

1. the random process $x(t)$ is non-stationary and
2. its statistical and temporal features are unknown.

Finding this transformation is equivalent to solving a *Blind Signal Flattening* (BSF) problem, in that there is no information available about the source process $x(t)$. Moreover, it is clear that the adaptive activation function neuron learns in an unsupervised way in that there are no output targets defined. Otherwise, the solution of the associated stationary non-blind problem is known in a closed form as *Probability Integral Transformation* (PIT, Sudjianto & Hassoun, 1994). Formally, from Sudjianto and Hassoun (1994), it is recalled that the nonlinear transformation,

E-mail address: sfr@unipg.it (S. Fiori).

0893-6080/00/$ - see front matter
PII: S0893-6080(00)00044-7

which makes it flat the probability distribution of $x(t)$ is:

$$h^{\mathrm{opt}}(x) = \mathscr{P}\mathscr{I}\mathscr{T}\{x\} \stackrel{\mathrm{def}}{=} \int_{-\infty}^{x} p_x(\xi)\,\mathrm{d}\xi, \tag{1}$$

where, by definition, the pdf $p_x(\xi)$ must satisfy:

$$\forall x : p_x(x) \geq 0, \qquad \int_{-\infty}^{+\infty} p_x(\xi)\,\mathrm{d}\xi = 1. \tag{2}$$

Also, in this paper we suppose the pdf to be symmetric, i.e. $p_x(\xi) = p_x(-\xi)$. On the basis of the conditions (2) and definition (1), we gather that irrespective of the flattening, $h(\cdot)$ has to be non-decreasing and to range between 0 and 1. Clearly the adaptive activation function neurons employed here must be structured so that their input–output functions inherently meet these requirements. In Fig. 9, an example of neuron with adaptive activation function is depicted. The BSF algorithm relative to this neural topology will be called *WARP*.

As learning rule for the coefficients, we use the stochastic gradient optimization of the neuron output's *Shannon differential entropy*, which in general maximizes if and only if the pdf of the output $y(t)$ becomes uniform within its range. It is worth noting that, in general, the density of maximum entropy is the Gaussian; the uniform one has maximum entropy only when the variable has bounded support (e.g. $y \in [0, 1]$), which of course is the case here.

Although the blind signal flattening problem is relevant by itself, in this work, we apply the new WARP algorithm to ICA (Comon, 1994) in order to solve the challenging problem of separating out mixed independent signals when the inputs are mixed sub-Gaussian signals (i.e. which have negative kurtosis) *and* super-Gaussian signals (i.e. which have positive kurtosis).

2. Learning the polynomial's coefficients: The WARP algorithm

In this section, an adaptive activation function neural unit is studied. Formally, the input–output description of a pseudo-polynomial adaptive activation function unit may be for instance given as:

$$y = h(x) = \tfrac{1}{2} + \tfrac{1}{2}\mathrm{sgm}[q(x)], \tag{3}$$

where x represents the neuron's net input, y the neuron's output, $q(x)$ is a polynomial in x whose coefficients are adaptively changed through time according to an optimization principle, and $\mathrm{sgm}(\cdot)$ denotes a generic sigmoidal function, bounded between -1 and $+1$, continuously differentiable almost everywhere at least twice. The presence of the squashing function $\mathrm{sgm}(\cdot)$ is motivated by the following reasons: (i) we wish to take advantage of the well-known good approximation capability of the polynomials, which unfortunately are unbounded functions, (ii) as entropy optimization makes no sense for unbounded functions, a kind of limitation should be introduced. The observation that the required non-linearity $h(\cdot)$ generally has the shape of a sigmoid, led us to introduce a non-linear (bounded) sigmoidal function which inherently shapes $h(\cdot)$ and makes it a convenient representation of $h^{\mathrm{opt}}(\cdot)$, in this sense.

Since function $h(\cdot)$ has to approximate $h^{\mathrm{opt}}(\cdot)$, that by definition is a non-decreasing function, polynomial $q(x)$ should be non-decreasing within the range of x as well, thus $q'(x) \stackrel{\mathrm{def}}{=} \mathrm{d}q/\mathrm{d}x > 0$ almost everywhere. Condition $q'(x) > 0$ is surely fulfilled if $q(x)$ assumes the following expression:

$$q(x) = a_0 + \sum_{i=0}^{r} a_{2i+1}^2 x^{2i+1}, \tag{4}$$

where $2r + 1$ is the order of the polynomial, $a_0 \in \mathbb{R}$ and $a_{2i+1} \in \mathbb{R}$ are free parameters. Note that the true coefficients of $q(x)$ are a_{2i+1}^2, which are always non-negative, and a_0, whose sign may be arbitrary. A schematic of the adaptive activation function neural unit that implements Eq. (3) is depicted in Fig. 9.

2.1. The proposed learning theory

Entropy H_y that has to be *maximized* w.r.t. a_0 and any a_{2i+1} is defined as:

$$H_y \stackrel{\mathrm{def}}{=} -\int_{\mathbb{R}} p_y(y)\log p_y(y)\,\mathrm{d}y, \tag{5}$$

where $p_y(y)$ is the pdf of $y(t)$; the dependence upon coefficient a_{2i+1} is understood. The entropy can be expressed in terms of $p_x(x)$ by recalling that $p_y = p_x/\psi$, $\psi \stackrel{\mathrm{def}}{=} |(\mathrm{d}h/\mathrm{d}x)|$. By using the above expressions, direct calculations show that

$$-H_y = -H_x - E_x[\log \psi]. \tag{6}$$

In order to maximize H_y, the Gradient Steepest Ascent (GSA) technique can be employed:

$$\Delta a_0 = +\eta\frac{\partial H_y}{\partial a_0}, \Delta a_{2i+1} = +\eta\frac{\partial H_y}{\partial a_{2i+1}}, \tag{7}$$

where η is a positive learning stepsize and $i = 0, 1, 2, \ldots, r$. The following quantities are therefore needed:

$$\frac{\partial \log\psi}{\partial a_0} = \frac{1}{\psi}\frac{\partial\psi}{\partial a_0}, \frac{\partial \log\psi}{\partial a_{2i+1}} = \frac{1}{\psi}\frac{\partial\psi}{\partial a_{2i+1}}.$$

In this section, we choose

$$\mathrm{sgm}(u) = \mathrm{th}(u) \stackrel{\mathrm{def}}{=} \frac{\mathrm{e}^{-u} - \mathrm{e}^{+u}}{\mathrm{e}^{-u} + \mathrm{e}^{+u}}.$$

Later, a different sigmoidal function will be introduced, which proves to be more interesting from a computational complexity point of view.

With some mathematical work, the following expression

$(i = 0, 1, \ldots, r)$ is obtained:

$$\frac{\partial \psi}{\partial a_{2i+1}} = -2\left\{ \frac{\partial q'(x)}{\partial a_{2i+1}} - 2[2h(x) - 1]\frac{\partial q(x)}{\partial a_{2i+1}} q'(x) \right\} \times [h(x) - 1]h(x). \quad (8)$$

From Eq. (4), for $0 \leq i \leq r$ direct calculations give

$$q'(x) = \sigma(x) \stackrel{\text{def}}{=} \sum_{i=0}^{r} (2i + 1)a_{2i+1}^2 x^{2i}, \quad (9)$$

$$\frac{\partial \sigma(x)}{\partial a_{2i+1}} = \mu_{2i+1}(x) \stackrel{\text{def}}{=} 2(2i + 1)a_{2i+1}x^{2i}, \quad (10)$$

$$\frac{\partial q(x)}{\partial a_{2i+1}} = 2a_{2i+1}x^{2i+1} = \frac{x}{2i + 1}\mu_{2i+1}(x).$$

The learning phase of the polynomial's coefficients is therefore governed by the following equations:

$$\begin{cases} \Delta a_{2i+1} = \eta\left[\frac{1}{\sigma(x)} - \frac{2}{2i + 1}x(2y - 1) \right]\mu_{2i+1}(x), \\ \Delta a_0 = -2\eta(2y - 1), \\ y = \frac{1}{2} + \frac{1}{2}\text{th}\,[q(x)], \end{cases} \quad (11)$$

where i ranges from 0 to r fixed, and where functions $\sigma(x)$ and $\mu_{2i+1}(x)$ are defined as in Eqs. (9) and (10), respectively, and $q(x)$ is given by Eq. (4).

Note that it is advisable to choose the starting points a_0^{init} a_{2i+1}^{init} different from zero because for such a choice the algorithm (11) gets stuck.

2.2. Discussion on WARP

About the learning theory for the proposed architecture, it could be interesting to consider the following observations:

1. Let us assume in Eq. (11) that $r = 0$ and $a_0 = 0$. In this way, $q(x) = a_1^2 x$, $\sigma(x) = a_1^2$ and $\mu_1(x) = 2a_1$, therefore algorithm (7) reduces to:
$$\Delta a_1 = 2\eta\left[\frac{1}{a_1} - 2a_1 x(2y - 1) \right]. \quad (12)$$
This adaptation rule recalls the one proposed by Bell and Sejnowski (1996) that Eq. (7) represents a generalization of Eq. (12) should be thought of as a learning rule for a linear neuron described by the input–output transference $z = a_1^2 x$ (where a_1^2 is the connection strength) that tries to maximize $H[\text{th}(z)]$. Note that the rule tries to keep $a_1 \neq 0$ avoiding the trivial solution $y(t) \equiv 0$.
2. The hypothesis that the pdf is symmetric might be easily relaxed by varying the structure of the adopted polynomial accordingly. A possible replacement of the expression (4) could be $q(x) = a_0 + \sum_{i=0}^{r} a_{2i+1}^2 (x - \chi_i)^{2i+1}$, with the $\{a_k\}$ and $\{\chi_k\}$ both being adaptive coefficients.
3. In order to further generalize our approach, it should be observed that coefficients a_{2i+1} must not necessarily be warped by a parabolic function; generally they can appear in Eq. (4) within a nonlinear function $\lambda(\cdot)$ at least non-negative and continuously differentiable almost everywhere at least once. Some examples of the suitable choices have been given by Fiori and Piazza (1998).

3. MOD technique for maximum entropy flattener approximation

Following Xu et al. (Xu et al., 1997, 1998a,b), the approximating function $\psi(x)$ can be assumed to have the following form:

$$\psi(x) \stackrel{\text{def}}{=} \sum_{j=1}^{n} \alpha_j \beta(\xi_j), \qquad \xi_j \stackrel{\text{def}}{=} b_j(x - c_j), \quad (13)$$

where $\beta(\cdot)$ is called the *basis function*, $b_j \in \mathbb{R}$ are called *slopes*, $c_j \in \mathbb{R}$ are termed *centers*, and $\alpha_j \in \mathbb{R}$ are termed *weights*; the integer number n determines the dimension of the basis and the approximation flexibility degree.

Coefficients α_j are interpreted as probability weights: each α_j plays the role of the probability that the term $\beta(\xi_j)$ gives a contribution to the representation, hence it is required that they satisfy the restrictions $\alpha_j > 0$, $\sum_{j=1}^{n} \alpha_j = 1$.

Following Xu et al. (1997, 1998a), let us now introduce an auxiliary function $\phi(\cdot)$ such that $\beta(\xi_j) = b_j\phi'(\xi_j)$ and assume $\phi(u) = 1/[1 + \exp(-u)]$. Also, in order to enforce parameters α_j to fulfill the prescribed restrictions, the following standard transformation (softmax) is used:

$$\alpha_j = \frac{\exp(\varrho_j)}{\sum_k \exp(\varrho_k)}, \quad (14)$$

where parameters $\varrho_j \in \mathbb{R}$ are new unconstrained variables used instead of the true weights α_j.

Now the entropy (5) depends upon the $3n$ coefficients (c_j, b_j, ϱ_j), which have to be adaptively changed in order to maximize $H_y = H_{h(x)}$. In formulas we get:

$$\Delta c_i = +k_c \frac{\partial H_y}{\partial c_i}; \qquad \Delta \varrho_i = +k_\varrho \frac{\partial H_y}{\partial \varrho_i}; \quad (15)$$

$$\Delta b_i = +k_b \frac{\partial H_y}{\partial b_i}; \qquad i = 1, \ldots, n, \quad (16)$$

where k_c, k_b and k_ϱ are positive real numbers. Writing the derivatives of ψ is much easier if new variables v_j are introduced as $v_j = v_j(x) \stackrel{\text{def}}{=} \phi[b_j(x - c_j)]$. In fact,

$$\psi(x) = \sum_{j=1}^{n} \alpha_j b_j (v_j - v_j^2). \quad (17)$$

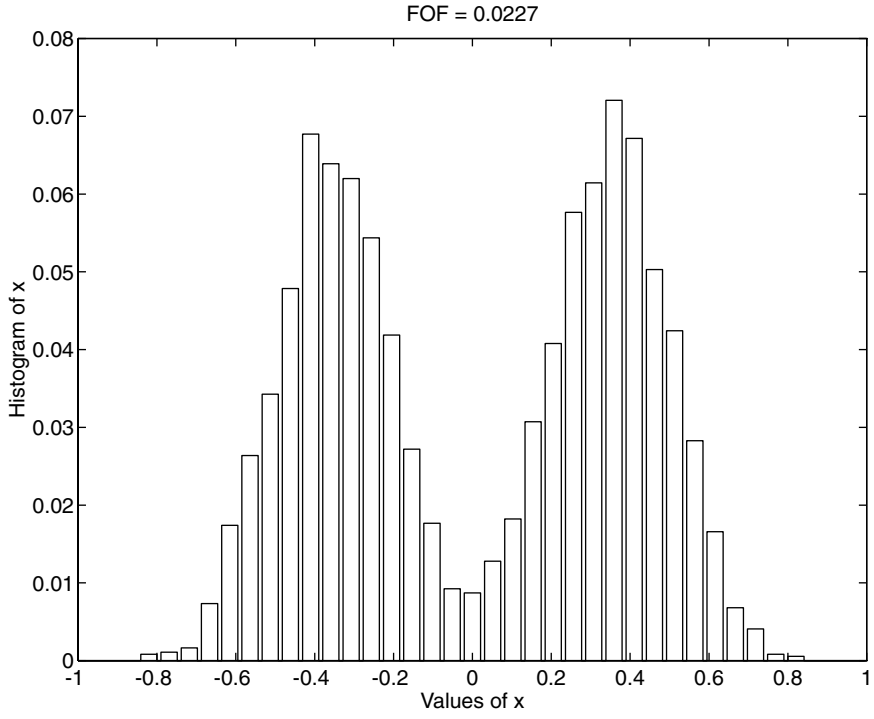

Fig. 1. Bi-Gaussian input histogram.

Direct calculations show that

$$\frac{\partial \psi}{\partial c_i} = -\alpha_i b_i^2 (1 - 2v_i)(v_i - v_i^2), \tag{18}$$

$$\frac{\partial \psi}{\partial b_i} = \alpha_i [1 + b_i (x - a_i)(1 - 2v_i)](v_i - v_i^2), \tag{19}$$

$$\frac{\partial \psi}{\partial \varrho_i} = \sum_{j=1}^{n} b_j (v_j - v_j^2) \alpha_j (\delta_{ij} - \alpha_i), \tag{20}$$

where δ_{ij} is the Kronecker's 'delta'.

The MOD-BSF learning algorithm can be summarized as follows:

1. On the basis of the old values of the centers c_i, the slopes b_i and the free weights ϱ_i ($i = 1, \ldots, n$) evaluate v_j, α_j (Eq. (14)) ($j = 1, \ldots, n$) and $\psi(x)$ (Eq. (17)).
2. Evaluate derivatives (18), (19) and (20).
3. Update coefficients c_i, b_i and ϱ_i by means of the learning equations:

$$\Delta c_i = \frac{k_c}{\psi} \frac{\partial \psi}{\partial c_i}, \qquad \Delta b_i = \frac{k_b}{\psi} \frac{\partial \psi}{\partial b_i}, \qquad \Delta \varrho_i = \frac{k_\varrho}{\psi} \frac{\partial \psi}{\partial \varrho_i}.$$

4. Tests on WARP and MOD

In support of the new statistical function approximation technique, in this section, some simulation results on WARP algorithm are reported and discussed, then a comparison of WARP and MOD algorithms is presented. The learning parameters have been chosen by performing several simulations and taking the values that granted a good trade-off between the convergence speed and numerical stability of the algorithms (also see Fiori et al., 1998a). Furthermore, to measure their approximation capability a *Figure Of Flatness* (FOF) has been defined as $FOF \stackrel{\text{def}}{=} \sum_{i=1}^{N} \epsilon_i^2$, where ϵ_i is the error pertaining to the ith discrete level of the output pdf compared with a uniform one (whose value is $1/N$); here $N = 40$. The more FOF approaches 0, the more an algorithm is performing well.

4.1. Computer simulations on WARP

The proposed WARP algorithm has been tested by using input signals having three kinds of pdf: Gaussian, bi-Gaussian (i.e. the superposition of two overlapping Gaussian distributions with same variance and different mean values) and bi-constant. The method works properly with all of them. Here we present results for *bi-Gaussian* and *bi-constant*. Simulations have been performed with the following data: $\eta = 0.04$, $r = 1$ and $a_k^{\text{init}} = 0.01$, $k = 0, 1, 3$.

In Fig. 1, a bi-Gaussian input's histogram can be seen and its FOF measure, while in Fig. 2 the corresponding output pdf is depicted. This graph has been obtained after 3000 samples. The final FOF of the output process should be

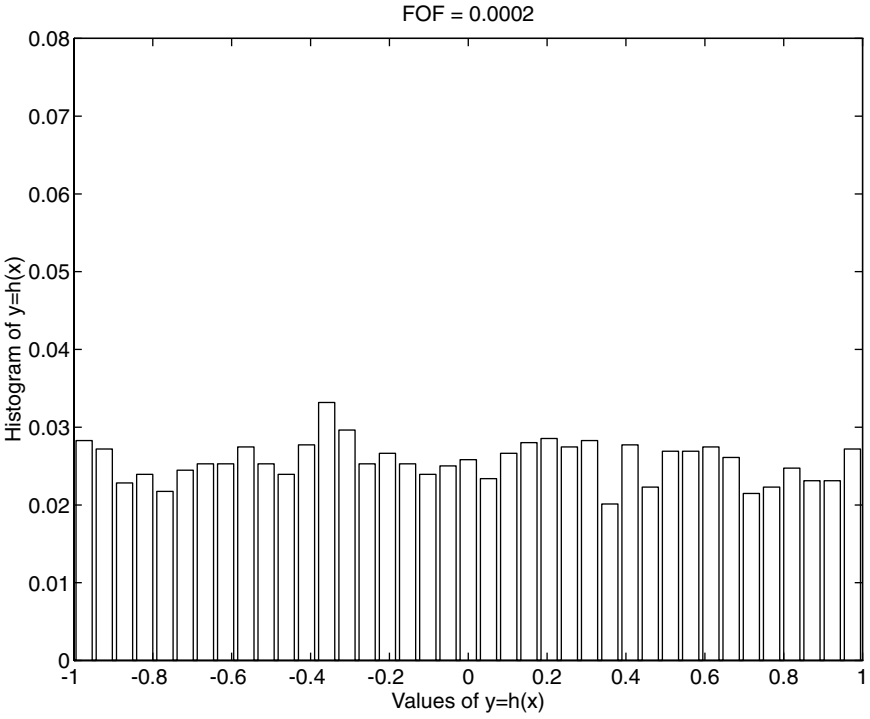

Fig. 2. Warped bi-Gaussian (3000 samples) histogram.

compared with the FOF of the input process. This simulation gives that the *Relative FOF* (i.e. the ratio between the FOF of the source signal and the FOF of its warped version, that has to be the greatest than possible) is $0.0227/0.0002 \cong 113.5$, thus the algorithm behaved satisfactorily.

Figs. 3 and 4 depict instead the input pdf and the output pdf (after 3000 samples) obtained for a bi-constant input.

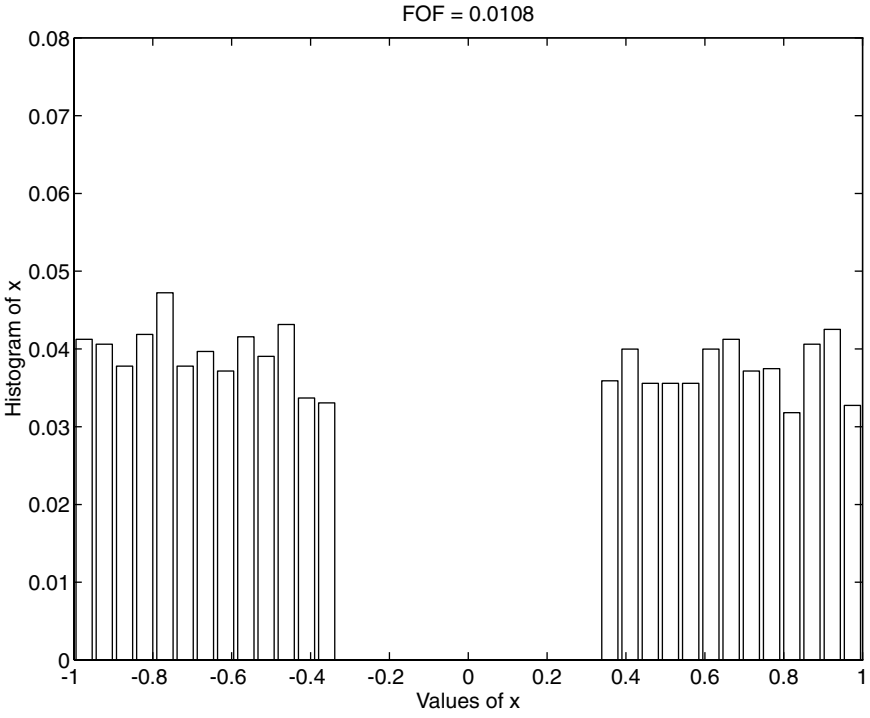

Fig. 3. Bi-constant input histogram.

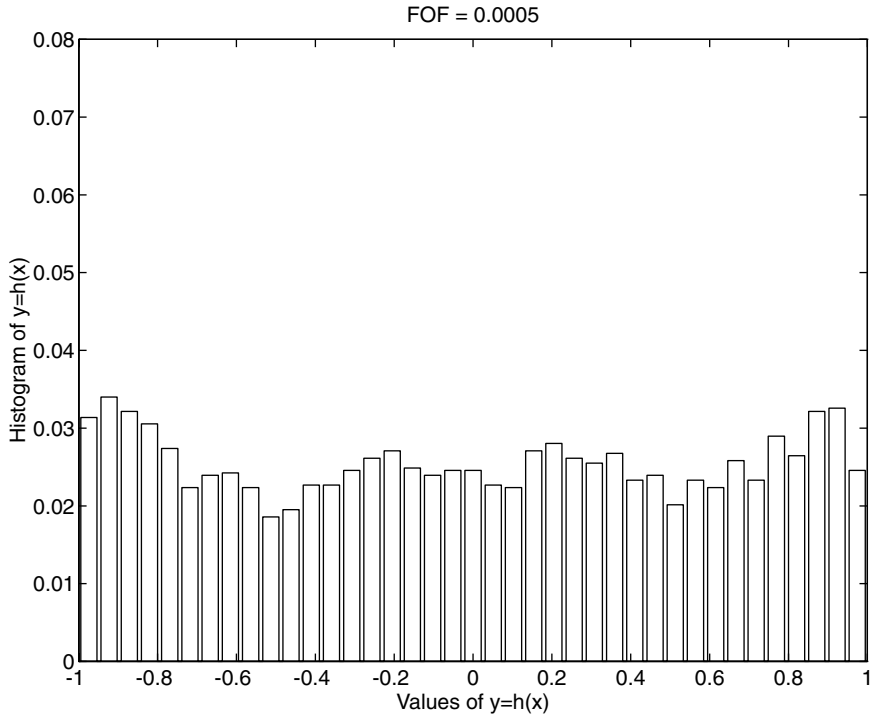

Fig. 4. Warped bi-constant (3000 samples) histogram.

Here the relative FOF is 0.0108/0.0005 ≅ 21.6. The pdf corresponding to the bi-constant signal is the most difficult one to be flattened.

Both in bi-Gaussian and bi-constant cases, output pdfs become quickly nearly flat (after 2000, 3000 samples).

4.2. Numerical comparison of WARP and MOD

Three kinds of input signals have been used for comparing the WARP and MOD algorithms: bi-Gaussian, bi-constant and real-world (speech) data. The learning

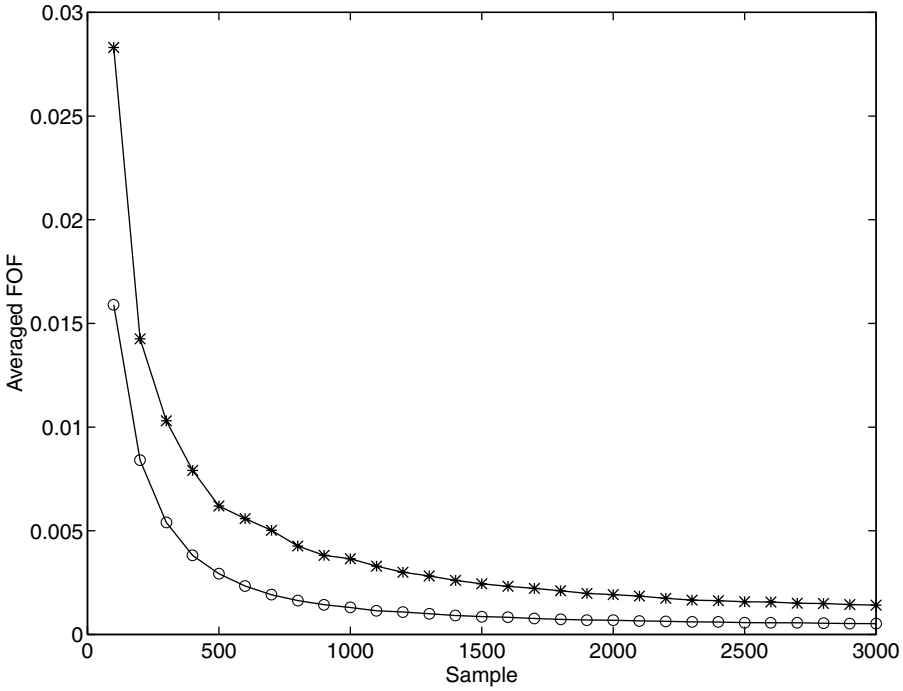

Fig. 5. WARP (circle)/MOD (star), bi-Gaussian data.

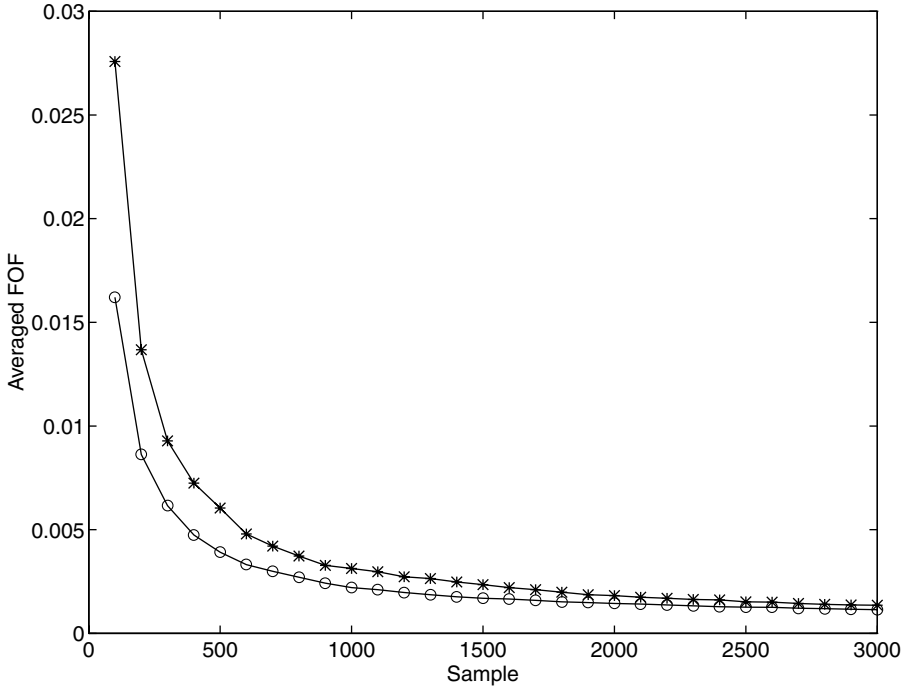

Fig. 6. WARP (circle)/MOD (star), bi-constant data.

parameters are the same as in the previous section for WARP and $n = 2$, $k_\varrho = 0.04$, $k_c = 0.04$, $k_b = 1$ for MOD.

Fig. 5 shows the FOF averaged over 50 bi-Gaussian data sets computed for every 100 samples. The WARP algorithm appears to be more accurate than the MOD. In Fig. 6, the FOF averaged over 50 bi-constant data sets is depicted.

As a further example, we tested the algorithms with a 22 kHz sampled speech signal. In this case, we chose $\eta = 0.02$, $k_\varrho = 0.01$, $k_a = 0.01$, $k_b = 0.7$, $n = 4$. Fig. 7 depicts the FOF versus the number of samples. Other numerical examples were reported by Fiori and Piazza (1998), along with the behavior of the polynomial's coefficients during the learning phase.

A close examination of the obtained results shows that the

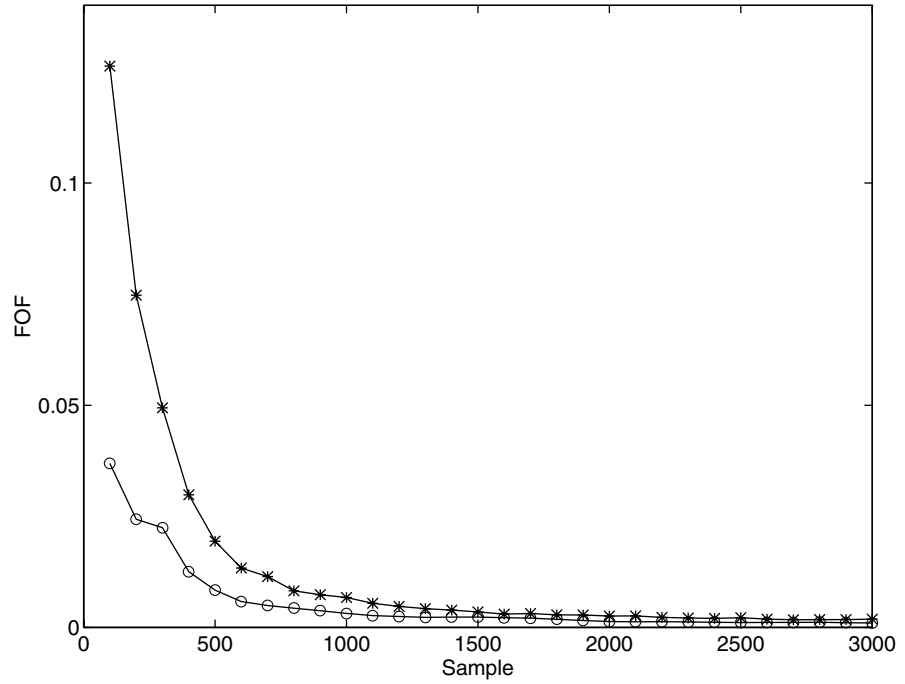

Fig. 7. WARP (circle)/MOD (star), speech data.

Table 1
Complexity of MOD and WARP

Algorithm	Multiplications	Divisions	Non-linearities	Parameters
MOD	$14n$	$5n+1$	$2n$	$3n$
WARP	$11r+9$	$r+2$	0	r

new WARP algorithm may have the same performances of the MOD-like convergence speed, steady-state precision and approximation capability.

4.3. Complexity comparison of WARP and MOD

A direct comparison between WARP and MOD technique shows that the adaptive activation function neuron needs r coefficients to be adapted at each iteration, against the $3n$ coefficients for MOD. Moreover, simulations show that in order to obtain similar performances it is necessary to choose $n > r$, like in the third experiment shown above where $n = 4$ and $r = 1$. In addition, the learning algorithm (11) is much simpler than the MOD-based one. In order to perform a fair complexity comparison, the number of operations (multiplications, divisions and non-linear functions) involved in the learning equations are shown in Table 1. The WARP appears to be easier to implement.

5. Application to independent component analysis—the M-WARP algorithm

Since the pioneering work of Jutten and Herault (1988, 1991), the problem of separating out linear instantaneously mixed statistically independent source signals by the Independent Component Analysis (ICA) technique has been widely investigated both in the neural networks community (Amari, Chen & Cichocki, 1997; Bell & Sejnowski, 1996; Cichocki & Moszczyński, 1992; Cichocki, Unbehauen & Rummert, 1994; Girolami & Fyfe, 1997; Hyvärinen & Oja, 1998; Taleb & Jutten, 1997; Xu et al., 1997; Xu et al., 1998a; Xu et al., 1998b; Yang & Amari, 1997), and in the signal processing community (Cardoso & Comon, 1996; Cardoso & Laheld, 1996; Comon, 1994; Delfosse & Loubaton, 1995; Dinç & Bar-Ness, 1992; Moreau & Macchi, 1996) and several algorithms and methods have been proposed by many authors. An excellent recent review, covering several aspects of the ICA theory and practice, is given by Chicocki, Karhunen, Kasprzak and Vigario (1999), Giannakopoulos, Karhunen and Oja (2000), Haykin (1996), Karhunen, Hyvärinen, Vigário, Hurri and Oja (1997), Lee (1998) and Liu (1996). Particularly, the Information-Theoretic approach by Bell and Sejnowski (1996) with the Natural Gradient modification developed by Amari (see Yang & Amari, 1997 and references therein) has attracted much interest in the neural network community. However, both analytical studies and computer experiments have shown that this algorithm cannot be ensured to work with all kind of source signals, that means it is effective depending on the source probability distribution. This behavior may be explained by recognizing that Bell–Sejnowski's method relies on the use of some non-linear functions whose optimal shapes are the cumulative distribution functions (cdfs) of the sources (Bell & Sejnowski, 1996; Yang & Amari, 1997), thus fixed non-linear functions like simple sigmoids would not be the best ones. On the other hand, in a blind problem the cdfs of the sources are unknown, thus the problem cannot be directly solved.

Since 1996, some researchers in the field started claiming the use of flexible (adaptive) non-linear functions could help solve the problem. To this aim, Pearlmutter and Parra (1996) proposed the use of linear combinations of parametric basis functions; Baram and Roth (1994), Roth and Baram (1996) and Taleb and Jutten (1997) employed a MLP in order to adaptively estimate the 'score function' or the pdf of the sources, respectively. Gustaffson (1998) proposed the use of an adjustable linear combination of quasi-Dirac's delta-functions, while Xu et al. (1997, 1998a,b) presented a 'mixture of density' based approximation technique. Other helpful observations about the usefulness of the mentioned approach are given by Amari et al. (1997), Obradovic and Deco (1997) and Xu et al. (1997). These flexible functions may be 'learnt' so that they approximate somehow the required statistical functions helping the separation algorithm to perform better. Particularly they overcome the problem of performing ICA of both sub-Gaussian and super-Gaussian sources without explicitly estimating their kurtosis.

The aim of this section is to extend the WARP algorithm to a multiple version (M-WARP) and apply it to source separation problems. Thus we first present the new learning equations, then we briefly recall the complete MOD algorithm used in Xu et al. (1997, 1998a) and finally illustrate and compare through computer simulations the performances of the two algorithms.

5.1. The independent component analysis technique

For separating out m linearly mixed independent sources, we use a neural network with m inputs and m outputs, whose linear part is described by the relationship:

$$\mathbf{x}(t) = \mathbf{W}(t)\mathbf{z}(t), \tag{21}$$

where $\mathbf{z}$ is the network input vector, $\mathbf{x} = [x_1 x_2 \cdots x_m]^{\mathrm{T}}$ denotes the network output vector, and $\mathbf{W}$ is the weight-matrix at time t. (Each row-vector $\mathbf{w}_j$ represents the weight vector of corresponding jth neuron.) The mixing model is

$$\mathbf{z}(t) = \mathbf{M}\mathbf{s}(t), \tag{22}$$

where $\mathbf{M}$ is a constant real-valued full-rank $m \times m$ mixing matrix and $\mathbf{s}$ the column vector containing the m source signals to be separated, i.e. we only deal with the instantaneous linear mixture problem. The only hypotheses made on

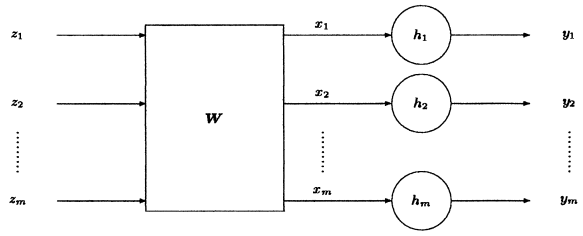

Fig. 8. Neural network used for achieving separation.

the source signals are as follows:

1. Each s_j is an independent identically distributed stationary random process.
2. The signals s_j are statistically independent at any time, i.e. their joint pdf $p_{s_1,s_2,\ldots,s_m}(s_1, s_2, \ldots, s_m)$ factorizes into the product of their symmetric marginal pdfs $p_{s_j}(s_j)$.
3. At most, one among the s_j may follow a Gaussian distribution.

The basic principle that the ICA technique is based on, is that after application of the mixing model (22), the resulting processes z_j are no longer statistically independent; thus, in order to achieve separation, that means finding a weight matrix $\mathbf{W}$ such that $\mathbf{WM}$ equals the identity matrix (except for arbitrary permutation and scaling (Comon, 1994)), the weight matrix may be learnt so that the network's outputs (21) become as independent as possible, i.e. they satisfy the already mentioned complete factorization principle, which for the network's outputs rewrites:

$$p_x(\mathbf{x}) = p_{x_1}(x_1)p_{x_2}(x_2)\cdots p_{x_m}(x_m). \tag{23}$$

A way to achieve separation is thus to define a measure of the mismatching between the two sides of the above equation, and an algorithm to iteratively find the weight matrix that minimizes this measure.

Several possible criteria have been reported in the scientific literature concerning ICA during the last years. Among others, a very interesting and fruitful approach is the one that relies on the Kullback–Leibler divergence among two distributions. Using our notation, the divergence is defined as:

$$\mathscr{D}(p_x(x)\|\prod_j p_{s_j}(x_j)) \stackrel{\text{def}}{=} \int_{\mathbb{R}^m} \log \frac{p_x(\mathbf{x})}{p_{s_1}(x_1)p_{s_2}(x_2)\cdots p_{s_m}(x_m)} p_x(x)\mathrm{d}x, \tag{24}$$

where we made use of the equality $\prod_j p_{x_j}(x_j) = \prod_j p_{s_j}(x_j)$, which must hold whenever condition (23) holds true.

It is important to recall that in a blind context, the marginal pdfs $p_{s_j}(\cdot)$ cannot be exactly known, thus they need to be (iteratively) approximated by means of some time-varying (parametric) functions, more formally:

$$p_{s_j}(x_j) \sim \psi_j(x_j) = \psi_j(x_j; \mathbf{a}_j), \tag{25}$$

where each $\mathbf{a}_j$ represents a vector of suitable size containing the adjustable parameters for the jth approximating

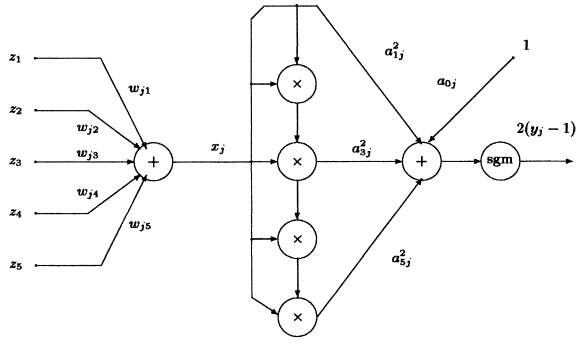

Fig. 9. An example of complete pseudo-polynomial adaptive activation function neuron used for performing the ICA of the mixed source signals (shown for $m = 5$, $r = 2$).

function. If we denote with $\mathbf{A}$ the matrix whose rows are the aforementioned vectors $\mathbf{a}_j$, then we may define a suitable approximation of the criterion (24) as:

$$\bar{\mathscr{D}}(\mathbf{W}, \mathbf{A}) \stackrel{def}{=} \int_{\mathbb{R}^m} \log \frac{p_{\mathbf{z}}(\mathbf{z})/|\det(\mathbf{W})|}{\psi_1(x_1; \mathbf{a}_1)\psi_2(x_2; \mathbf{a}_2)\cdots\psi_m(x_m; \mathbf{a}_m)} p_{\mathbf{z}}(\mathbf{z})\, d\mathbf{z},$$

where each x_j should be thought of as $x_j = \mathbf{w}_j\mathbf{z}$. With some algebra we obtain:

$$\bar{\mathscr{D}}(\mathbf{W}, \mathbf{A}) = -H_z - \frac{1}{|\det(\mathbf{W})|} - \sum_{j=1}^{m} E_z[\log \psi_j(x_j; \mathbf{a}_j)], \tag{26}$$

where $H_{\mathbf{z}}$ denotes the Shannon differential entropy of the multivariate random process $\mathbf{z}$, which does not depend on matrices $\mathbf{W}$ and $\mathbf{A}$. The non-linear functions ψ_j are provided by making the neural layer non-linear, that is, by supposing each neuron be endowed with a non-linear adjustable activation function h_j. This way the whole network is described by:

$$\mathbf{y} = \mathbf{h}(\mathbf{Wz}) = [h_1(x_1)h_2(x_2)\ldots h_m(x_m)]^{\mathrm{T}}, \tag{27}$$

and its structure is depicted in Fig. 8. The mentioned non-linear functions relate by:

$$\psi_j(x_j; \mathbf{a}_j) = h_j'(x_j; \mathbf{a}_j) = \frac{\mathrm{d}h_j(x_j; \mathbf{a}_j)}{\mathrm{d}x_j}, \tag{28}$$

thus any h_j should approximate the cdf of the jth source signal.

To iteratively adjust the weight matrices in order to minimize the cost function $\bar{\mathscr{D}}$, we use the following learning rules:

$$\Delta\mathbf{W} = -\eta_W \frac{\partial\bar{\mathscr{D}}}{\partial\mathbf{W}}(\mathbf{W}^{\mathrm{T}}\mathbf{W}), \tag{29}$$

$$\Delta\mathbf{A} = -\frac{\partial\bar{D}}{\partial\mathbf{A}}\mathbf{H}_{\mathbf{A}}; \tag{30}$$

Eq. (29) represents a natural-gradient descent flow (Xu et al., 1998b); the constant η_W represents the learning step size for $\mathbf{W}$, while the matrix $\mathbf{H}_{\mathbf{A}}$ contains a learning step size for each column of $\mathbf{A}$.

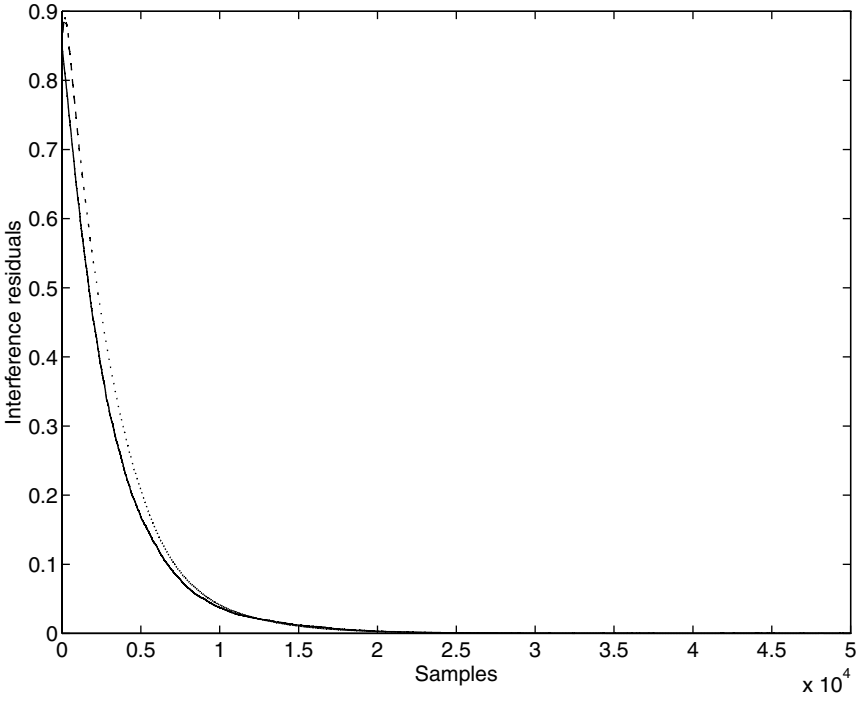

Fig. 10. Two noises. Solid = MOD, Dotdashed = M-WARP.

If we define, for easier notation, the new functions:

$$g_j(x_j;\mathbf{a}_j) \stackrel{def}{=} \frac{\psi_j'(x_j;\mathbf{a}_j)}{\psi_j(x_j;\mathbf{a}_j)} = \frac{h_j''(x_j;\mathbf{a}_j)}{h_j'(x_j;\mathbf{a}_j)},$$

and replace in the above formulas the mean values by their instantaneous estimates (i.e. by dropping down expectation operator), we obtain the following learning rules:

$$\Delta\mathbf{W} \propto [\mathbf{I}_m + \mathbf{g}(\mathbf{x})\mathbf{x}^{\mathrm{T}}]\mathbf{W}, \tag{31}$$

$$\Delta\mathbf{a}_j \propto \frac{1}{\psi_j(x_j)} \frac{\partial\psi_j(x_j)}{\partial\mathbf{a}_j}. \tag{32}$$

Consider a network formed by interconnecting a series of complete adaptive activation function neurons like the one depicted in Fig. 9. The linear part is used for separating out the source signals, while the non-linear (activation) part is used for approximating the source cdf. In the figure, $x(t)$ represents the neuron's net input and $y(t)$ its output, while the $a_k^2(t)$ are the coefficients, at time t; each block marked with ' × ' evaluates instantaneously the product of its inputs, while blocks marked with ' + ' perform summations; sgm(·) is a generic sigmoidal function, and a continuous line marked with a variable or a constant denotes a coefficient link.

It is worth noticing that the criterion (26) may be derived in different ways and is closely related to Maximum Entropy (ME) and Minimal Mutual Information (MMI) criteria (see, for instance, Nadal, Brunel & Parga, 1998; Yang & Amari, 1997). Moreover, in the single-neuron case ($m = 1$ and we may assume $\mathbf{W}$ = scalar = 1), the criterion (26) reduces to criterion (6).

5.2. Multiple WARP (M-WARP) algorithm

In order to approximate the cdfs of the source signals, we use for each neuron the adaptive activation function (3), where now $\mathrm{sgm}(u) = \mathrm{erf}(u)$, that is:

$$y_j = h_j(x_j) \stackrel{\mathrm{def}}{=} \tfrac{1}{2} + \tfrac{1}{2}\mathrm{erf}[q_j(x_j)], \tag{33}$$

where the index j denotes the jth neuron, erf(·) denotes the mathematical 'error function' defined as:

$$\mathrm{erf}(u) \stackrel{\mathrm{def}}{=} \frac{2}{\sqrt{\pi}} \int_0^u \mathrm{e}^{-\xi^2}\,\mathrm{d}\xi,$$

while the polynomials for each neuron again have the form:

$$q_j(x_j) = a_{0j} + \sum_{i=0}^{r_j} a_{2i+1j}^2 x_j^{2i+1}, \qquad j = 1,\ldots,m, \tag{34}$$

where $2r_j + 1$ are the orders of the polynomials; here we chose the 'erf' function because it simplifies the learning equations.

By defining $\mu_{2i+1j}(x_j) \stackrel{\mathrm{def}}{=} 2(2i+1)a_{2i+1j}x_j^{2i}$, with $i =$

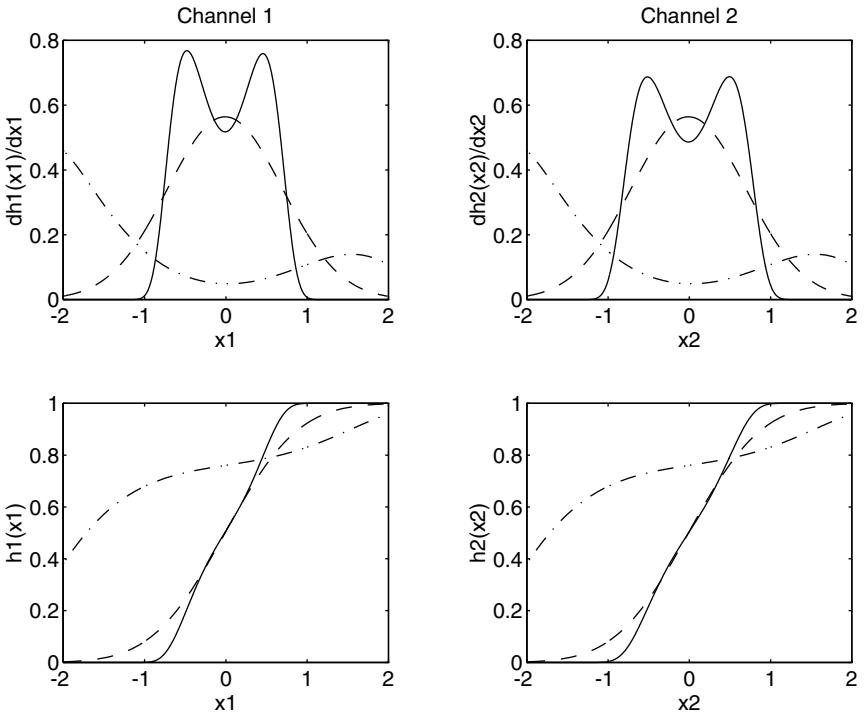

Fig. 11. Two noises, M-WARP. Dotdashed = At the beginning, Solid = After 50,000 steps, Dashed = No adaptation.

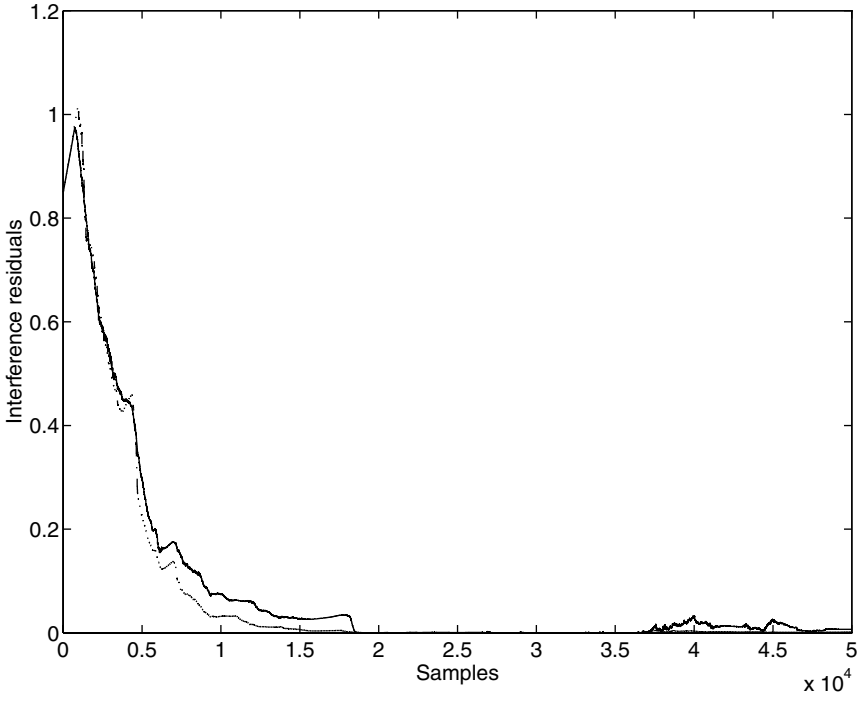

Fig. 12. Two voices. Solid = MOD, Dot-dashed = M-WARP.

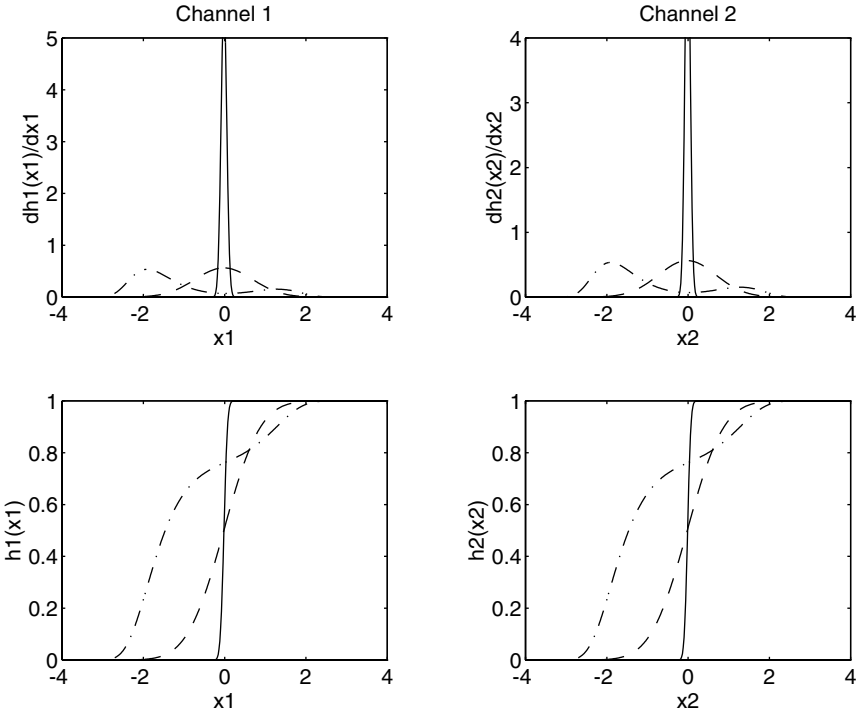

Fig. 13. Two voices, M-WARP. Dot-dashed = At the beginning, Solid = After 50,000 steps, Dashed = No adaptation.

$0, 1, \ldots, r_j$, the M-WARP learning rule can be written as:

$$\begin{cases} \Delta a_{0j} = -2\eta_0 q_j(x_j), \\ \Delta a_{2i+1j} = \eta_{2i+1}\left[\dfrac{1}{q_j'(x_j)} - \dfrac{2}{2i+1} q_j(x_j) x_j\right]\mu_{2i+1j}(x_j), \\ i = 1, \ldots, r_j,\ j = 1, \ldots, m. \end{cases} \tag{35}$$

It could be noted that these equations represent the multiple version of rules (11), but the new rules are simpler because they directly employ the polynomials $q_j(\cdot)$ instead of their warped versions y_j as in (11), that is, the 'erf' sigmoidal non-linearity does not need to be evaluated at all. Also, we need to compute the non-linear functions $g_j(\cdot)$ that are required for adapting the weight matrix $\mathbf{W}$ by Eq. (31). In our case, they take on the expression:

$$g_j(x_j) = \frac{1}{\psi_j}\frac{\mathrm{d}\psi_j}{\mathrm{d}x_j} = -2q_j(x_j)q_x'(x_j) + \frac{q_j''(x_j)}{q_j'(x_j)}, \tag{36}$$

where $q_j''(x_j) = \sum_{i=1}^{r_j} 2i(2i+1)a_{2i+1j}^2 x_j^{2i-1}$.

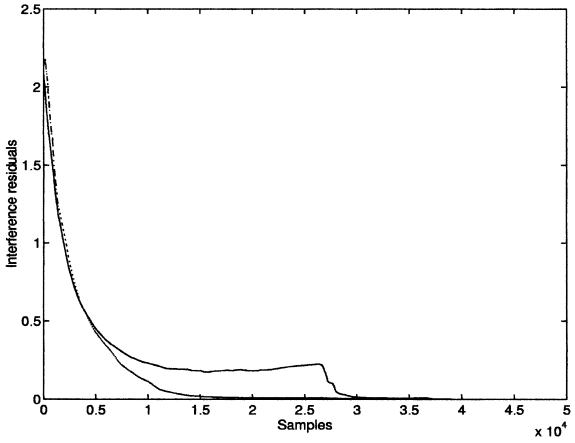

Fig. 14. Sinusoid, voice, noise. Solid = MOD, Dot-dashed = M-WARP.

5.3. MOD algorithm

The complete MOD algorithm by Xu et al. (1997, 1998a,b), arising from the Bayesian–Kullback Ying–Yiang learning theory (Xu et al., 1998b), is used here for comparison to M-WARP. The learning equations were described by Xu et al. (1997, 1998a,b) and are not reported here for the sake of brevity. Note that they require selecting the expansion order n_j for each neuron, and a set of learning stepsizes $\{k_\varrho, k_c, k_b\}$ for the adaptive coefficients, as described in Section 3.

5.4. Computer simulation results on ICA

In order to make a fair comparison between the two algorithms, we chose the learning rates of the adapting rules so that both algorithms show approximately the same performances, like convergence speed and steady-state precision. Moreover, we repeated the experiments proposed by Xu et al. (1997, 1998a,b).

The learning stepsize η_W in Eq. (31) is kept constant in all experiments, and is equal to 0.0001. As a performance measure, we used the residual interference defined as the

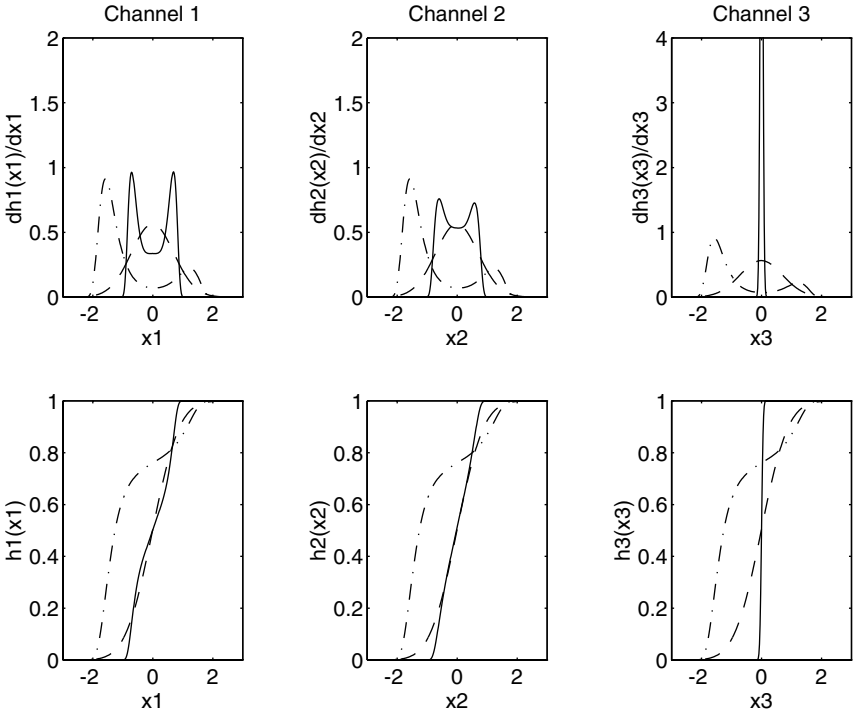

Fig. 15. Sinusoid, voice, noise, M-WARP. Dotdashed = At the beginning, Solid = After 50,000 steps, Dashed = No adaptation.

sum of the $m^2 - m$ smallest squared entries of **WM** as in Hyvärinen and Oja (1998).

In *Experiment 1*, we used two independent identically distributed source sequences (uniformly distributed within $[-1, +1]$). The initial conditions relative to the MOD algorithm are the same used in Xu et al. (1998b). Particularly, for each channel, we have $n_j = 7$. For running the M-WARP algorithm, we assumed $r_j = 1, a_{0j}^{\text{init}} = 1/2, a_{1j}^{\text{init}} = 1/3, a_{3j}^{\text{init}} = 1/4$, $\eta_0 = \eta_{2i+1} = 0.001$ for any i, for any channel. Fig. 10 shows the interference residuals averaged on 10 trials of M-WARP and MOD algorithms, while Fig. 11 shows the functions h_j and h_j' at the beginning of learning, after 50,000 steps and, for comparison like in Xu et al. (1998b), their shape with no parameter adaptation for M-WARP.

In *Experiment 2*, we considered again two source sequences, a female's and a male's speech signal, both sampled at 8 kHz. The test conditions are similar to those of the Experiment 1, except for the learning rates that are now set to $\eta_0 = \eta_{2i+1} = 0.0005$. The pictures displayed in Figs. 12 and 13 have the same meaning as in the preceding Experiment.

In *Experiment 3*, the blind separation algorithms have been tested with three source signals: a pure sinusoid, a uniformly distributed noise and a speech signal. The parameters for MOD algorithm have been chosen as in Xu et al. (1997); also, for each channel we used $n_j = 5$. For M-WARP, we used $r_j = 2$, $a_{0j}^{\text{init}} = 0.5$, $a_{1j}^{\text{init}} = 0.4$, $a_{3j}^{\text{init}} = 0.3$, $a_{5j}^{\text{init}} = 0.2$, and $\eta_0 = \eta_{2i+1} = 0.001$. The results are shown in Figs. 14 and 15 for $j = 1, 2, 3$.

Simulation results of Experiments 1 and 2 show that similar performances may be obtained but with a noticeable difference between the numbers of parameters to be adapted. In fact, they are 46 for the MOD algorithm and 10 for the M-WARP. Furthermore, the computational complexity of the two algorithms is rather different, because any 1000 iterations of M-WARP cost about 4.45 s against 17.96 s of MOD (MATLAB code on a Pentium II machine with 233 MHz processor). Moreover, the second experiment shows that the M-WARP algorithm seems to be more stable after convergence than the MOD. In Experiment 3, the number of required parameters is 54 against the 21 parameters of the M-WARP, while any 1000 iterations cost 6.97 s against 21.53 s of MOD.[1]

Other computer experiments have been performed in order to assess the separation capability of the proposed neural algorithm in the presence of more than three source signals, as well as some intrinsic properties of the learning rules, like for instance, the self-whitening ability. The main conclusions were that

1. the proposed algorithm works well in presence of mixed sub-Gaussian and super-Gaussian sources;
2. as the number of sources grows, the performances becomes more sensitive to the degree of the polynomials used;
3. as the number of parameters to be learnt grows, it is easier for the algorithm to be trapped into local minima,

[1] These numerical results about complexity have been presented for the first time in a preliminary report (Fiori, 1999).

which can be problematic for gradient-based optimization techniques, thus the performances may degrade.

6. Conclusion and further work

The aim of this paper was to present a novel Blind Source Separation technique based on the experience of our research group in the polynomial adaptive activation function neural networks. As claimed in the earlier papers (Piazza et al., 1992; Piazza, Uncini & Zenobi, 1993; Vecci et al., 1997), the flexible activation functions approach allows us to obtain reduced-complexity neural structures, which exhibit the same performance of closely related approaches (Xu et al., 1997, 1998a,b), as illustrated by computer simulation results performed both on synthetic and real-world data. Other simulation results, concerning neural hashing and dehashing (Alon & Orlitsky, 1996; Majewski et al., 1996) have been reported by Fiori et al. (1998a).

As concluding remarks, we wish to propose here some directions along which further studies could be directed:

- Even if a degree of $2r + 1 = 5$–7 for the polynomial seems adequate in all our simulations, a theory for choosing the right degree is required; a possible solution could rely on a feature selection theory allowing to adaptively reduce r (and successively grow it in case of operations in non-stationary enviroments) to the minimum value which grants good performances; a brief review of feature selection techniques, and a theory developed by us and successfully applied to a robot guidance problem, may be found in Fiori et al. (2000).
- The obtained results about cdf/pdf approximation seem to be encouraging; a possible improvement could be the reformulation of the problem in terms of spline-LUT (look-up table) based neurons (Vecci et al., 1997).

Acknowledgements

This research was partially supported by Italian MURST. The author wishes to thank Dr P. Baldassarrri who kindly prepared the experimental set-up for Section 5.4 and Prof. P. Burrascano of DIE for proof-reading the early version of the manuscript and for providing useful suggestions that helped making clearer some parts of the paper.

References

Amari, S.-I., Chen, T.-P., & Cichocki, A. (1997). Stability analysis of learning algorithms for blind source separation. *Neural Networks*, *10* (8), 1345–1351.

Alon, N., & Orlitsky, A. (1996). Source transforming and graph entropies. *IEEE Transactions on Information Processing*, *42* (5), 1329–1339.

Baram, Y., & Roth, Z. (1994). *Density shaping by neural networks with application to classification, estimation and forecasting*, Technical Report CIS-94-20, Center for Intelligent Systems, Technion, Israel Institute for Technology, Haifa..

Bell, A. J., & Sejnowski, T. J. (1996). An information maximisation approach to blind separation and blind deconvolution. *Neural Computation*, *7* (6), 1129–1159.

Bellini, S. (1986). Bussgang techniques for blind equalization. *IEEE Global Telecommunication Conference Records*, 1634–1640.

Cichocki, A., & Moszczyński, L. (1992). New learning algorithm for blind separation of sources. *Electronics Letters*, *28* (21), 1986–1987.

Cichocki, A., Unbehauen, R., & Rummert, E. (1994). Robust learning algorithm for blind separation of signals. *Electronics Letters*, *30* (17), 1386–1387.

Chicocki, A., Karhunen, J., Kasprzak, W., & Vigario, R. (1999). Neural networks for blind separation with unknown number of sources. *Neurocomputing*, *24*, 55–93.

Cardoso, J.-F., & Comon, P. (1996). Independent component analysis, a survey of some algebraic methods. *Proceedings of the International Symposium on Circuits and Systems (IEEE-ISCAS)*, *2*, 93–96.

Cardoso, J.-F., & Laheld, B. (1996). Equivarian adaptive source separation. *IEEE Transactions on Signal Processing*, *44* (12), 3017–3030.

Chen, C. T., & Chang, W. D. (1996). feedforward neural network with function shape autotuning. *Neural Networks*, *9* (4), 627–641.

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

Delfosse, N., & Loubaton, P. (1995). Adaptive blind separation of independent sources: a deflation approach. *Signal Processing*, *45*, 59–83.

Dinç, A., & Bar-Ness, Y. (1992). BOOTSTRAP: a fast blind adaptive signal separator. *Proceedings International Conference on Acoustics, Speech and Signal Processing (IEEE-ICASSP)*, *2*, 325–327.

Fiori, S. (1999). Blind source separation by new M-WARP algorithm. *Electronics Letters*, *35* (4), 269–270.

Fiori, S., & Piazza, F. (1998). A study on functional-link neural units with maximum entropy response. *Proceedings of International Conference on Artificial Neural Networks*, *2*, 493–498.

Fiori, S., Bucciarelli, P., & Piazza, F. (1998a). Blind signal flatting using warping neural modules. *Proceedings of the International Joint Conference on Neural Networks (IEEE-IJCNN)*, *3*, 2312–2317.

Fiori, S., Uncini, A., & Piazza, F. (1998b). Gradient-based blind deconvolution with flexible approximated Bayesian estimator. *Proceedings of the International Joint Conference on Neural Networks (IEEE-IJCNN)*, *2*, 854–858.

Fiori, S., Faustini, A., & Burrascano, P. (2000). *Non-uniform image sampling for robot motion control by the GFS neural algorithm*, Proceedings of International Joint Conference on Neural Networks, July 10–16, 1999, Washington, DC, in press.

Giannakopoulos, X., Karhunen, J., & Oja, E. (2000). An experimental comparison of neural algorithms for independent component analysis and blind separation. *International Journal of Neural Systems*, (in press).

Girolami, M., & Fyfe, C. (1997). Kurtosis extrema and identification of independent components: a neural network approach. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, *4*, 3329–3333.

Gustaffson, M. (1998). Gaussian mixture and kernel based approach to blind source separation using neural networks. *Proceedings of the International Conference on Artificial Neural Networks*, *2*, 869–874.

Haykin, S. (1996). Neural networks expand SP's horizons. *IEEE Signal Processing Magazine*, 24–49.

Hyvärinen, A., & Oja, E. (1998). Independent component analysis by general non-linear Hebbian-like rules. *Signal Processing*, *64* (3), 301–313.

Karhunen, J., Hyvärinen, A., Vigário, R., Hurri, J., & Oja, E. (1997). Applications of neural blind separation to signal and image processing. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (IEEE-ICASSP)*, *1*, 131–134.

Jutten, C., & Herault, J. (1988). Independent component analysis versus

principal component analysis. *Proceedings of the EUSIPCO, 2*, 643–646.

Jutten, C., & Herault, J. (1991). Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture. *Signal Processing, 24*, 1–10.

Lee, T.-W. (1998). *Independent component analysis—theory and practice*, Dordrecht: Kluwer Academic.

Linsker, R. (1989). *An application of the principle of maximum information preservation to linear systems, Advances in neural information processing system (NIPS*88)*. Los Altos, CA: Morgan-Kaufmann (pp. 186–194).

Linsker, R. (1992). Local synaptic rules suffice to maximize mutual information in a linear network. *Neural Computation, 4*, 691–702.

Liu, R.-W. (1996). Blind signal processing: an introduction. *Proceedings of the International Symposium on Circuits and Systems (IEEE-ISCAS), 2*, 81–84.

Majewski, B. S., Wormald, N. C., Havas, G., & Czech, Z. J. (1996). A family of perfect hashing methods. *Computer Journal, 39*, 547–554.

Mel, B. W. (1994). Information processing in dendritic tree. *Neural Computation, 6*, 1031–1085.

Miller, G., & Horn, D. (1998). Probability density estimation using entropy maximization. *Neural Computation, 10*, 1925–1938.

Moreau, E., & Macchi, O. (1996). High-order contrasts for self-adaptive source separation. *International Journal of Adaptive Control and Signal Processing, 10*, 19–46.

Nadal, J. P., Brunel, N., & Parga, N. (1998). Nonlinear feedforward networks with stochastic inputs: infomax implies redundancy reduction. *Network: Computation in Neural Systems, 9* (2), 207–217.

Obradovic, D., & Deco, G. (1997). Unsupervised learning for blind source separation: an information-theoretic approach. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (IEEE-ICASSP)*, 127–130.

Pao, Y.-H. (1989). *Adaptive pattern recognition and neural networks*, Reading MA: Addison-Wesley (chap. 8).

Pearlmutter, B. A., & Parra, L. C. (1996). Maximum likelihood blind source separation: a context-sensitive generalization of ICA. In M. M. Mozer, M. I. Jordan & T. Petsche, *Proceedings of the Neural Information Processing System (NIPS*9)* (pp. 613–619). .

Piazza, F., Uncini, A., & Zenobi, M. (1992). Artificial neural networks with adaptive polynomial activation function. *Proceedings of the International Joint Conference on Neural Networks, 2*, 343–349.

Piazza, F., Uncini, A., & Zenobi, M. (1993). Neural networks with digital LUT activation function. *Proceedings of the International Joint Conference on Neural Networks, 2*, 1401–1404.

Plumbley, M. D. (1993). Efficient information transfer and anti-Hebbian neural networks. *Neural Networks, 6*, 823–833.

Rumelhart, D. E., & McClellend, J. L. (1986). In D. E. Rumelhart & J. L. McClellend, *Parallel distributed processing* (Vol. 1). Cambridge, MA: MIT Press.

Roth, Z., & Baram, Y. (1996). Multidimensional density shaping by sigmoids. *IEEE Transactions on Neural Networks, 7* (5), 1291–1298.

Sudjianto, A., & Hassoun, M. H. (1994). Nonlinear Hebbian rule: a statistical interpretation. *Proceedings of the International Conference on Neural Networks, (IEEE-ICNN), 2*, 1247–1252.

Taleb, A., & Jutten, C. (1997). *Entropy optimization—application to source separation, Proceedings of the International Conference on Artificial Neural Networks*. Berlin: Springer (pp. 529–534).

Vapnik, V. (1997). *The support vector method, Proceedings of the International Conference on Artificial Neural Networks*. Berlin: Springer (pp. 263–271).

Vecci, L., Piazza, F., & Uncini, A. (1997). Learning and approximation capabilities of adaptive spline activation function neural networks. *Neural Networks, 11* (2), 259–270.

Xu, L., Cheung, C. C., Ruan, J., & Amari, S.-I. (1997). Nonlinearity and separation capability: further justifications for the ICA algorithm with a learned mixture of parametric densities. *Proceedings of the European Symposium on Artificial Neural Networks*, (ESANN'97), 291–296.

Xu, L., Cheung, C. C., & Amari, S.-I. (1998a). Learned parametric mixture based ICA algorithm. *Neurocomputing (Special issue on Independence and Artificial Neural Networks), 22* (1–3), 69–80.

Xu, L., Cheung, C. C., Yang, H. H., & Amari, S.-I. (1998b). Independent component analysis by the information-theoretic approach with mixture of densities. *Proceedings of the International Joint Conference on Neural Networks (IEEE-IJCNN)*, 1821–1826.

Yang, H. H., & Amari, S.-I. (1997). Adaptive online learning algorithms for blind separation: maximum entropy and minimum mutual information. *Neural Computation, 9*, 1457–1482.

Zurada, S. M. (1992). *Introduction to neural artificial systems*, West Publishing Company.