PERGAMON

Neural Networks 15 (2002) 85–94

Neural Networks

www.elsevier.com/locate/neunet

Contributed article

Hybrid independent component analysis by adaptive LUT activation function neurons

Simone Fiori

Neural Networks and Adaptive Systems Research Group, Department of Industrial Engineering—University of Perugia, I-60026, Numana, An, Italy

Received 2 October 2000; accepted 9 May 2001

Abstract

The aim of this paper is to present an efficient implementation of unsupervised adaptive-activation function neurons dedicated to one-dimensional probability density estimation, with application to independent component analysis. The proposed implementation is a computationally light improvement to adaptive pseudo-polynomial neurons, recently presented in Fiori, S. (2000a). Blind signal processing by the adaptive activation function neurons. *Neural Networks*, *13*(6), 597–611, and is based upon the concept of 'look-up table' (LUT) neurons.

Keywords: Adaptive activation function neurons; Look-up table neurons; Independent component analysis; Minimal mutual information principle; Kullback–Leibler divergence; Natural gradient

1. Introduction

The aim of independent component analysis (ICA) technique is to recover a number of statistically independent signals from their unknown linear mixtures, under simple consistency conditions. Namely, a mixture of independent source signals is supposed to be observed:

$$\mathbf{u}(t) = \mathbf{M}\mathbf{s}(t), \tag{1}$$

where $\mathbf{M}$ is an unknown constant real-valued full-rank $n \times n$ mixing matrix and $\mathbf{s}(t)$ is the vector-stream containing the n source signals to be separated. The only hypotheses made on the unknown sources are: (1) each $s_i(t)$ is an independent identically distributed (IID) stationary random process; (2) the $s_i(t)$ are statistically independent at any time; and (3) at most one among the source signals has Gaussian distribution.

For separating out the independent sources from the mixture, a neural network with n inputs and n outputs can be used; it is described by the relationship:

$$\mathbf{x}(t) = \mathbf{W}(t)\mathbf{u}(t), \tag{2}$$

where $\mathbf{u}$ is the network input vector and $\mathbf{W}$ denotes the connection-matrix. As the mixing model is linear, a linear separating structure is effective, and the network's output $\mathbf{x}(t)$ in Eq. (2) is taken as an estimate of the true source stream $\mathbf{s}(t)$. Under the above conditions, the sources may be recovered up to arbitrary scaling and permutation (Comon, 1994).

Over recent years, several possible solutions to ICA estimation problems have been proposed, which originated from very different perspectives. It is worth citing the method based on maximization of negentropy (Girolami & Fyfe, 1997), the use of high-order cumulants (Comon, 1994), the techniques based on non-linear principal component analysis (Karhunen, Oja, Wang, Vigário & Joutsensalo, 1997; Oja, 1997; Fiori, 2000b), the ones based on maximum likelihood estimation (Belouchrani & Cardoso, 1995; Pearlmutter & Parra, 1997; Pham, Garrat & Jutten, 1992), the recent proposal of Bayesian ICA (Knuth, 1998; Roberts, 1998) and maximum-a-posteriori (MAP) parameter estimation for ICA (Parra, Mueller, Spence, Ziehe & Sajda, 2000), and an algorithm based on a purely geometrical method (Puntonet & Prieto, 1998). It is worth noting that some existing ICA techniques benefited from the closely-related topic of blind deconvolution, long investigated in the context of adaptive filtering (Fiori, 2002; Fiori & Maiolini, 2000). Also, several researchers have focused their interest on a particularly fruitful research stream related to minimal mutual information (MMI) (Bell & Sejnowski, 1996; Yang & Amari, 1997).

The basic principle that the MMI-ICA technique is based on is that after application of the mixing model (1), the observed signals $u_i(t)$ are no longer statistically independent.

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

0893-6080/02/$ - see front matter
PII: S0893-6080(01)00105-8

Thus, in order to achieve separation, the network connection-matrix may be learned so that the network's outputs (2) become as independent as possible, that is, they satisfy the complete factorization principle:

$$p_{\mathbf{x}}(\mathbf{x}) = p_1(x_1)p_2(x_2)\cdots p_n(x_n), \tag{3}$$

where $p_{\mathbf{x}}(\mathbf{x})$ denotes the joint probability density function of the outputs, and the functions:

$$p_i(x_i) \stackrel{\text{def}}{=} \int_{\mathbb{R}^{n-1}} p_{\mathbf{x}}(\mathbf{x})\mathrm{d}x_1\cdots\mathrm{d}x_{i-1}\mathrm{d}x_{i+1}\mathrm{d}x_n,$$

denote the marginal probability densities of network output signals; by definition each $p_i(x_i)$ is non-negative and integrates to 1. A way to achieve separation is, thus, to define a measure of the disagreement between the two sides of Eq. (3), and a learning algorithm to learn the connection-matrix in order to minimize such disagreement. A useful and widely employed measure is the mutual information between a network's output signals, which coincides to Kullback–Leibler informational divergence between the output signals' joint probability density function and the product of marginal probabilities.

Once an appropriate criterion has been defined as a function of a network's connection-matrix, the learning phase may be formally conceived as an optimization procedure that allows iterative searching for the connection pattern that minimizes or maximizes the criterion. As improvements to original gradient-based learning rules (Bell & Sejnowski, 1996), several new optimization techniques, oriented to efficient extraction of independent components, have been recently developed, such as the relative gradient (Cardoso & Laheld, 1996), the fixed point technique (Hyvärinen & Oja, 1997), the Riemannian gradient on Stiefel manifold and orthogonal group (Fiori, 2001; Nishimori, 1999) and the use of the class of Jacobi algorithms for contrast function optimization (Cardoso, 1999). A widely used technique which conjugates easiness of implementation, fast convergence and the interesting property of equivariancy (i.e. the performances do not depend on mixing matrix and, in particular, on its conditioning), is the natural-gradient one by Amari (Amari, 1998; Yang & Amari, 1997). The natural-gradient version of stochastic MMI learning rule writes (Yang & Amari, 1997):

$$\Delta\mathbf{W} = \eta[\mathbf{I} - \Phi(\mathbf{x})\mathbf{x}^T]\mathbf{W}, \tag{4}$$

$$\Phi(\mathbf{x}) \stackrel{\text{def}}{=} \left(-\frac{r_1'(x_1)}{r_1(x_1)}, -\frac{r_2'(x_2)}{r_2(x_2)}, \ldots, -\frac{r_n'(x_n)}{r_n(x_n)}\right)^T, \tag{5}$$

where $r_i(x_i)$ stands, in the optimal case for $p_i(x_i)$, and η is a positive learning step size.

However, in a blind context, the marginal probability density functions of a network's output signals are unknown (being functions of the source signals' probability distributions) and variable (as they depend on a network's connection pattern, which adapts through time). Thus, the non-linear functions $r_i(x_i)$ may only be retained as estimates/approximations of the true marginal densities. Usually, the density functions are divided in two classes: the set of sub-Gaussian ones (i.e. having negative kurtosis); and the set of super-Gaussian ones (having positive kurtosis). When the source signals have only positive or negative kurtoses, the problem may be overcome by choosing appropriate non-linear functions (Cardoso & Laheld, 1996; Comon & Moreau, 1997; Girolami & Fyfe, 1997; Karhunen et al., 1997). The problem becomes harder to solve when the sources are mixed sub- and super-Gaussian, which is referred to as hetero-kurtotic or hybrid ICA problem.

To tackle this problem, Bell and Sejnowski (1996) proposed the use of a family of flexible (non-adaptive) non-linear functions, while Thawonmas, Cichocki and Amari (1998) employ a self-switching (deflationary) algorithm to automatically change the shape of the non-linearity as the sign of the kurtosis of each network's output signal has stabilized. A similar approach was then proposed by Lee, Girolami and Sejnowski (1999) to extend the Bell–Sejnowski algorithm to the separation of sub- and super-Gaussian sources. Both rely on guessing the kurtoses of the sources from a network's output signals.

On the other hand, some researchers claim that the use of adaptive non-linear functions could help: Pearlmutter and Parra (1996) proposed the use of linear combinations of parametric basis functions; Taleb and Jutten (1997) employed a MLP in order to adaptively estimate the 'score function' of the sources; Gustaffson (1998) proposed the use of an adjustable linear combination of quasi-Dirac's-delta-functions; while Xu, Cheung and Amari (1998) used the well-known 'mixture of kernel' approximation technique. These flexible functions may be learned so that they approximate the required statistical functions helping the separation algorithm to manage the hybrid case.

In our recent work (Fiori & Bucciarelli, 2001), we proposed to employ a pseudo-polynomial adaptive activation function neuron in order to iteratively estimate the probability density functions $p_i(x_i)$ from observations of random-streams x_i; we extended the concept of adaptive (spline) activation function neurons (Catmull & Rom, 1974), extensively studied under different frameworks by Chen and Chang (1996); Chen and Manry (1993); Hu and Shao (1992); Yamada and Yabuta (1992) and recently by Trentin (2001) and Vecci, Piazza and Uncini (1998); see also Mathews (1991) for the relationship with adaptive Volterra filtering. We also formulated the adaptation equations for the unsupervised case. Our observation was that the cumulative distribution function $P_i(x_i) \stackrel{\text{def}}{=} \int_{-\infty}^{x_i} p_i(\xi_i)\mathrm{d}\xi_i$ of a signal is a monotonically non-decreasing saturating (sigmoidal) function whose approximating parametric form may be regarded as a learnable non-linear activation function for a neuron in the separating network. As a parametric form, we choose $R_i(x) = \mathrm{sgm}(q_i(x))$, where $\mathrm{sgm}(\cdot) \in [0, 1]$ is whatever sigmoidal function, such as

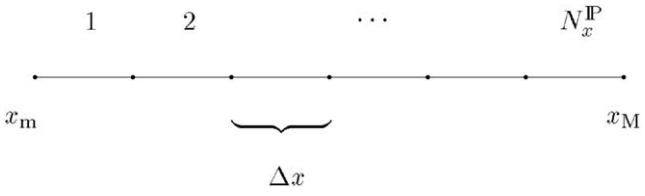

Fig. 1. Enumeration of sub-intervals for input signal quantization.

$1/2 + (1/2)\tanh(x)$ or $(2/\sqrt{\pi}) \int_{-\infty}^{x} \exp(-\xi^2)\mathrm{d}\xi$, and $q_i(x)$ is a polynomial of proper degree being monotonically non-decreasing, e.g. $q_i(x) = \sum_k q_{k,i}^2 x^{2k-1} + q_{0,i}$, which allows representing multi-modal as well as uni-modal distributions; the parameters q_k adapt through an unsupervised learning rule. Such pseudo-polynomial approach was then applied to hybrid independent component analysis (Fiori, 2000a) and has proven to give comparable numerical results with respect to the standard mixture-of-kernel technique, but with a considerable computational saving.

The mentioned contributions have specific merits and drawbacks. It is quite apparent that they provide quite good approximations of the required statistical functions, in an adaptive (time-varying) situation. However, concluding our paper (Fiori, 2000a) we argued that these methods are far from exhibiting minimal computational complexity, which is a desirable property to save computational resources, e.g. in digital implementations for real-time processing on dedicated machines. In fact, pseudo-polynomial functions, constrained to have suitable shapes for probability/cumulative function approximation and the mixture-of-kernel method are very computationally demanding.

These considerations inspired in us the idea of employing the very simplest way to estimate probability density functions. A possible answer to the practical requirement of lessening the computational complexity comes from a typical digital implementation of the MLP networks, where the computation of the activation function of each neuron is performed by the help of a look-up table (LUT), that can be a RAM or ROM memory, which is addressed by the result of the weighted sum on a network's inputs and gives as response the content of the addressed cell. If the LUT is previously filled with samples drawn from a suitable function, e.g. a sigmoid, the given activation function is efficiently implemented, i.e. it works fast as it requires a minimal amount of computation, which might grow when more or less sophisticated interpolation techniques are employed (for a review see, e.g. Piazza, Uncini & Zenobi, 1993, and references therein). In such a way, only fixed activation functions can be implemented.

A useful improvement to this basic implementation is to adapt the values contained in the LUT during network learning: in this way the neurons can have suitable (not known a priori) activation functions efficiently implemented (Karam & Sari, 1990). Particularly, Piazza et al. (1993) treated the LUT's entries as free parameters to adapt in a supervised way through usual back-propagation. As no constraints were enforced on LUT's contents, when tested on standard classification problems such as XOR and 4-bit parity, their learning process resulted in curious (non-monotonic) shapes (recently the power of non-monotonic activation functions has been investigated by Crespi, 1999).

On the basis of these findings, in the present paper we retain the basic structure of LUT neuron and extended its learning to the unsupervised case for density shaping oriented to hybrid independent component analysis. In this case, some constraints should be taken into account, as the non-linear function to be adaptively stored on the LUT must be positive and sum to one. In comparison to the mentioned estimation techniques, the one presented here is far lighter about computational/structural complexity, since ultimately the updating of a LUT for density estimation requires only the value of a counter to be incremented. In the following, details on digital LUT neuron's structure definition and implementation are given, and some problems, such as the computation of the digital derivative of a neuron's response probability density function (required in Eq. (5)), are addressed. Then, numerical simulation results on synthetic and real-world signals are shown to illustrate the effectiveness of the proposed approach to hybrid ICA, and a comparison of the computational complexity of the proposed method for estimating the score function $-p'(x)/p(x)$ with other techniques from the literature completes the discussion.

2. Iterative blind density shaping by the LUT neurons

Let us consider the scalar random variable $x \in \mathbb{R}$, whose statistical distribution is described by the continuous density function $p(x)$. We suppose that $p(x)$ differs significantly from zero only in a finite interval $\mathbb{P} \stackrel{\mathrm{def}}{=} [x_{\mathrm{m}}, x_{\mathrm{M}}]$, outside of which $p(x) \approx 0$. The above interval partitions into $N_x^{\mathbb{P}} \geq 3$ equally-wide sub-intervals of width $\Delta x > 0$, such that:

$$|x_{\mathrm{M}} - x_{\mathrm{m}}| = N_x^{\mathbb{P}} \Delta x; \tag{6}$$

for simplicity, it is convenient to choose $N_x^{\mathbb{P}}$ as an odd integer. The intervals are then enumerated as shown in Fig. 1; as a consequence, each value of x can be associated an integer value $n_x \in \{1, \ldots, N_x^{\mathbb{P}}\}$ corresponding to the interval that it falls inside, by the formula:

$$n_x = \left\lfloor \frac{x - x_{\mathrm{m}}}{\Delta x} \right\rfloor + 1, \tag{7}$$

where symbol $\lfloor \cdot \rfloor$ stands for 'floor' operator.

Given a set of N_x^{ob} observations of random variable x, which distribute on $\mathbb{P}$ according to $p(x)$, the total amount of times n_x^{ob} that a value of x falls inside the n_x-th interval reflects the distribution, namely the quantity:

$$r(n_x) \stackrel{\mathrm{def}}{=} \frac{n_x^{\mathrm{ob}}}{N_x^{\mathrm{ob}}} \tag{8}$$

Initialize x_{m}, x_{M}, $N_x^{\mathbb{P}}$, $\mathbf{r} := \mathbf{0}$.

Compute $\Delta x = \frac{|x_{\mathrm{M}} - x_{\mathrm{m}}|}{N_x^{\mathbb{P}}}$.

Repeat

  Read x

  Compute $n_x = \lfloor \frac{x - x_{\mathrm{m}}}{\Delta x} \rfloor + 1$

  If $(n_x \geq 1)$ AND $(n_x \leq N_x^{\mathbb{P}})$ then

    $r_{n_x} := r_{n_x} + 1$

  EndIf

Until EndOfObservation

Compute $\mathbf{r} := \frac{\mathbf{r}}{\sum_{k=1}^{N_x^{\mathbb{P}}} r_k}$

Fig. 2. Iterative algorithm for the estimation $\mathbf{r}_{n_x}$ of $p(x)$ (symbol ':=' means variable assignment).

may be assumed as an estimate of the true density function. The quality of the estimation depends on the accuracy of the choice of interval $\mathbb{P}$, on the number of partitions $N_x^{\mathbb{P}}$, and on the cardinality of available data N_x^{ob}.

The mentioned estimation theory, known as relative-occurrence-frequency method or histogram method (e.g. Bishop, 1995), may readily be extended to iterative probability density function estimation. To this aim, let the vector $\mathbf{r} \in \mathbb{R}^{N_x^{\mathbb{P}}}$ be defined, and suppose only one sample of x at a time is available to update the estimate of its density function $p(x)$. A simple way to update the estimation $\mathbf{r}$ is to increment the value of accumulator r_{n_x} every time the value of x falls inside the n_x-th interval. The iterative estimation algorithm is reported in Fig. 2. The if–then check is used to

AtStartOnly: Initialize x_{m}, x_{M}, $N_x^{\mathbb{P}}$, $\mathbf{r} := \mathbf{0}$, $\Phi := 0$.

AtStartOnly: Compute $\Delta x = \frac{|x_{\mathrm{M}} - x_{\mathrm{m}}|}{N_x^{\mathbb{P}}}$.

Input x

Compute $n_x = \lfloor \frac{x - x_{\mathrm{m}}}{\Delta x} \rfloor + 1$

If $(n_x \geq 1)$ AND $(n_x \leq N_x^{\mathbb{P}})$ then

  $r_{n_x} := r_{n_x} + 1$

  If $(n_x > 1)$ AND $(n_x < N_x^{\mathbb{P}})$ then

    $r_i^1 := r_i / (\sum_k r_k)$, $i = n_x - 1, n_x, n_x + 1$

    $\Phi := -\frac{r_{n_x+1}^1 - r_{n_x-1}^1}{2(r_{n_x}^1 + \epsilon)\Delta x}$

  EndIf

EndIf

Output Φ

Fig. 3. A LUT unit for the estimation of $\Phi(x)$.

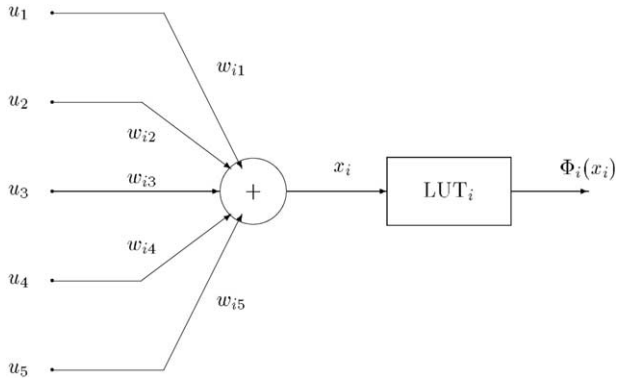

Fig. 4. A complete LUT-neuron (shown for $n = 5$).

prevent the index n_x from exceeding the range $[1, N_x^{\mathbb{P}}]$, which may occur if the interval $\mathbb{P}$ does not suffice to contain the dynamics of x. The normalization at the end of the cycle is also necessary to meet $\sum_k r_k = 1$, the consistency condition.[1] After adaptation, the vector $\mathbf{r}$ constitutes a look-up table (LUT), as the estimated density function $p(x)$ computed on value $\bar{x}$ obtains by computing first the associated interval index $n_{\bar{x}}$ by Eq. (7) and then by looking at the $n_{\bar{x}}$-th element of the table.

In order to apply this simple iterative estimation algorithm to independent component analysis, we need to construct a LUT module which returns the (approximated) mapping $x \rightarrow \Phi(x)$, where $\Phi(\cdot)$ is any score function necessary in Eq. (5). To this aim, we may approximate $p'(x)$ almost everywhere with the symmetric digital derivative formula:

$$p'(x) \approx \frac{1}{2}\left(\frac{r_{n_x} - r_{n_x-1}}{\Delta x} + \frac{r_{n_x+1} - r_{n_x}}{\Delta x} \right), \tag{9}$$

thus, the LUT neuron algorithm reads as in Fig. 3. Note that if $x \notin \mathbb{P}$ or $x \in [x_{\mathrm{m}}, x_{\mathrm{m}} + \Delta x[\cup]x_{\mathrm{M}} - \Delta x, x_{\mathrm{M}}]$, then $\Phi(x)$ is forced to zero. Also, note that for numerical reasons only, the computation of the non-linear learning function Φ still requires the normalization of three entries of the accumulator-vector $\mathbf{r}$, as well as the addition of a small constant ϵ in order to avoid a division by zero.

An exemplary complete LUT neuron is depicted in Fig. 4. The linear part learns to estimate a source signal by means of rule (4), while the LUT part tries to estimate *and track* the correct mapping $\Phi(\cdot)$ as learning proceeds.

It is also interesting to note that the estimation of the

[1] Note that, saying t the current learning step index, for the unnormalized vector $\mathbf{r}$ the approximation $\sum_k r_k \approx t$ holds. We cannot ensure perfect equality because it is possible that infrequently the index n_x falls outside the range $[1, N_x^{\mathbb{P}}]$, which causes the missing of the corresponding input sample, therefore at time t it is possible that the LUT has been updated less than t times. Such occurrence may be avoided by enlarging the interval $\mathbb{P}$ to contain the whole signals' dynamics: in this case we would have $\sum_k r_k = t$.

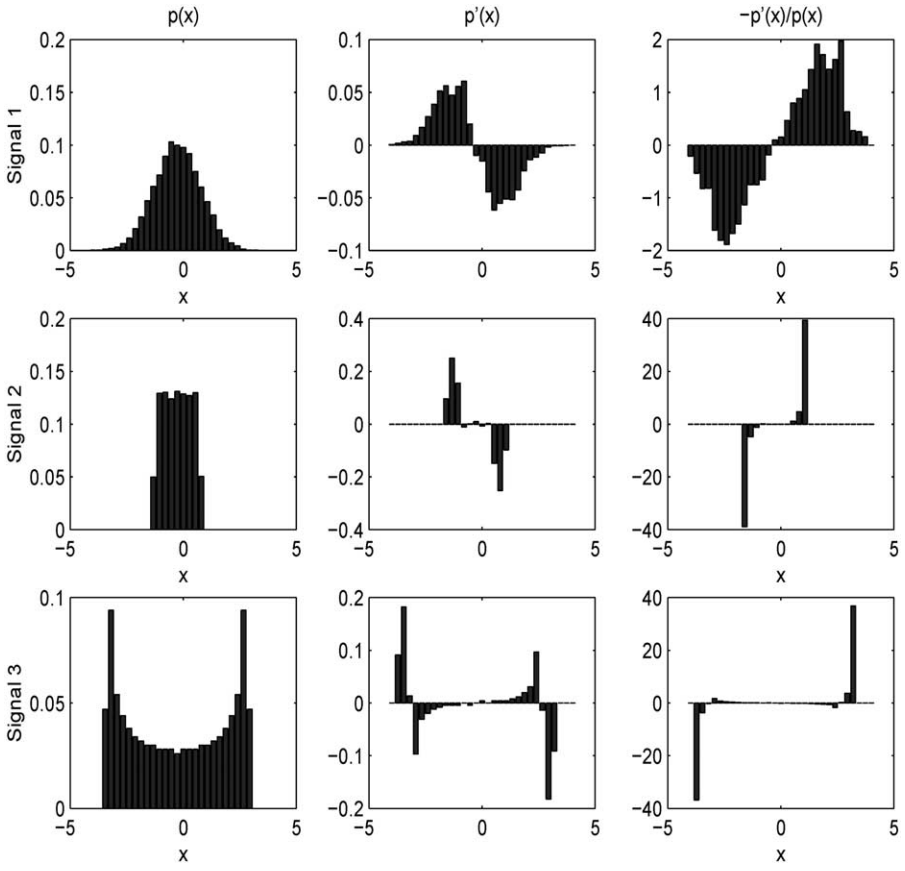

Fig. 5. Example of the estimation of $p(x)$, $p'(x)$ and $\Phi(x)$ for a Gaussian (normal), a uniform and sinusoidal random variable. The histograms reflect the final contents of the LUTs after learning.

required statistical functions is affected by the fact that initially the outputs of the ICA network differ from the true source signals, and may, thus, contain large errors. This phenomenon might be partially mitigated by initializing the LUT content to the samples of some standard sigmoidal function on the basis of prior information on the sources. However, in this paper, we suppose not to have any a priori knowledge about the sources statistics, therefore such possible improvement is not exploited. Another possible improvement would be to 'forget' the very past input samples by reducing the temporal memory of the LUT neuron. This modification, however, implies a different and more costly management of the input signal, and is not investigated here. As will be clear from simulation results, the initial mismatching would only slightly delay network convergence to the correct connection pattern.

3. Experimental results

An illustrative example of static density function estimation is reported in Fig. 5. A normal Gaussian, a random variable uniformly distributed in $[-1, 1]$ and a sinusoidal signal with amplitude 3 were generated; in this example we took $x_{\mathrm{m}} = -4$, $x_{\mathrm{M}} = 4$, $N_x^{\mathbb{P}} = 31$ and $N_x^{\mathrm{ob}} = 20{,}000$. The quality of the approximation looks quite good with respect to the degree of precision controlled by $N_x^{\mathbb{P}}$.

A first result on independent component analysis is reported in Fig. 6, obtained with a complete 3×3 neuro-LUT network with the following data: $x_{\mathrm{m}} = -4$, $x_{\mathrm{M}} = 4$, $N_x^{\mathbb{P}} = 21$, $\epsilon = \exp(-6)$, $\mathbf{W}(0) = 0.1\mathbf{I}$, $\eta = 0.0003$.

The independent signal recovering performance index, termed signal-to-interference ratio (SIR), has been defined according to other authors by taking into account that the matrix product $\mathbf{P} \stackrel{\mathrm{def}}{=} \mathbf{WM}$ should tend to a quasi-diagonal matrix, having only one entry per row different from zero, namely:

$$\mathrm{SIR} \stackrel{\mathrm{def}}{=} \frac{\sum_i \sum_j P_{ij}^2}{\sum_i \max_j \{P_{ij}^2\}} - 1. \tag{10}$$

It is interesting to note that, as also evidenced by this simulation, the initial connection-matrix should have small entries, for instance be of the form $\mathbf{W}(0) = \alpha\mathbf{I}$, with $\alpha \ll 1$. This way the values of $\mathbf{x}$ initially

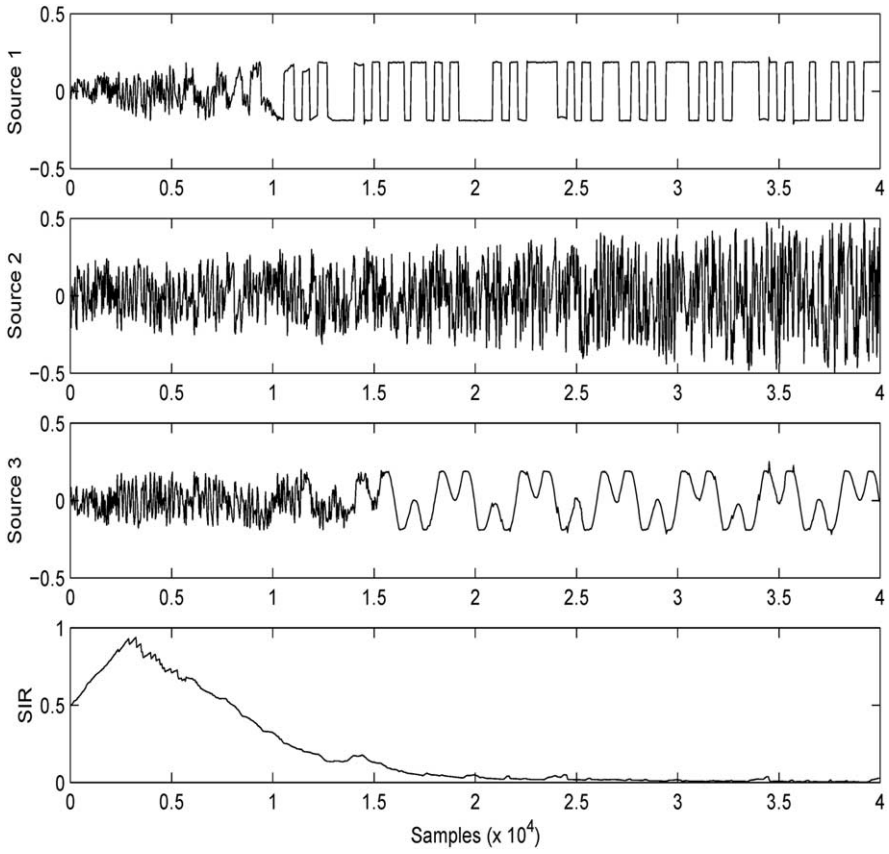

Fig. 6. A result on ICA: the first three graphs represent source signals recovering versus learning progress, while the fourth graph shows the performance index.

concentrate around 0. Thus, $\Phi(\mathbf{x}) \approx 0$ and $\Delta\mathbf{W} \approx \eta\mathbf{W}$, thus for small t it holds $\mathbf{W}(t) \approx (1+\eta)^t \alpha\mathbf{I}$. This way the connection-matrix entries grow and the matrix scales to bring the network's outputs values within the prescribed interval $\mathbb{P}$, that the outputs signals values must be confined within. Also note that at the beginning of learning the SIR value grows a little because of the predicted initial bad approximation of the true sources probability density functions.

A more difficult separation task was carried out with five grey-level natural images, whose statistical distributions are shown in Fig. 7. Their kurtoses are in $\{1.9809, 0.3998, -0.0467, -0.8430, -1.6713\}$. The separation of real-world images is known as a difficult problem because they are never completely independent, thus perfect separation is sometimes not possible with plain ICA technique. Also, the last image (a picture of a polyester slab) has been recently used by Costa and Fiori (2001) and Fiori and Piazza (2000) to compare still image compression techniques based on PCA-type neural networks, and has been found quite hard to process because of its particular spectral composition. The results with a 5×5 neuro-LUT network have been obtained by using parameters values $N_x^{\mathbb{P}} = 31$, $x_M = 2$, $x_m = -2$, $\mathbf{W}(0) = 0.1\mathbf{I}$, $\eta = 0.0002$. The original, mixed and recovered images are shown in Fig. 8, while the SIR versus time is depicted in Fig. 9.

4. Discussion and conclusions

In the tutorial paper (Cardoso, 1998), Cardoso mentions as an open problem the estimation of source signal's probability density functions; the source distributions are looked as 'nuisance' parameters, but for large enough sample size, it is possible to estimate the distributions and still obtain the same asymptotic performances as if the distributions were known in advance. Therefore, the design of practical algorithms exhibiting 'source adaptivity' is still an interesting question.

In the preceding sections, we presented recent improvements over the results of a previous work (Fiori, 2000a) concerning hybrid independent component analysis by pseudo-polynomial adaptive activation function neurons. The use of pseudo-polynomials to approximate probability density functions allows the complexity of the neural model to be reduced with respect to other well-known approaches such as the mixture-of-kernel one without degradation of the

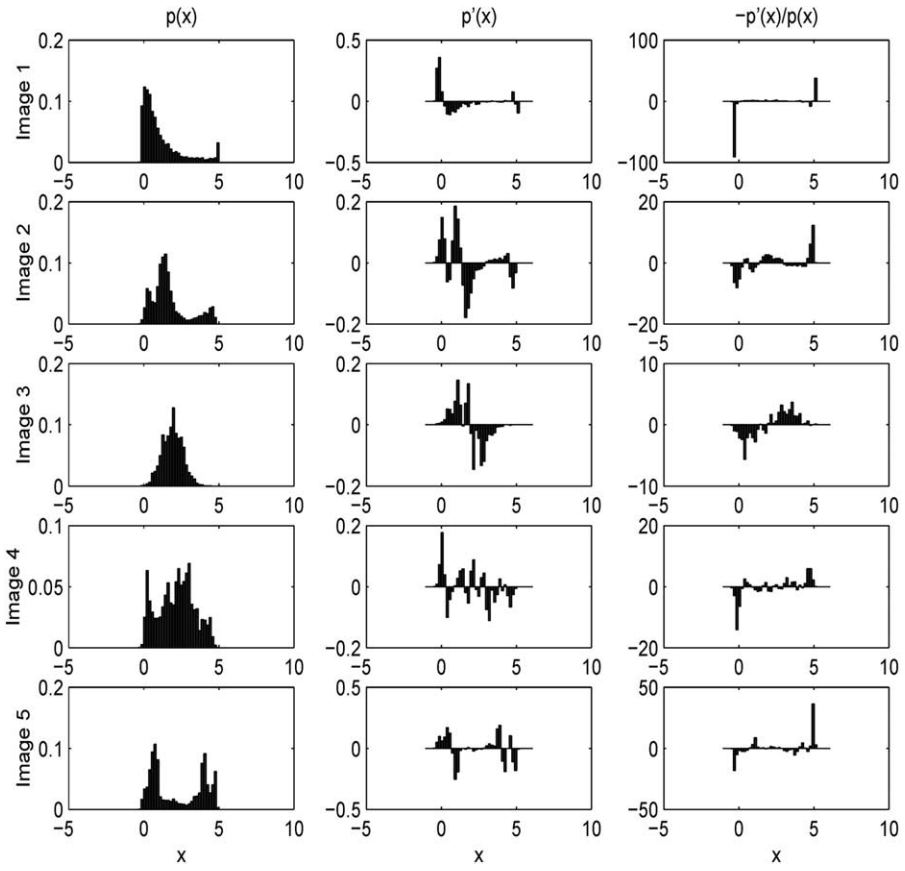

Fig. 7. Statistical distribution of five images used for the second hybrid ICA experiment.

network's performances. The proposed use of LUT-neurons as histogram approximator further lightens the neural structure and permits the overall complexity to be kept under better control.

It is worth giving a quantitative measure of such computational saving. Table 1 shows the number of operations (multiplication, divisions, generic non-linear function evaluations, and parameters to adapt) per iteration involved—and strictly necessary—for learning and computing a score function $-p'(x)/p(x)$ by using the mixture-of-kernel (MOK) approach (Xu et al., 1998), four types of pseudo-polynomial adaptive activation function neurons (FAN, Fiori, 1999, 2000a; Fiori & Bucciarelli, 2001), the classical EM algorithm (Dempster, Laird & Rubin, 1977; for a recent review, see, e.g. Ormoneit & Tresp, 1998), and the proposed LUT-neuron.

Table 1
Complexity comparison of MOK, four types of FAN, the EM algorithm and the proposed LUT neuron ($\times$: the number of multiplications, /: number of divisions, NL: number of non-linear function evaluations such as exponentiation and square-rooting)

Algorithm	$\times$	/	NL	Par.s to adapt
MOK	$14n+3$	$n+1$	$2n$	$3n$
FAN1	$7r$	1	$r+2$	$r+1$
FAN2	$7r$	1	$r+1$	$r+1$
FAN3	$9r-1$	1	1	$r+1$
FAN4	$9r-1$	1	0	$r+1$
EM	$9mn+5n$	$2mn+m+n+1$	$2mn$	$3n$
LUT	6	2	0	1

In Table 1, the variable m stands for the cardinality of training samples batch used to get a 'provisional' estimation of the quantities required to run the EM algorithm, the variable n denotes the number of the (Gaussian) kernels in the MOK and EM algorithms, while $2r+1$ is the order of polynomial in the FAN neuron. As a working hypothesis, all the constant quantities required in the learning algorithms have been supposed already computed and available in the memory. We also took into account a reasonable level of optimization of the algorithms consisting in supposing that all the necessary variable quantities can be stored in memory and need to be evaluated only once—until they are subject to change.

Note that the LUT-neuron complexity does not depend on any parameter, in fact the number of the considered operations does not change with LUT structure (e.g. with the number of bins). The computational saving provided by the LUT-neuron with respect to the considered algorithms clearly appears.

Generally speaking, the histogram method also

Fig. 8. Experiment with five grey-level images: original, mixed and recovered images.

overcomes one of the main drawbacks of polynomial-based approximators that hardly extend from mono-dimensional to multi-dimensional (multivariate) probability densities, while histogram-based and, e.g. kernel methods easily do (Fiori & Bucciarelli, 2001). However, in our approach to hybrid ICA, multi-dimensional density shaping is not required. Thus, we did not exploit this important feature.

In practical implementations, the most sensitive parameter is the sub-intervals width Δx arising from Eq. (6) once that $\mathbb{P}$ and $N_x^{\mathbb{P}}$ have been decided. The value of Δx should be chosen as a good compromise between the necessity of having an accurate representation of the density functions (which advise small intervals width values) and the need of providing smooth enough mappings $x \rightarrow \Phi(x)$ (large values of Δx). To this aim, it is worth noting that Δx plays the role of a regularization term for the non-linear part of LUT-neurons as it controls the smoothness of the approximation. The regularity of $r_{n_x} \approx p(x)$ may also be enhanced by exploiting some prior knowledge. If it is, e.g. a priori known that some properties such as $p(x) = p(-x)$ should hold, it is possible to enforce this symmetry in the algorithm of Fig. 2 by replacing the seventh line with $r_{n_x} := r_{n_x} + 1/2, r_{N_x^{\mathbb{P}}-n_x+1} := r_{N_x^{\mathbb{P}}-n_x+1} + 1/2$; in the general case this improvement is not profitable, because real-world distributions may be non-symmetric, as can be readily seen in Fig. 7.

Another sensitive point in the LUT-neuron definition is the computation of digital derivative of $p(x)$'s histogram to get an estimation of $p'(x)$. The derivation is an ill-posed problem which often requires regularization; in a digital context there exist several known ways to exploit regularization effects: once again a good choice of the interval width Δx is important, as well as the exploitation of numerical tricks as the use of symmetric (Tustin's) derivative (9) that can be proven to perform far better than simpler forward/backward (Euler's) derivative.

A question arises when deciding the number of observations N^{ob} necessary to attain a sufficiently good approximation of the true density functions. This is a complex statistical problem, but simple considerations arising from practical implementation suggest that some hundreds of samples per bin (e.g. from 100 to 500 samples/bin) do suffice to represent real-world probability density functions; this suggests the heuristic relationship $N^{\text{ob}} \geq 100\gamma N^{\mathbb{P}}$, with $\gamma \in [1, 5]$. In the compound ICA-LUT framework, however, the quantity N^{ob} often coincides to the number of available training data, thus it needs to be selected by also taking into account the network convergence speed, therefore usually $N^{\text{ob}} \gg 100N^{\mathbb{P}}$.

With this work, we aimed at proposing a solution to the hybrid ICA based on the on-line estimation of the probability density function of separating neurons output signals through a neuro-LUT implementation of histogram technique. This is a very simple and computationally light algorithm to perform such a signal processing task: it exhibits performances similar to more complex learning algorithms (Fiori, 2000a) while allowing for a noticeable computational saving.

Acknowledgements

This research has been partially supported by the Italian MURST. The author wishes to acknowledge the many interesting discussions on low-complexity (polynomial) neural structures and their efficient implementations with Professor A. Uncini (Dept. INFOCOM, University of Rome 'La Sapienza', Italy). Grateful thanks also goes to Dr L. Massaccesi (Telecom Italia, Ancona, Italy) who kindly provided some digital images for the experiments in Section 3 and Dr P. Bucciarelli (Zentral Institut für Biomedizinische Technik, University of Ulm, Germany) for the insightful discussions on density shaping by numerical techniques.

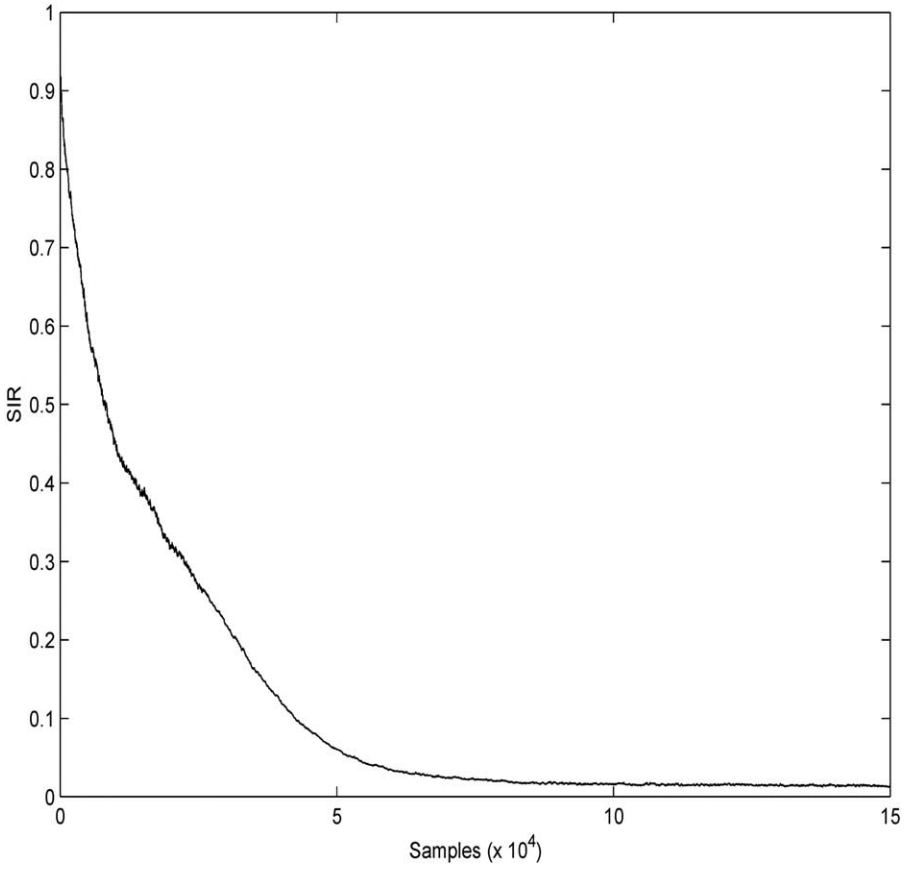

Fig. 9. Experiment with five grey-level images: SIR versus time.

References

Amari, S. -I. (1998). Natural gradient works efficiently in learning. *Neural Computation*, *10*, 251–276.

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

Belouchrani, A., & Cardoso, J.-F. (1995). Maximum likelihood source separation by the expectation–maximization technique: deterministic and stochastic implementation. In *Proc. of International Symposium on Non-Linear Theory and Applications (NOLTA)* (pp. 49–53).

Bishop, C. (1995). *Neural networks for pattern recognition*, Oxford University Press.

Cardoso, J.-F. (1998). Blind signal separation: statistical principles. In R.-W. Liu, L. Tong (Eds.), *Proceedings of the IEEE (Special issue on 'Blind identification and estimation')*, *90*(8), 2009–2026.

Cardoso, J. -F. (1999). High-order contrasts for independent component analysis. *Neural Computation*, *11* (1), 157–192.

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

Catmull, E, & Rom, R. (1974). A class of local interpolating splines. In R. E. Barnhill & R. F. Riesenfeld, *CAGD* (pp. 317–326). New York: Academic Press.

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

Chen, M. S., & Manry, M. T. (1993). Conventional modeling of the multi-layer perceptron using polynomial basis functions. *IEEE Trans. Neural Networks*, *4* (1), 164–166.

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

Comon, P., & Moreau, E. (1997). Improved contrast dedicated to blind separation in communications. In *Proc. International Conference on Acoustics, Speech and Signal Processing* (pp. 3453–3456).

Costa, S., & Fiori, S. (2001). Image compression using principal component neural networks. *Image and Vision Computing Journal (Special issue on 'Artificial neural network for image analysis and computer vision'*, *19* (9–10), 649–668.

Crespi, B. (1999). Storage capacity of non-monotonic neurons. *Neural Networks*, *12*, 1377–1389.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood for incomplete data via the EM algorithm. *Journal of the Royal Statistical Mathematical Society, B*, *39* (1), 1–38.

Fiori, S. (1999). Entropy optimization by the PFANN network: application to independent component analysis. *Network: Computation in Neural Systems*, *10* (2), 171–186.

Fiori, S. (2000a). Blind signal processing by the adaptive activation function neurons. *Neural Networks*, *13* (6), 597–611.

Fiori, S. (2000b). Blind separation of circularly distributed source signals by the neural extended APEX algorithm. *Neurocomputing*, *34* (1–4), 239–252.

Fiori, S. (2001). A theory for learning by weight flow on Stiefel–Grassman manifold. *Neural Computation*, *13*(7), 1625–1647.

Fiori, S. (2002). Notes on cost functions and estimators for 'Bussgang' adaptive blind equalization. *European Transactions on Telecommunications* (forthcoming).

Fiori, S., & Bucciarelli, P. (2001). Probability density estimation using adaptive activation function neurons. *Neural Processing Letters*, *13* (1), 31–42.

Fiori, S., & Maiolini, G. (2000). Weighted least-squares blind deconvolution of non-minimum phase systems. *IEE Proceedings—Vision, Image and Signal Processing*, *147* (6), 557–563.

Fiori, S., & Piazza, F. (2000). A general class of ψ-APEX PCA neural algorithms. *IEEE Trans. Circuits and Systems—Part I*, *47* (9), 1394–1398.

Girolami, M., & Fyfe, C. (1997). Extraction of independent signal sources using a deflationary exploratory projection pursuit network with lateral inhibition. *IEE Proceedings—Vision, Image and Signal Processing*, *14* (5), 299–306.

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

Hu, Z., & Shao, H. (1992). The study of neural network control system. *Control and Decision*, *7*, 361–366.

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

Karam, G., & Sari, H. (1990). Data pre-distortion techniques using intersymbol interpolation. *IEEE Trans. Communications*, *38* (10), 1716–1723.

Karhunen, J., Oja, E., Wang, L., Vigário, R., & Joutsensalo, J. (1997). A class of neural networks for independent component analysis. *IEEE Trans. Neural Networks*, *8* (3), 486–504.

Knuth, K. H. (1998). Bayesian source separation and localization. In A. Mohammad-Djafari (Ed.), *SPIE'98 Proceedings: Bayesian Inference for Inverse Problems*, San Diego (pp. 147–158).

Lee, T. -W., Girolami, M., & Sejnowski, T. J. (1999). Independent component analysis using an extended informax algorithm for mixed sub-Gaussian and super-Gaussian sources. *Neural Computation*, *11* (2), 417–441.

Mathews, V. J. (1991). Adaptive polynomial filtering. *IEEE Signal Processing Magazine*, 10–26.

Nishimori, Y. (1999). Learning algorithm for ICA by geodesic flows on orthogonal group. In *Proceedings of the International Joint Conference on Neural Networks*, Washington, DC, July 1999.

Oja, E. (1997). The nonlinear PCA learning rule in independent component analysis. *Neurocomputing*, *17*, 25–45.

Ormoneit, D., & Tresp, V. (1998). Averaging, maximum penalized likelihood and Bayesian estimation for improving Gaussian mixture probability density estimates. *IEEE Trans. Neural Networks*, *9* (4), 639–649.

Parra, L., Mueller, K.-R., Spence, C., Ziehe, A., & Sajda, P. (2000). Unmixing hyperspectral data. *Advances in Neural Information Processing Systems (NIPS*12)* (pp. 942–948).

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 (Eds.), *Proc. Neural Information Processing System (NIPS*9)* (pp. 613–619).

Pham, D., Garrat, P., & Jutten, C. (1992). Separation of a mixture of independent sources through a maximum likelihood approach. In *Proc. European Signal Processing Conference* (pp. 771–774).

Piazza, F., Uncini, A., & Zenobi, M. (1993). Neural networks with digital LUT activation function. In *Proc. International Joint Conference on Neural Networks (IJCNN'93)*, Nagoya (Japan), vol. 2 (pp. 1401–1404).

Puntonet, C. G., & Prieto, B. (1998). Neural net approach for blind separation of sources based on geometric properties. *Neurocomputing*, *18* (1–3), 141–164.

Roberts, S. J. (1998). Independent component analysis: source assessment and separation, a Bayesian approach. *IEE Proceedings—Vision, Image and Signal Processing*, *145* (3), 149–154.

Taleb, A., & Jutten, C. (1997). Entropy optimization—application to source separation. In *Proc. International Conference on Artificial Neural Networks* (pp. 529–534).

Trentin, E. (2001). Networks with trainable amplitude of activation. *Neural Networks*, *14*(4/5), 471–493.

Thawonmas, R., Cichocki, A., & Amari, S. -I. (1998). A cascade neural network for blind signal extraction without spurious equilibria. *IEICE Trans. Fundamentals*, *E81-A* (9), 1833–1846.

Vecci, L., Piazza, F., & Uncini, A. (1998). Learning and approximation capabilities of adaptive spline activation function neural networks. *Neural Networks*, *11* (2), 259–270.

Xu, L., Cheung, C. C., & Amari, S. -I. (1998). Learned parametric mixture based ICA algorithm. *Neurocomputing*, *22* (1–3), 69–80 Special issue on independence and artificial neural networks.

Yamada, T., & Yabuta, T. (1992). Neural network controller using autotuning method for non-linear functions. *IEEE Trans. Neural Networks*, *4*, 595–601.

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.