BLIND SOURCE SEPARATION IN NONLINEAR MIXTURES BY ADAPTIVE SPLINE NEURAL NETWORKS

Mirko Solazzi(*), Raffaele Parisi (**) and Aurelio Uncini (**)

(*) Dipartimento di Elettronica e Automatica - University of Ancona
Via Brecce Bianche, 60131 Ancona-Italy.
Fax:+39 (071) 2204464 - email: mrksol@eealab.unian.it - Internet: http://nnsp.eealab.unian.it/

(**) Dipartimento INFOCOM - University of Rome "La Sapienza"
Via Eudossiana 18, 00184 Rome - Italy.
Fax +39 (06) 4873300 email: aurel@ieee.org - Internet: http://infocom.uniroma1.it/aurel

ABSTRACT

In this paper a novel paradigm for blind source separation in the presence of nonlinear mixtures is presented and described. The proposed approach employs a neural model based on adaptive B-spline functions. Signal separation is achieved through an information maximization criterion. Experimental results and comparison with existing solutions confirm the effectiveness of the proposed architecture.

1. INTRODUCTION

Blind source separation (BSS) usually considers instantaneous *linear* memoryless mixtures [1]]-[[4]. In this case the data model is expressed by:

$$\mathbf{x}(t) = \mathbf{A} \cdot \mathbf{s}(t) \quad , \tag{1}$$

where A is a real or complex rectangular M×N matrix (M ≥ N), $\mathbf{s}(t) = [s_1(t), s_2(t)..., s_N(t)]^T$ is the vector of statistically independent sources and $\mathbf{x}(t) = [x_1(t), x_2(t)..., x_M(t)]^T$ is the vector of observed variables.

In many realistic situations, however, the basic linear mixing model (1) is not satisfactory and nonlinear mixing appears more appropriate [5]]-[[9]. The data model for nonlinear mixtures is:

$$\mathbf{x}(t) = f(\mathbf{A} \cdot \mathbf{s}(t)) , \tag{2}$$

where f represents the nonlinear mapping. As a particular case of nonlinear mixing, post-nonlinear mixing (PNL) [7] is represented by the following formula:

$$x_i(t) = f_i\left(\sum_{j=1}^{N} a_{i,j} s_j(t)\right), \tag{3}$$

where f_i are invertible and derivable nonlinear functions and $a_{i,j}$ are the entries of the mixing matrix **A**. The PNL model can be used to describe several typical scenarios, like for example nonlinearities introduced by the preamplifiers of receiving sensors in sensor arrays, under the assumption of linear mixing behavior of the environment.

The PNL model has a favorable separability property [7], meaning that the separated sources **y** can be obtained from the unknown sources in the same way of linear mixtures. Using the same notation in [7]:

$$\mathbf{y} = \mathbf{P}\Lambda\mathbf{s} + \mathbf{t} , \tag{4}$$

where **P** and L are permutation and diagonal matrices respectively and **t** is a constant translation vector.

In this paper we consider a more general nonlinear model, in which a linear mixing **B** is added to generate cross-correlation between channels (see Fig. 1, where for simplicity the number of independent source signals is equal to the number of mixtures, *M*=*N*). In this case the observed mixtures are:

$$x_k(t) = \sum_{i=1}^{N} b_{k,i} f_i\left(\sum_{j=1}^{N} a_{i,j} s_j(t)\right) \quad k = 1,2,...M \ . \tag{5}$$

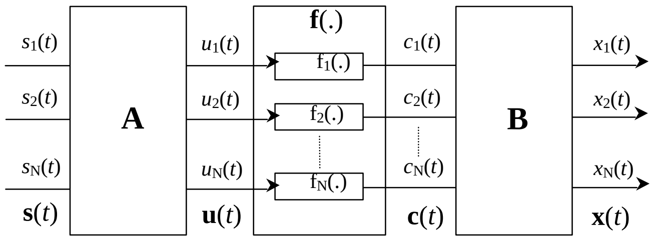

Fig. 1. Nonlinear mixing model.

2. INFOMAX CRITERION

Sources s_j (j=1,2,..,N) are assumed to be statistically independent. This means that their joint probability density function (pdf) factorizes:

$$p(s_1, s_2, .., s_N) = \prod_{j=1}^{N} p_j(s_j) , \tag{6}$$

where $p_j(s_j)$ is the marginal density of the j-th source signal.

All separation structures aim to make the outputs **y** independent. A measure of the degree of independence is the Kullback-Leibler (KL) divergence between the probability distributions $p_y(\mathbf{y})$ and $p(\mathbf{y}) = \prod_i p_i(y_i)$:

$$KL\left[p_y(\mathbf{y}) \,||\, p(\mathbf{y})\right] = \int p_y(\mathbf{y}) \log \frac{p_y(\mathbf{y})}{p(\mathbf{y})} d\mathbf{y} \ . \tag{7}$$

Minimization of the KL divergence can make the estimated source signals independent.

The KL divergence is equivalent to the Shannon's mutual information $I(\mathbf{y})$ between the components of vector $\mathbf{y}$:

$$KL\left[p_y(\mathbf{y}) \,\|\, p(\mathbf{y})\right] = I(\mathbf{y}) = \sum_{i=1}^{N} H(y_i) - H(\mathbf{y}), \tag{8}$$

where $H(\mathbf{y}) = -E\left\{\log\left(p_y(\mathbf{y})\right)\right\}$ is the entropy of $\mathbf{y}$ and $H(y_i)$ the entropy of its i-th component. The use of mutual information as a cost function is not easy because the marginal entropies $H(y_i)$ strictly depend on the marginal densities $p_i(y_i)$ of the output, which are unknown and vary during the adaptation process.

In the following, a new demixing system is introduced, based on the use of spline nonlinear functions to compensate for channel non-linearities and an INFOrmation MAXimization (INFOMAX) learning algorithm.

3. NONLINEAR BLIND SEPARATION SYSTEM

3.1 Adaptive splines

Due to their local adaptation characteristics, splines demonstrated particularly effective in the design of non-linear adaptive systems [11]]-[[12]. Let $h(x)$ be a general nonlinear function. Spline approximation consists in subdividing $h(x)$ in multiple tracts (*spans*), each one being locally approximated by a spline curve:

$$y = h(x) = \bar{h}(u,i). \tag{9}$$

Spline approximation requires a number N of *control points* Q_i and a local variable $u \in [0,1)$ for each span. In particular a cubic spline span is controlled by four control points $Q_i, Q_{i+1}, Q_{i+2}, Q_{i+3}$ (Fig. 2).

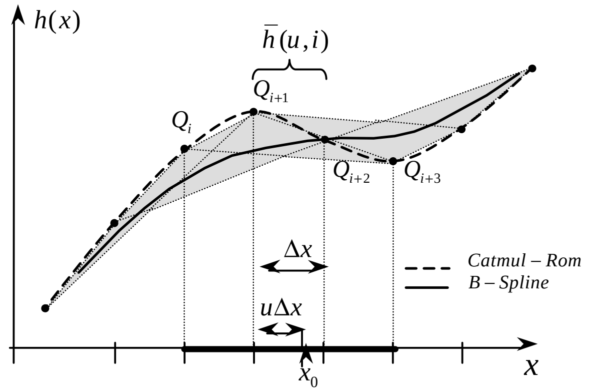

Fig. 2. Cubic-Spline interpolation of control points.

A dummy variable z is introduced:

$$z = \frac{x}{\Delta x} + \frac{N-1}{2}, \tag{10}$$

where Δx is the fixed distance between two adjacent control points. Parameters i (i=1,2,..,N) and u for each span are then derived:

$$\begin{aligned} i &= \lfloor z \rfloor \\ u &= z - i \end{aligned} \tag{11}$$

In matrix form the output can be expressed as:

$$y = \bar{h}(u,i) = \mathbf{T}_u \cdot \mathbf{M} \cdot \mathbf{Q}_i, \tag{12}$$

where:

$$\mathbf{T}_u = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}, \tag{13}$$

$$\mathbf{Q}_i = \begin{bmatrix} Q_i & Q_{i+1} & Q_{i+2} & Q_{i+3} \end{bmatrix}^T, \tag{14}$$

and:

$$\mathbf{M} = \frac{1}{2}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix} \text{ or } \mathbf{M} = \frac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}. \tag{15}$$

Matrix $\mathbf{M}$ determines the type of spline interpolation. The leftmost matrix in (15) constrains the spline to pass through all control points (Catmull-Rom spline), while the rightmost one gives a smoother characteristic and continuous derivatives (B-spline).

3.2 System architecture

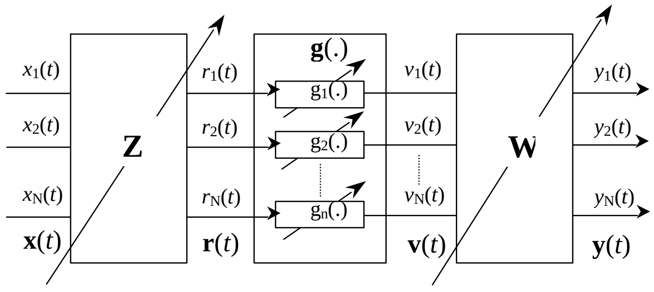

Fig. 3: Adaptive spline blind separation system.

Fig. 3 depicts the structure of the nonlinear BSS system. It consists of three stages:

1- a linear stage $\mathbf{Z}$, to reduce the cross-correlation between channels. In the particular case of PNL mixing, $\mathbf{Z} = \mathbf{I}$ (identity matrix);
2- a nonlinear layer $\mathbf{g}(.)$, to compensate for channel distortions;
3- a demixing matrix $\mathbf{W}$, conveniently determined through an adaptation process.

The parameter set for this model includes the entries of matrices $\mathbf{W}$ and $\mathbf{Z}$ and the spline control points $Q_{i+m}^{g_j}$ for each nonlinearity $g_j(.)$.

4. UNSUPERVISED LEARNING ALGORITHM

4.1 Parameter estimation

The parameters of the separating BSS system can be derived by an unsupervised learning algorithm, based on the INFOMAX approach and the gradient descent method. Minimization of $I(\mathbf{y})$ in (8) requires the computation of its gradient with respect to model parameters. In detail, to estimate the demixing matrix coefficients it is necessary to compute:

$$\frac{\partial}{\partial \mathbf{W}} I(\mathbf{y}) = \sum_{i=1}^{N} \frac{\partial}{\partial \mathbf{W}} H(y_i) - \frac{\partial}{\partial \mathbf{W}} H(\mathbf{y}), \tag{16}$$

where $H(\mathbf{y}) = -E\left\{\log\left(p_y(\mathbf{y})\right)\right\} =$

$$= E\left\{\log|\mathbf{W}| + \log\left(\prod_{i=1}^{N} \dot{g}(r_i)\right) + \log|\mathbf{Z}|\right\} + H(\mathbf{x}) \ . \tag{17}$$

In [2] the following lemma, reproduced here for convenience, was demonstrated:

Lemma 1: Let $\mathbf{x} = (x_1, x_2, ..., x_N)$ be a random variable and let $y = h(q, \mathbf{x})$ be a function of **x**, differentiable w.r.t. the non-random parameter q and such that *y* accepts a differentiable pdf $p_y(y)$. Then:

$$\frac{dH(y)}{dq} = -E\left[y_y\left[h(q,\mathbf{x})\right]\frac{dh(q,\mathbf{x})}{dq}\right] \tag{18}$$

where $y_y(y) = p'_y(y)/p_y(y)$ is called *score function* of *y*. ÿ

Using lemma 1 it can be shown that:

$$\frac{\partial}{\partial \mathbf{W}} I(\mathbf{y}) = -E\left[\Psi(\mathbf{y})^T \mathbf{v}^T\right] - \left[\mathbf{W}^T\right]^{-1}, \tag{19}$$

where $\Psi(\mathbf{y}) = \left[y_{y_1}(y_1), y_{y_2}(y_2), ..., y_{y_N}(y_N)\right]$.

The derivatives of the mutual information w.r.t. the parameters of spline functions $g(.)$ are:

$$\frac{\partial}{\partial Q_{i+m}^{g_j}} I(\mathbf{y}) = \sum_{i=1}^{N} \frac{\partial}{\partial Q_{i+m}^{g_j}} H(y_i) - \frac{\partial}{\partial Q_{i+m}^{g_j}} H(\mathbf{y}) \ . \tag{20}$$

Writing $y_i = \sum_{k=1}^{N} w_{ik} g_k\left(\sum_{l=1}^{N} z_{kl} x_l\right)$ and using lemma 1 again, the first term in the r.h.s. of (20) becomes:

$$\sum_{i=1}^{N} \frac{\partial}{Q_{i+m}^{g_j}} H(y_i) = -E\left[\sum_{i=1}^{N} y_{y_i}(y_i) \frac{\partial \sum_{k=1}^{N} w_{ik} g_k(r_k)}{\partial Q_{i+m}^{g_j}}\right] =$$

$$= -E\left[\left(\sum_{i=1}^{N} y_{y_i}(y_i) w_{ij}\right)\left(\mathbf{T}_{u(r_j)} \cdot \mathbf{M}_m\right)\right], \tag{21}$$

while for the second term:

$$\frac{\partial}{\partial Q_{i+m}^{g_j}} H(\mathbf{y}) = E\left[\frac{\partial \log\left(\dot{g}_k(r_k)\right)}{\partial Q_{i+m}^{g_j}}\right] = \left.\frac{\dot{\mathbf{T}}_u \cdot \mathbf{M}_m}{\dot{\mathbf{T}}_u \cdot \mathbf{M} \cdot \mathbf{Q}_i^{g_j}}\right|_{\substack{u=u(r_j)\\ i=i(r_j)}}, \tag{22}$$

where $\dot{\mathbf{T}}_u = \begin{bmatrix} 3u^2 & 2u & 1 & 0 \end{bmatrix}$, $m \in (0,1,2,3)$ and $\mathbf{M}_\mathrm{m}$ is the *m*-th column of matrix **M**.

Combination of (21) and (22) gives:

$$\frac{\partial}{\partial Q_{i+m}^{g_j}} I(\mathbf{y}) = -E\left[\left(\Psi(\mathbf{y})^T \mathbf{W}\right)_j \left(\mathbf{T}_{u(r_j)} \cdot \mathbf{M}_m\right)\right] - \left.\frac{\dot{\mathbf{T}}_u \cdot \mathbf{M}_m}{\dot{\mathbf{T}}_u \cdot \mathbf{M} \cdot \mathbf{Q}_i^{g_j}}\right|_{\substack{u=u(r_j)\\ i=i(r_j)}} . \tag{23}$$

Finally, derivatives w.r.t. **Z** must be computed:

$$\frac{\partial}{\partial \mathbf{Z}} I(\mathbf{y}) = \sum_{i=1}^{N} \frac{\partial}{\partial \mathbf{Z}} H(y_i) - \frac{\partial}{\partial \mathbf{Z}} H(\mathbf{y}) \ . \tag{24}$$

Writing $r_k = \sum_{l=1}^{N} z_{kl} x_l$ we have:

$$\sum_{i=1}^{N} \frac{\partial}{\partial \mathbf{Z}} H(y_i) = -E\left[\sum_{i=1}^{N} y_{y_i}(y_i) \frac{\partial \sum_{k=1}^{N} w_{ik} g_k(r_k)}{\partial \mathbf{Z}}\right] =$$

$$= -E\left[\sum_{i=1}^{N} y_{y_i}(y_i) \sum_{k=1}^{N} w_{ik} \dot{g}_k(r_k) \frac{\partial r_k}{\partial \mathbf{Z}}\right], \tag{25}$$

and:

$$\frac{\partial}{\partial \mathbf{Z}} H(\mathbf{y}) = E\left[\frac{\partial}{\partial \mathbf{Z}} \log\left(\prod_{k=1}^{N} \dot{g}_k(r_k)\right) + \frac{\partial}{\partial \mathbf{Z}} \log|\mathbf{Z}|\right] =$$

$$= E\left[\sum_{k=1}^{N} \frac{\ddot{g}_k(r_k)}{\dot{g}_k(r_k)} \frac{\partial r_k}{\partial \mathbf{Z}}\right] + \left[\mathbf{Z}^T\right]^{-1} . \tag{26}$$

Combining the two terms, (23) becomes:

$$\frac{\partial}{\partial \mathbf{Z}} I(\mathbf{y}) = -E\left[\dot{\mathbf{G}}\mathbf{W}^T \Psi(\mathbf{y})^T \cdot \mathbf{x}^T\right] - E\left[\mathbf{G}(\mathbf{r})^T \cdot \mathbf{x}^T\right] - \left[\mathbf{Z}^T\right]^{-1} \tag{27}$$

where:

$$\mathbf{G}(\mathbf{r}) = \begin{bmatrix} \dfrac{\ddot{g}_1(r_1)}{\dot{g}_1(r_1)} & \dfrac{\ddot{g}_2(r_2)}{\dot{g}_2(r_2)} & \cdots & \dfrac{\ddot{g}_N(r_N)}{\dot{g}_n(r_N)} \end{bmatrix}, \tag{28}$$

$$\ddot{g}_j(r) = \ddot{\bar{g}}_j(u,i) = \begin{bmatrix} 6u & 2 & 0 & 0 \end{bmatrix} \cdot \mathbf{M} \cdot \mathbf{Q}_i^{g_j} = \ddot{\mathbf{T}}_u \cdot \mathbf{M} \cdot \mathbf{Q}_i^{g_j}, \tag{29}$$

$$\dot{g}_j(r) = \dot{\bar{g}}_j(u,i) = \dot{\mathbf{T}}_u \cdot \mathbf{M} \cdot \mathbf{Q}_i^{g_j}, \tag{30}$$

and $\dot{\mathbf{G}}$ is a diagonal matrix with elements $g_{jj} = \dot{g}_j(r)$.

4.2 Estimation of score functions

The derivatives of $I(\mathbf{y})$ with respect to model parameters point out the relevance of score functions. Following the approach described in [7], it is convenient to introduce convenient nonlinear parametric functions $f_j(\mathbf{Q}^j, y_j)$, in order to approximate each score function $y_{y_j}(y_j)$. The parameter vector $\mathbf{Q}^j$ is estimated by minimizing the following cost function:

$$e_j = \frac{1}{2} E\left[\left| f_j(\mathbf{Q}^j, y_j) - y_{y_j}(y_j) \right|^2 \right], \tag{31}$$

according to a gradient descent algorithm:

$$\mathbf{Q}^j[p+1] = \mathbf{Q}^j[p] - m_Q \left(\frac{\partial e_j}{\partial \mathbf{Q}^j} \right)_{\mathbf{Q}^j = \mathbf{Q}^j[p]} . \tag{32}$$

The gradient of e_j w.r.t. parameters $\mathbf{Q}^j$ is:

$$\frac{\partial e_j}{\partial \mathbf{Q}^j} = E\left[\frac{\partial f_j(\mathbf{Q}^j, y_j)}{\partial \mathbf{Q}^j} \left(f_j(\mathbf{Q}^j, y_j) - y_{y_j}(y_j) \right) \right]. \tag{33}$$

Unfortunately (33) contains the score functions $y_{y_j}(y_j)$, that are unknown. To overcome this problem, the following lemma can be used (see [2] for the proof):

Lemma 2: Let $\mathbf{x}$ be a random variable, and let $y_X(x)$ be its score function. If f is a differentiable function satisfying $\lim_{|x|\to+\infty} p_X(x) f(x) = 0$, then:

$$E\left[f(x) y_X(x) \right] = -E\left[f'(x) \right].$$

Lemma 2 can be applied to (33), giving:

$$\frac{\partial e_j}{\partial \mathbf{Q}^j} = E\left[\frac{\partial f_j(\mathbf{Q}^j, y_j)}{\partial \mathbf{Q}^j} f_j(\mathbf{Q}^j, y_j) + \frac{\partial^2 f_j(\mathbf{Q}^j, y_j)}{\partial y_j \partial \mathbf{Q}^j} \right]. \tag{34}$$

In the present work, functions $f_j(\mathbf{Q}^j, y_j)$ are implemented by using the spline approximation scheme defined above. So, each nonlinear function is locally defined by the following expressions:

$$f_j(\mathbf{Q}^j, y_j) = \bar{f}_j(u,i) = \mathbf{T}_u \cdot \mathbf{M} \cdot \mathbf{Q}_i^{f_j}, \tag{35}$$

$$\frac{\partial e_j}{\partial Q_{i+m}^{f_j}} = \frac{\partial \bar{f}_j(u,i)}{\partial Q_{i+m}^{f_j}} \bar{f}_j(u,i) + \frac{\partial^2 \bar{f}_j(u,i)}{\partial y_j \partial Q_{i+m}^{f_j}} =$$

$$= \left(\mathbf{T}_u \cdot \mathbf{M}_m \right)\left(\mathbf{T}_u \cdot \mathbf{M} \cdot \mathbf{Q}_i^{f_j} \right) + \frac{1}{\Delta y} \dot{\mathbf{T}}_u \cdot \mathbf{M}_m \Bigg|_{\substack{u=u(y_j) \\ i=i(y_j)}} . \tag{36}$$

4.3 Learning algorithm

The learning algorithm can be summarized as in the following:

1- Initialize $\mathbf{W}$ and $\mathbf{Z}$ to the identity matrix $\mathbf{I}$, spline functions g(.) to linear functions and score functions $f_j(\mathbf{Q}^j, y_j)$ to zero in the whole domain.

2- Update the control points of the score functions by formula:

$$Q_{i+m}^{f_j}[p+1] = Q_{i+m}^{f_j}[p] - m_f \frac{\partial e_j}{\partial Q_{i+m}^{f_j}}, \tag{37}$$

and impose:

$$Q_1^{f_j} \geq Q_2^{f_j} \geq \ldots \geq Q_N^{f_j},$$

to ensure the monotonously not increasing characteristic of the score functions, sorting the Q^j vectors.

3- Adjust $\mathbf{W}$, using an equivariant algorithm by post-multiplying (19) by $\mathbf{W}^T\mathbf{W}$ [4]:

$$\mathbf{W}[p+1] = \left[\mathbf{I} + h_w \mathbf{K} \right] \mathbf{W}[p], \tag{38}$$

where the entries of matrix $\mathbf{K}$ are:

$$k_{i,j} = \begin{cases} (1 - 0.5 y_i^2) & \text{if } i = j \\ f_i(\mathbf{Q}^i, y_i) y_j & \text{otherwise} \end{cases} \tag{39}$$

Normalize $\mathbf{W}$ using: $\mathbf{W} = \mathbf{W} / \max\left| w_{i,j} \right|$.

4- Locally update the control points of functions g(.):

$$Q_{i+m}^{g_j}[p+1] = Q_{i+m}^{g_j}[p] - h_g \frac{\partial I(\mathbf{y})}{\partial Q_{i+m}^{g_j}}, \tag{40}$$

and impose $Q_1^{g_j} < Q_2^{g_j} < \ldots < Q_N^{g_j}$.

5- Adjust matrix $\mathbf{Z}$, using an equivariant algorithm by post-multiplying (27) by $\mathbf{Z}^T\mathbf{Z}$ [4]:

$$\mathbf{Z}[p+1] = \left\{ \mathbf{I} + h_z \left[\mathbf{I} + \mathbf{G}(\mathbf{r})^T \cdot \mathbf{r}^T + \dot{\mathbf{G}} \mathbf{W}^T \mathbf{H}(\mathbf{y})^T \cdot \mathbf{r}^T \right] \right\} \mathbf{Z}[p] . \tag{41}$$

6- Repeat steps from 2) to 5).

5. EXPERIMENTAL RESULTS

Several experimental tests have been performed to validate the proposed approach. In particular, in this section the performance of the spline-based approach are compared to those of the RBF network separating system described in [8], in two typical problems.

Experiment 1: Consider a two-channel nonlinear mixture with cubic nonlinearity :

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \mathbf{B} \begin{bmatrix} (.)^3 \\ (.)^3 \end{bmatrix} \left(\mathbf{A} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} \right), \tag{42}$$

where matrices $\mathbf{B}$ and $\mathbf{A}$ are defined as:

$$\mathbf{B} = \begin{bmatrix} 0.25 & 0.86 \\ -0.86 & 0.25 \end{bmatrix} ; \quad \mathbf{A} = \begin{bmatrix} 0.5 & 0.9 \\ -0.9 & 0.5 \end{bmatrix} .$$

The source vector $\mathbf{s}(t)$ consists of an amplitude-modulated signal and a sinusoidal signal:

$$\mathbf{s}(t) = \left[0.5\left(1 + \sin\left(6pt\right)\right)\cos\left(100pt\right); \quad \sin(20pt) \right]^T \tag{43}$$

Figures 4, 5 and 6 show the input signals, the nonlinear mixtures and the separated signals respectively. The learning process took less than one minute on a Pentium II 300MHz workstation. Learning rates of the order of 10^{-4} were used. As showed in Fig.6, the proposed algorithm is able to clearly separate the nonlinear mixtures.

Experiment 2: In this experiment a four channel nonlinear mixture was considered, having the following sigmoidal nonlinearity :

$$\mathbf{x}(t) = \mathbf{B} \cdot \tanh\left(\mathbf{A} \cdot \mathbf{s}(t)\right) . \qquad (44)$$

The mixing matrices **A** and **B** are nonsingular and chosen as:

$$\mathbf{A} = \begin{bmatrix} 0.2844 & 0.5828 & 0.4329 & 0.5298 \\ 0.4692 & 0.4235 & 0.2259 & 0.6405 \\ 0.0648 & 0.5155 & 0.5798 & 0.2091 \\ 0.9883 & 0.3340 & 0.7604 & 0.3798 \end{bmatrix};$$

$$\mathbf{B} = \begin{bmatrix} 0.7349 & 0.1556 & 0.4902 & 0.4507 \\ 0.6873 & 0.1911 & 0.8159 & 0.4122 \\ 0.3461 & 0.4225 & 0.4608 & 0.9016 \\ 0.1660 & 0.8560 & 0.4574 & 0.0056 \end{bmatrix}.$$

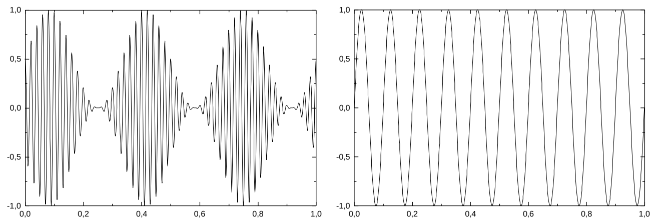
Fig. 4. Experiment 1: source signals.

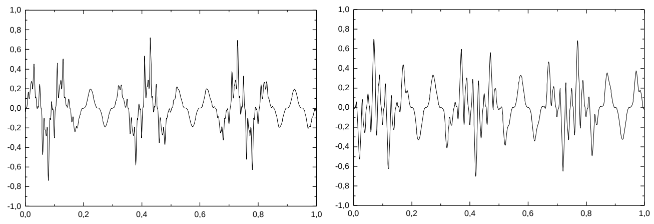
Fig. 5. Experiment 1: nonlinear mixtures of source signals.

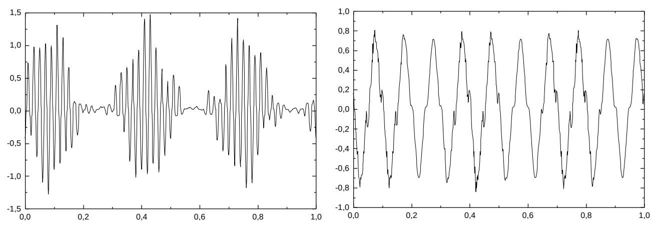
Fig. 6. Experiment 1: separated signals.

The source vector **s**(t) consists of a binary signal, a sinusoid, a saw-toothed wave (*ramp*) with period T=0.1667 and a high-frequency carrier (Fig.7):

$$\mathbf{s}(t) = \left[\operatorname{sgn}[\cos(12pt)];\sin(8pt);ramp(t,T);\sin(60pt)\right]^T \qquad (45)$$

Fig. 8 shows the nonlinear mixtures. The learning process took about 80 secs. and produced the separated signals of Fig.9.

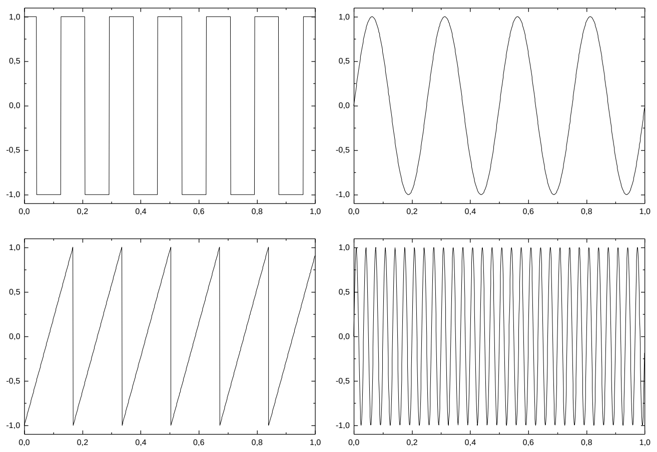
Fig. 7. Experiment 2: source signals.

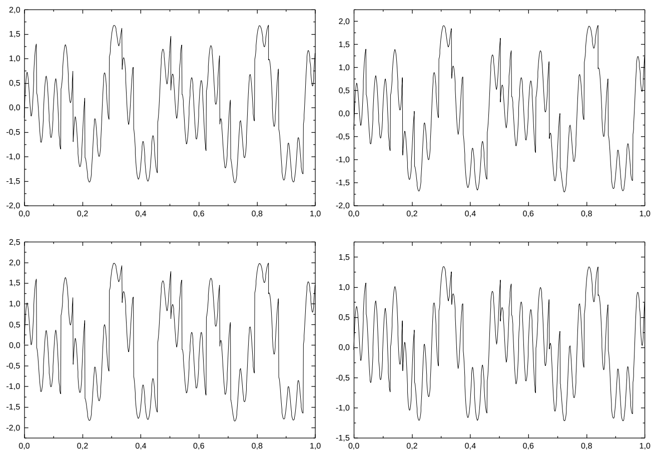
Fig. 8. Experiment 2: nonlinear mixtures of source signals.

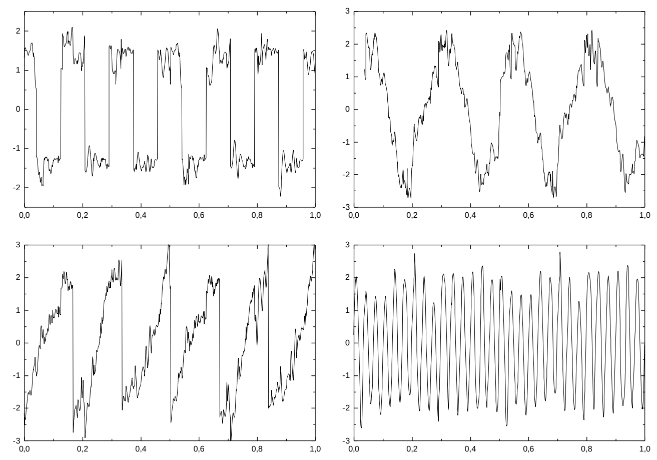
Fig. 9. Experiment 2: separated signals.

6. CONCLUSION

A new model for blind demixing of nonlinear mixtures based on flexible B-spline activation function neuron has been proposed.

Based on a gradient ascent method on the basis of the INFOMAX criterion with score function estimation, a suitable learning algorithm of parameters of the nonlinear separating model has been derived.

Due to local adaptation capabilities of spline functions, this model is characterized by fast learning convergence rate and it can be applied or real-time signal separation. Another relevant feature is that separation of strong nonlinear mixtures is possible without any knowledge of original source signals, as experimentally demonstrated.

7. REFERENCES

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

[2] L. Xu, C.C. Cheung, H.H. Yang, S. Amari, "Independent component analysis by the information theoretic approach with mixture of densities", *Proc. of IEEE ICNN-97*, Houston TX, USA, Vol. 3, pp. 1821-1826, June 1997.

[3] J.F. Cardoso "Blind signal separation: statistical principles", *Proc. IEEE*, Vol.86, No. 10, October 1998.

[4] S.I. Amari and A. Cichocki, "Adaptive blind signal processing-Neural network approaches", *Proc. IEEE*, Vol.86, No. 10, October 1998.

[5] P. Pajunen, A. Hyvarinen and J. Karhunen, "Nonlinear blind sources separation by self-organizing maps", *Proc. ICONIP*, Hong Kong, Vol.2 pp.1207-1210, Sept. 1996.

[6] T.-W. Lee, B. Koehler and R. Orglmeister, "Blind source separation of nonlinear mixing models", *Neural networks for Signal Processing* VII, 1997.

[7] A. Taleb and C. Jutten, "Source separation in post nonlinear mixtures", *IEEE Trans. on Signal Processing*, Vol. 47, No. 10, pp. 2807-2820, October 1999.

[8] Y. Tan, J.Wang and J.M.Zurada, "Nonlinear blind source separation using a radial basis function", *IEEE Trans. on Neural Networks*, Vol.12, No.1, pp. 124-134, January 2001.

[9] M. Solazzi, F. Piazza, A. Uncini, "Nonlinear blind source separation by spline neural networks", *ICASSP 2001*, Salt Lake City, USA, May 8-11, 2001.

[10] S. Guarnieri, F. Piazza and A. Uncini, "Multilayer feedforward networks with adaptive spline activation function", *IEEE Trans. on Neural Network*, Vol. 10, No. 3, pp.672-683, May 1999

[11] L. Vecci, F. Piazza, A. Uncini, "Learning and approximation capabilities of adaptive spline activation function neural networks", *Neural Networks*, Vol. 11, No. 2, pp.259-270, March 1998.

[12] A. Uncini, L. Vecci, P. Campolucci and F. Piazza, "Complex-valued neural networks with adaptive spline activation function for digital radio links nonlinear equalization", *IEEE Trans. on Signal Processing*, Vol. 47, No. 2, February 1999.