Published in Signal Processing (Elsevier), vol. 85, no. 5, pp. 975-995, May 2005.

A general approach for mutual information minimization and its application to Blind Source Separation

Massoud Babaie-Zadeh[1] and Christian Jutten[2]

Abstract

In this paper, a non-parametric "gradient" of the mutual information is first introduced. It is used for showing that mutual information has no local minima. Using the introduced "gradient", two general gradient based approaches for minimizing mutual information in a parametric model are then presented. These approaches are quite general, and principally they can be used in any mutual information minimization problem. In blind source separation, these approaches provide powerful tools for separating any complicated (yet separable) mixing model. In this paper, they are used to develop algorithms for separating four separable mixing models: linear instantaneous, linear convolutive, Post Non-Linear (PNL) and Convolutive Post Non-Linear (CPNL) mixtures.

Index Terms

Mutual Information, Information theoretic learning, Gradient of mutual information, Score Function Difference (SFD), Independent Component Analysis (ICA), Blind Source Separation (BSS), Convolutive mixtures, Post Non-Linear (PNL) mixtures, Convolutive Post Non-Linear (CPNL) mixtures,

I. Introduction

Let $s_1(t), s_2(t), \dots, s_M(t)$ be M statistically independent source signals, from which only N different mixtures $x_1(t), x_2(t), \dots, x_N(t)$ have been observed, *i.e.* :

$$x_i(t) = \mathcal{F}_i\big(s_1(t), \dots, s_N(t)\big), \quad i = 1, \dots, N \tag{1}$$

where $\mathcal{F}_i$ denotes the unknown mixing system, which may be nonlinear and/or have memory. In vector form, the above equation is written as $\mathbf{x}(t) = \mathcal{F}\big(\mathbf{s}(t)\big)$, where $\mathbf{s}(t) \triangleq (s_1(t), \dots, s_M(t))^T$ and $\mathbf{x}(t) \triangleq (x_1(t), \dots, x_M(t))^T$ are source and observation vectors, respectively. Then, the goal of Blind Source Separation (BSS) is to recover the original source signals by knowing only these observations. The problem is called *Blind* since there is no or very little information about the sources or about the mixing system. This problem has applications in different areas including feature extraction, brain imaging, telecommunications, speech

[1] Electrical engineering departement, Sharif university of technology, Tehran, Iran.

[2] Laboratoire des Images et des Signaux (LIS), UMR CNRS 5083, Institut National Polytechnique de Grenoble and Université Joseph Fourier, Grenoble, France.

This work has been partially funded by the European project BLISS (IST-1999-14190), by Sharif university of technology, and by Iran Research Telecom Center (ITRC). Ch. Jutten is professor at Université Joseph Fourier. Authors' email addresses are `mbzadeh@yahoo.com` and `Christian.Jutten@inpg.fr`.

enhancement, *etc* [1]. Throughout this paper, it is always assumed that the number of observations is equal to the number of sources, that is, $M = N$.

Since the sole assumption is the independence of sources, the basic idea in blind source separation consists in estimating a system $\mathcal{G}$, only from the observed data $\mathbf{x}(t)$, such that the components of $\mathbf{y}(t) = \mathcal{G}(\mathbf{x}(t))$ are statistically independent. This method, based on statistical independence, constitutes a generic approach called Independent Component Analysis (ICA). In general (nonlinear) case, it can be shown [2] that ICA does not lead to BSS. However, if some structural constraints are imposed on mixing and separating systems, the ICA approach may result in BSS, with eventually a few indeterminacies. For example, when both mixing and separating systems are assumed to be linear and memoryless (*i.e.* $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t)$ and $\mathbf{y}(t) = \mathbf{B}\mathbf{x}(t)$, where $\mathbf{A}$ and $\mathbf{B}$ are $N \times N$ regular matrices), then we have the well-known instantaneous linear mixtures, for which the equivalency of ICA and BSS has been already proved [3]: if the components of $\mathbf{y}$ are independent, and if there is at most one Gaussian source, then the outputs will be equal to the source signals up to a scale and a permutation indeterminacy.

Consequently, for using the ICA approach for source separation, we impose a structural constraint [1] on mixing and separating systems. Then, the separating system is a parametric mapping:

$$\mathbf{y}(t) = \mathcal{G}(\mathbf{x}(t); \boldsymbol{\theta}) \tag{2}$$

where $\mathcal{G}$ is a "known" separating system with "unknown" parameters $\boldsymbol{\theta}$. It is also required that this mixing-separating model be "separable", that is, the independence of the outputs (ICA) insures the separation of the sources (BSS). Under these conditions, the problem of source separation reduces to finding the parameter $\boldsymbol{\theta}$ that maximizes the independence of the outputs.

To measure the degree of independence of the outputs, their mutual information may be used. The mutual information of the random variables $y_1, y_2, \ldots, y_N$ is defined as [6]:

$$I(\mathbf{y}) = \int_{\mathbf{y}} p_{\mathbf{y}}(\mathbf{y}) \ln \frac{p_{\mathbf{y}}(\mathbf{y})}{\prod_i p_{y_i}(y_i)} d\mathbf{y} \tag{3}$$

where $\mathbf{y} = (y_1, \ldots, y_N)^T$. This is nothing but the Kullback-Leibler divergence between $p_{\mathbf{y}}(\mathbf{y})$ and $\prod_i p_{y_i}(y_i)$. It is well-known that $I(\mathbf{y})$ is always nonnegative and vanishes if and only if the components of $\mathbf{y}$ are independent. Consequently, the solution of the BSS problem for the model (2) is the vector $\boldsymbol{\theta}$ that minimizes $I(\mathbf{y})$. Mutual information has been already used in BSS by several authors (*e.g.* [3], [7], [8], [4]).

For minimizing $I(\mathbf{y})$ with respect to $\boldsymbol{\theta}$, gradient based methods may be applied. To construct such a method, knowing the "differential" of the mutual information, that is, its variation resulting from a small deviation in its argument, is very useful. Such a non-parametric differential has been recently proposed [9] (see also Section III).

The aim of this paper is to consider two general gradient based approaches (which are called "gradient" and "Minimization-Projection" approaches) for minimizing mutual information of $\mathbf{y}$ in a model like (2). These new approaches, and the model (2), are quite general and can be used in any mutual information minimization problem. In BSS, they provide powerful tools for having a unifying approach for separating any complicated (yet separable) mixing model. As some examples, we use these approaches for separating four separable mixing models (see Section VI). Moreover, it is shown in this paper that mutual information has no "local minimum" (see Theorem 2 and Remark 2 after the theorem).

[1] Other regularization approaches have also been proposed, based on smooth mappings, *i.e.* multilayer perceptrons [4] or based on Bayesian models with ensemble learning [5].

The paper is organized as follows. In Section II, multi-variate score functions and Score Function Difference (SFD) are introduced and their properties are discussed. Using SFD, the "gradient" of mutual information is presented in Section III. Using properties of multi-variate score functions and SFD, we propose estimation algorithms for estimating multi variate score functions in Section IV. Two general approaches for minimizing mutual information are then introduced in Section V. These approaches are used for blind separating four separable models, presented in Section VI: linear instantaneous mixtures in Section VII, linear convolutive mixtures in Section VIII, Post Non-Linear (PNL) mixtures in Section IX and Convolutive Post Non-Linear (CPNL) mixtures in Section X.

II. Multi-variate score functions

Multi-variate score functions have been first introduced in [10], by extending the concept of score function of a random variable to random vectors. The "gradient" of mutual information can be expressed in terms of these score functions (see Theorem 1). This section starts with a review of definitions, and continues by discussing their properties.

A. Definitions

Recall the definition of the score function of a random variable:

Definition 1 (Score Function): The score function of a scalar random variable x is the opposite of the log derivative of its density, that is, $\psi_x(x) \triangleq -\frac{d}{dx}\ln p_x(x) = -\frac{p'_x(x)}{p_x(x)}$, where p_x denotes the probability density function (PDF) of x.

In conjunction with this definition, we define two different types of score functions for a random vector $\mathbf{x} = (x_1, \ldots, x_N)^T$:

Definition 2 (MSF): The Marginal Score Function (MSF) of $\mathbf{x}$ is the vector of score functions of its components, that is, $\boldsymbol{\psi}_{\mathbf{x}}(\mathbf{x}) \triangleq (\psi_1(x_1), \ldots, \psi_N(x_N))^T$, where:

$$\psi_i(x_i) \triangleq -\frac{d}{dx_i}\ln p_{x_i}(x_i) = -\frac{p'_{x_i}(x_i)}{p_{x_i}(x_i)} \tag{4}$$

Definition 3 (JSF): The Joint Score Function (JSF) of $\mathbf{x}$ is the gradient of "$-\ln p_{\mathbf{x}}(\mathbf{x})$", that is $\boldsymbol{\varphi}_{\mathbf{x}}(\mathbf{x}) \triangleq (\varphi_1(\mathbf{x}), \ldots, \varphi_N(\mathbf{x}))^T$, where:

$$\varphi_i(\mathbf{x}) \triangleq -\frac{\partial}{\partial x_i}\ln p_{\mathbf{x}}(\mathbf{x}) = -\frac{\frac{\partial}{\partial x_i}p_{\mathbf{x}}(\mathbf{x})}{p_{\mathbf{x}}(\mathbf{x})} \tag{5}$$

We will see that the difference of these two score functions contains a lot of information about the independence of the components of a random vector. Thus, it worths to give it a formal name:

Definition 4 (SFD): The Score Function Difference (SFD) of $\mathbf{x}$ is the difference of its MSF and JSF, that is, $\boldsymbol{\beta}_{\mathbf{x}}(\mathbf{x}) \triangleq \boldsymbol{\psi}_{\mathbf{x}}(\mathbf{x}) - \boldsymbol{\varphi}_{\mathbf{x}}(\mathbf{x})$.

B. Properties

The first property presented here points out that SFD contains information about the independence of the components of a random vector:

Property 1: The components of a random vector $\mathbf{x} = (x_1, \ldots, x_N)^T$ are independent if and only if $\boldsymbol{\beta}_{\mathbf{x}}(\mathbf{x}) \equiv \mathbf{0}$, that is, if and only if $\boldsymbol{\varphi}_{\mathbf{x}}(\mathbf{x}) = \boldsymbol{\psi}_{\mathbf{x}}(\mathbf{x})$.

The proof is simple and can be found in [10].

Property 2: For a random vector $\mathbf{x} = (x_1, \ldots, x_N)^T$:

$$\beta_i(\mathbf{x}) = \frac{\partial}{\partial x_i} \ln p(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N | x_i) \tag{6}$$

where $\beta_i(\mathbf{x})$ denotes the i-th component of the SFD of $\mathbf{x}$.

Proof: Without loss of generality, let $i = 1$. Then:

$$\begin{aligned}\beta_1(\mathbf{x}) &= \psi_1(x_1) - \varphi_1(\mathbf{x}) = \frac{\partial}{\partial x_1} \ln p_{\mathbf{x}}(\mathbf{x}) - \frac{\partial}{\partial x_1} \ln p_{x_1}(x_1) = \frac{\partial}{\partial x_1} \ln \frac{p_{\mathbf{x}}(x_1, \ldots, x_N)}{p_{x_1}(x_1)} \\ &= \frac{\partial}{\partial x_1} \ln p_{x_2, \ldots, x_N}(x_2, \ldots, x_N \mid x_1)\end{aligned} \tag{7}$$

■

It is interesting to note the relationship between this property and Property 1. For example, for the two dimensional case, (6) becomes:

$$\beta_1(x_1, x_2) = \frac{\partial}{\partial x_1} \ln p(x_2 \mid x_1) \tag{8}$$

$$\beta_2(x_1, x_2) = \frac{\partial}{\partial x_2} \ln p(x_1 \mid x_2) \tag{9}$$

In other words, $\beta_1(x_1, x_2) = 0$ if $p(x_2|x_1)$ does not depend on x_1, that is, when x_1 and x_2 are independent. For the N-dimensional case too, $\beta_i(\mathbf{x}) = 0$ if "x_i and the other components of $\mathbf{x}$ are independent", that is, if $p(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N|x_i) = p(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)$, or $p(\mathbf{x}) = p(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)p(x_i)$, $\forall i = 1, \ldots, n$.

The next property is, in fact, the generalization of a similar property of the score function of a scalar random variable [11], [12].

Property 3: Let $\mathbf{x}$ be a random vector with density $p_{\mathbf{x}}$ and JSF $\boldsymbol{\varphi}_{\mathbf{x}}$ (whose i-th component is denoted by $\varphi_i(\mathbf{x})$). Moreover, let $f(\mathbf{x})$ be a multivariate function with continuous partial derivatives such that:

$$\lim_{x_i \to \pm\infty} \int_{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N} f(\mathbf{x}) p_{\mathbf{x}}(\mathbf{x})\, dx_1 \cdots dx_{i-1}\, dx_{i+1} \cdots dx_N = 0 \tag{10}$$

then:

$$E\left\{f(\mathbf{x})\varphi_i(\mathbf{x})\right\} = E\left\{\frac{\partial f}{\partial x_i}(\mathbf{x})\right\} \tag{11}$$

Note that the condition (10) is not too restrictive for usual sources, because for most physical signals, $p_{\mathbf{x}}(\mathbf{x})$ decreases rapidly when $||\mathbf{x}||$ goes to infinity. Indeed, most real signals are "bounded" (finite energy), and for them (10) holds.

Proof: Without loss of generality, let $i = 1$. We write:

$$E\{f(\mathbf{x})\varphi_1(\mathbf{x})\} = \int f(\mathbf{x})\varphi_1(\mathbf{x}) p_{\mathbf{x}}(\mathbf{x}) d\mathbf{x} = -\int_{x_2, \ldots, x_N} \int_{x_1} f(\mathbf{x}) \frac{\partial p_{\mathbf{x}}(\mathbf{x})}{\partial x_1}\, dx_1\, dx_2 \cdots dx_N \tag{12}$$

Now, by applying integration by parts for the inner integral and using (10) the desired relation will be obtained. ■

Corollary 1: For bounded random vector $\mathbf{x}$, $E\left\{\varphi_i(\mathbf{x})\, x_j\right\}$ is equal to one for $i = j$ and is equal to zero for $i \neq j$. In matrix form, $E\left\{\boldsymbol{\varphi}_{\mathbf{x}}(\mathbf{x})\, \mathbf{x}^T\right\} = \mathbf{I}$, where $\mathbf{I}$ denotes the identity matrix.

Corollary 2: Suppose we would like to estimate $\varphi_i(\mathbf{x})$ by a parametric function $f(\mathbf{x}; \mathbf{w})$, where $\mathbf{w} = (w_1, \dots, w_K)^T$ is the parameters vector. Then:

$$\underset{\mathbf{w}}{\operatorname{argmin}}\, E\left\{\left(\varphi_i(\mathbf{x}) - f(\mathbf{x}; \mathbf{w})\right)^2\right\} = \underset{\mathbf{w}}{\operatorname{argmin}} \left\{E\left\{f^2(\mathbf{x}; \mathbf{w})\right\} - 2E\left\{\frac{\partial f}{\partial x_i}(\mathbf{x}, \mathbf{w})\right\}\right\} \tag{13}$$

This corollary shows a nice property of JSF: *even without priors about $\varphi_i(\mathbf{x})$, we can design a minimum mean square error (MMSE) estimator for it.*

Property 4: For a random vector $\mathbf{x} = (x_1, \dots, x_N)^T$ we have:

$$\psi_i(x) = E\left\{\varphi_i(\mathbf{x}) \mid x_i = x\right\} \tag{14}$$

where φ_i and ψ_i denote the i-th component of JSF and MSF of $\mathbf{x}$, respectively.

Proof: Without loss of generality let $i = 1$. Then:

$$\begin{aligned} E\{\varphi_1(\mathbf{x}) \mid x_1\} &= \int_{x_2,\dots,x_N} \varphi_1(\mathbf{x})\, p(x_2, \dots, x_N \mid x_1)\, dx_2 \cdots dx_N \\ &= -\int_{x_2,\dots,x_N} \frac{\frac{\partial}{\partial x_1} p_{\mathbf{x}}(\mathbf{x})}{p(\mathbf{x})} \cdot \frac{p(\mathbf{x})}{p_{x_1}(x_1)}\, dx_2 \cdots dx_N \\ &= -\frac{1}{p_{x_1}(x_1)} \cdot \frac{\partial}{\partial x_1} \int_{x_2,\dots,x_N} p(\mathbf{x})\, dx_2 \cdots dx_N = -\frac{1}{p_{x_1}(x_1)} \cdot \frac{\partial}{\partial x_1} p_{x_1}(x_1) = \psi_1(x_1) \end{aligned} \tag{15}$$

which proves the property. ■

The above property needs more discussion. Consider φ_i as a function[2] of x_i, denoted by $\varphi_i(x_i)$. If x_i is independent from the other variables, then Property 1 implies $\varphi_i(x_i) = \psi_i(x_i)$. Now, if the other variables depend on x_i, $\varphi_i(x_i)$ is no longer equal to $\psi_i(x_i)$, however, the above property states that its "mean" will be still equal to $\psi_i(x_i)$. In other words, the statistical dependence, can introduce some fluctuations in $\varphi_i(x_i)$, but only around its constant "mean". We will later use this property for estimating SFD.

Example. Let s_1 and s_2 be two independent random variables with uniform distributions on the interval $[-0.5, 0.5]$. Consider now random variables $x_1 \triangleq s_1$ and $x_2 \triangleq s_2 + ks_1$. For $k = 0$, x_1 and x_2 are independent and hence $\varphi_1(x_1) = \psi_1(x_1)$. Now, if k varies, $\psi_1(x_1)$ remains unchanged (because it does not depend on k) but $\varphi_1(x_1)$ will change. However, Property 4 states that its "mean" remains unchanged. Figure 1 shows the estimated[3] $\varphi_1(x_1)$ and $\psi_1(x_1)$ for $k = 0.5$.

Therefore, it is concluded that: *The SFD is, in fact, a measure of the variations of JSF* (around its smoothed value).

Property 5: Let $\mathbf{y} = \mathbf{B}\mathbf{x}$, where $\mathbf{x}$ and $\mathbf{y}$ are random vectors and $\mathbf{B}$ is a non-singular square matrix. Then:

$$\boldsymbol{\varphi}_{\mathbf{y}}(\mathbf{y}) = \mathbf{B}^{-T} \boldsymbol{\varphi}_{\mathbf{x}}(\mathbf{x}) \tag{16}$$

[2] Strictly speaking, it is a "relation" not a "function", because for each value of x_i there are several values for φ_i.

[3] These curves have been obtained by using *kernel estimators* (see Section IV-A).

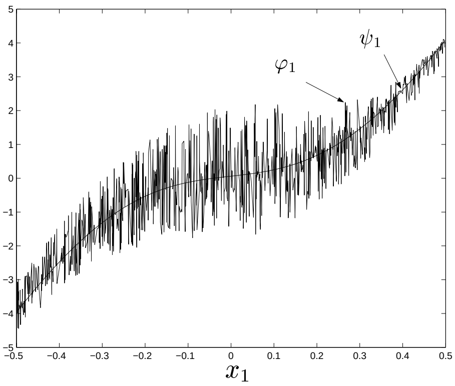

Fig. 1. φ_1 and ψ_1 versus x_1 in the presence of statistical dependence.

Proof: From $\mathbf{y} = \mathbf{B}\mathbf{x}$ we have:

$$p_{\mathbf{y}}(\mathbf{y}) = \frac{p_{\mathbf{x}}(\mathbf{x})}{|\det \mathbf{B}|} \Rightarrow \ln p_{\mathbf{x}}(\mathbf{x}) = \ln p_{\mathbf{y}}(\mathbf{y}) + \ln|\det \mathbf{B}| \tag{17}$$

Therefore, for $i = 1, \dots, N$:

$$\varphi_{\mathbf{x},i}(\mathbf{x}) = -\frac{\partial}{\partial x_i} \ln p_{\mathbf{x}}(\mathbf{x}) = -\frac{\partial}{\partial x_i} \ln p_{\mathbf{y}}(\mathbf{y}) = -\sum_k \frac{\partial}{\partial y_k} \ln p_{\mathbf{y}}(\mathbf{y}) \cdot \frac{\partial y_k}{\partial x_i} = \sum_k b_{ki}\, \varphi_{\mathbf{y},k}(\mathbf{y}) \tag{18}$$

where $\varphi_{\mathbf{x},i}(\mathbf{x})$ and $\varphi_{\mathbf{y},i}(\mathbf{y})$ denote the i-th components of the JSFs of $\mathbf{x}$ and $\mathbf{y}$, respectively. From the above relation, we have $\boldsymbol{\varphi}_{\mathbf{x}}(\mathbf{x}) = \mathbf{B}^T \boldsymbol{\varphi}_{\mathbf{y}}(\mathbf{y})$, which proves the property. ■

III. "Gradient" of Mutual Information

In this Section, we use the multivariate score functions defined previously, for deriving a "gradient" for mutual information. The variation of mutual information resulting from a small deviation in its argument (it may be called "differential" of mutual information) is given by the following theorem [9].

Theorem 1 (Differential of mutual information): Let $\mathbf{x}$ be a bounded random vector, and let $\boldsymbol{\Delta}$ be a 'small' random vector with the same dimension, then:

$$I(\mathbf{x} + \boldsymbol{\Delta}) - I(\mathbf{x}) = E\left\{\boldsymbol{\Delta}^T \boldsymbol{\beta}_{\mathbf{x}}(\mathbf{x})\right\} + o(\boldsymbol{\Delta}) \tag{19}$$

where $\boldsymbol{\beta}_{\mathbf{x}}$ is the SFD of $\mathbf{x}$, and $o(\boldsymbol{\Delta})$ denotes higher order terms in $\boldsymbol{\Delta}$.

Remark. Equation (19) may be stated in the following form (which is similar to what is done in [13]):

$$I(\mathbf{x} + \mathcal{E}\mathbf{y}) - I(\mathbf{x}) = E\left\{(\mathcal{E}\mathbf{y})^T \boldsymbol{\beta}_{\mathbf{x}}(\mathbf{x})\right\} + o(\mathcal{E}) \tag{20}$$

where $\mathbf{x}$ and $\mathbf{y}$ are bounded random vectors, $\mathcal{E}$ is a matrix with small entries, and $o(\mathcal{E})$ stands for a term that converges to zero faster than $||\mathcal{E}||$. This equation is mathematically more sophisticated, because it defines more formally the term 'small random vector' which is used in (19) without any formal mathematical definition. Conversely, (19) is simpler, and easier to use in developing gradient based algorithms for optimizing a mutual information.

Recall that for a multivariate (differentiable) function $f(\mathbf{x})$ we have:

$$f(\mathbf{x}+\boldsymbol{\Delta}) - f(\mathbf{x}) = \boldsymbol{\Delta}^T \cdot (\nabla f(\mathbf{x})) + o(\boldsymbol{\Delta}) \tag{21}$$

A comparison of this equation with (19) shows that *SFD can be called (or viewed as) the "stochastic gradient" of mutual information*.

Recall now the Property 1, which states that SFD is zero *if and only if* $I(\mathbf{x})$ is minimum (zero). For a usual multivariate function $f(\mathbf{x})$, $\nabla f(\mathbf{x}) = \mathbf{0}$ is only a *necessary* condition for $\mathbf{x}$ to be a global minimum of f (because it holds for every local minimum, too). However, if this condition becomes also a *sufficient* condition for a function f, it can be concluded that all minimums of f are global. Considering SFD as a "gradient" for mutual information, Property 1 states that vanishing the gradient of mutual information is a necessary and also a sufficient condition for I to be minimum (zero). This leads us to the conclusion that *mutual information has no "local minima"*. This fact is more formally stated in the following theorem:

Theorem 2: Let $\mathbf{x}_0$ be a random vector with a continuously differentiable PDF. If for any "small" random vector $\boldsymbol{\Delta}$, $I(\mathbf{x}_0) \le I(\mathbf{x}_0 + \boldsymbol{\Delta})$ holds, then $I(\mathbf{x}_0) = 0$.

Proof: Recall that for a usual multivariate function f, if $\nabla f(\mathbf{x}) \neq \mathbf{0}$, then a small variation in the opposite direction of the gradient results in a reduction in the value of the function. In other words, $f(\mathbf{x}+\boldsymbol{\Delta}) < f(\mathbf{x})$, where $\boldsymbol{\Delta} = -\mu\nabla f(\mathbf{x})$ and μ is a small positive constant. This result can be seen by letting $\boldsymbol{\Delta} = -\mu\nabla f(\mathbf{x})$ in (21), and using the fact that $\|\nabla f(\mathbf{x})\|^2 > 0$. Now, we apply a similar reasoning for I, and combining it with Property 1 proves the theorem.

Assuming $\forall\boldsymbol{\Delta}$, $I(\mathbf{x}_0) \le I(\mathbf{x}_0 + \boldsymbol{\Delta})$ holds, if $\boldsymbol{\beta}_{\mathbf{x}_0}(\mathbf{x}) \neq \mathbf{0}$ (which is, by Property 1, equivalent to $I(\mathbf{x}_0) \neq 0$), then a small variation in its opposite direction must reduce mutual information. In other words, if we let $\boldsymbol{\Delta} = -\mu\boldsymbol{\beta}_{\mathbf{x}_0}(\mathbf{x}_0)$, then using Theorem 1 we have (up to first order terms):

$$I(\mathbf{x}_0 + \boldsymbol{\Delta}) - I(\mathbf{x}_0) = -\mu E\left\{\left\|\boldsymbol{\beta}_{\mathbf{x}_0}(\mathbf{x}_0)\right\|^2\right\} < 0 \tag{22}$$

(see also Remark 1 below). Hence $I(\mathbf{x}_0 + \boldsymbol{\Delta}) < I(\mathbf{x}_0)$, which is a contradiction. Consequently, $\boldsymbol{\beta}_{\mathbf{x}_0}(\mathbf{x})$ must be zero. By Property 1, this is equivalent to $I(\mathbf{x}_0) = 0$, that is, I is in its global minimum. ■

Remark 1: In the above proof, for writing (22), we need to conclude $E\left\{\left\|\boldsymbol{\beta}_{\mathbf{x}_0}(\mathbf{x}_0)\right\|^2\right\} \neq 0$ by knowing that $\boldsymbol{\beta}_{\mathbf{x}_0}(\mathbf{x})$ is not identically zero. This certainly holds if the PDF of $\mathbf{x}_0$ is continuously differentiable, as assumed in the theorem. However, it also holds for every "usual" random vector, and the required restriction is highly less than having a continuously differentiable PDF for $\mathbf{x}_0$.

Remark 2: Although the above theorem guarantees that the mutual information has no local minimum, it expresses nothing about the local minimum of a parametric separating system with respect to the parameter space. In other words, for the nonlinear separating system $\mathbf{y} = g(\mathbf{x}; \mathbf{w})$, the function $h(\mathbf{w}) = I(\mathbf{y})$ may have some local minima with respect to the parameter vector $\mathbf{w}$. For example, if we consider the following mixing system (corresponding to a rotation with angle θ):

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} \tag{23}$$

then it has been shown [14], [15] that $h(\theta) = I(y_1, y_2)$ may contain local minima with respect to θ, specially when s_1 and s_2 have multimodal PDF's. This is not in contradiction with the above theorem, because it is a minimization of mutual information under constraints. For clarity, think about a two-variate function $z = f(x, y)$. Even if f has no local minima with respect to (x, y), the function $h(t) = f(x(t), y(t))$ may have local minima with respect to t. In fact, f represents a 'surface' in the 3-dimensional space which has no local minima, but h represents a 'curve' on this surface, which may have local minima.

The same problem exists, for example, in MMSE estimation of a nonlinear model: although the function $(\cdot)^2$ has no local minima, the function $h(\mathbf{w}) = E\left\{(\mathbf{y} - g(\mathbf{x}; \mathbf{w}))^2\right\}$ may contain local minima (as a function of $\mathbf{w}$). The local minimum, can also be introduced by the estimation method of SFD.

Finally, the theorem does not imply the uniqueness of the minimum ; mutual information may have several global minima, all of them correspond to $I = 0$.

IV. Estimating Multi-Variate score functions

For using SFD in minimizing mutual information, it must be first estimated from the observed data. The MSF of a random vector is nothing but the score functions of its components, and hence it can be estimated by any estimation method of a score function (see for example [12], [11], [16]). In this section, the estimation of JSF and SFD is considered.

A. Estimating JSF

The methods presented here for estimating JSF are, in fact, the generalizations of the corresponding methods for estimating uni-variate score functions.

1) Kernel Estimators: A multi-variate *kernel* function $k(\mathbf{x}) = k(x_1, \ldots, x_N)$ is defined as the PDF of a zero mean random vector. That is, $k(\mathbf{x})$ is a kernel function if and only if: (a) $\forall \mathbf{x} \in \mathbb{R}^N, k(\mathbf{x}) \geq 0$, (b) $\int_{\mathbb{R}^N} k(\mathbf{x})\, d\mathbf{x} = 1$ and (c) $\int_{\mathbb{R}^N} \mathbf{x}\, k(\mathbf{x})\, d\mathbf{x} = \mathbf{0}$. The *bandwidth* of the kernel in direction i is defined as the variance of the i-th component of the corresponding random vector: $h_i \triangleq \int_{\mathbb{R}^N} x_i^2\, k(\mathbf{x})\, d\mathbf{x}$.

Let now $\mathbf{x}$ be a random vector, from which the samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ have been observed. Then the kernel estimation of the PDF of $\mathbf{x}$ is [17], [18]:

$$\hat{p}_{\mathbf{x}}(\mathbf{x}) \triangleq \frac{1}{T} \sum_{t=1}^{T} k(\mathbf{x} - \mathbf{x}_t) \tag{24}$$

And from there, the kernel estimation of the i-th component of JSF will be:

$$\hat{\varphi}_i(\mathbf{x}) \triangleq -\frac{\frac{\partial}{\partial x_i}\hat{p}_{\mathbf{x}}(\mathbf{x})}{\hat{p}_{\mathbf{x}}(\mathbf{x})} = -\frac{\sum_{t=1}^{T} \frac{\partial k}{\partial x_i}(\mathbf{x} - \mathbf{x}_t)}{\sum_{t=1}^{T} k(\mathbf{x} - \mathbf{x}_t)} \tag{25}$$

Remark 1. The bandwidth parameter in the above equations determines the degree of smoothness of the estimator: the larger bandwidth, the smoother estimated PDF. If it is chosen too small, the estimated PDF will be too fluctuating, and if it is chosen too large, the estimated PDF will be a rough shape of the kernel itself. Optimal choice can be done by cross-validation (see [18] for instance). However, a rule of thumb is given in the Remark 3 below.

Remark 2. The bandwidth of the kernel can be different in each direction, which depends on the spread of data in that direction. However, as suggested by Fukunaga [19], after a whitening process on data, we can use isotropic kernels (kernels with equal bandwidths in all directions). To state this idea more clearly, let $\mathbf{R}_{\mathbf{x}}$ denote the covariance matrix of $\mathbf{x}$, and $\mathbf{T}$ be its Cholesky decomposition, *i.e.* $\mathbf{T}$ is an upper triangular matrix which satisfies $\mathbf{R}_{\mathbf{x}} = \mathbf{T}\mathbf{T}^T$. If we now define the random vector $\mathbf{y} = \mathbf{T}^{-1}\mathbf{x}$, then $\mathbf{R}_{\mathbf{y}} = \mathbf{I}$, that is, the variance of $\mathbf{y}$ is the same in any directions, and moreover the variables y_i are uncorrelated. Hence, it is natural to use isotropic kernels for estimating PDF and JSF of $\mathbf{y}$, and then we can use the equations (see Property 5):

$$\hat{p}_{\mathbf{x}}(\mathbf{x}) = \frac{\hat{p}_{\mathbf{y}}(\mathbf{y})}{|\det \mathbf{T}|}, \quad \hat{\boldsymbol{\varphi}}_{\mathbf{x}}(\mathbf{x}) = \mathbf{T}^{-T}\hat{\boldsymbol{\varphi}}_{\mathbf{y}}(\mathbf{y}) \tag{26}$$

for estimating PDF and JSF of $\mathbf{x}$.

Remark 3. When the above prewhitening is done, as a rule of thumb the value $h_{\,opt} = c\,N^{-1/(d+4)}$ can be used for the bandwidth [18], where d is the dimension of the random vector, and c is a constant which depends on the type of the kernel. For d-dimensional Gaussian kernels:

$$c = \left(\frac{4}{2d+1}\right)^{\frac{1}{d+4}} \tag{27}$$

For example, for scalar Gaussian kernels $c \approx 1.06$, and for 2-dimensional Gaussian kernels $c \approx 0.96$.

2) Minimum Mean Square Error (MMSE) estimation: Property 3 can be used in designing Minimum Mean Square Error (MMSE) estimators for $\varphi_i(\mathbf{x})$ (see also Corollary 2 of the property). Here we consider a parametric model which is linear with respect to the parameters.

Let $\varphi_i(\mathbf{x})$ be estimated as a linear combination of multi-variate functions $\{k_1(\mathbf{x}), \dots, k_L(\mathbf{x})\}$, that is, $\hat{\varphi}_i(\mathbf{x}) = \sum_{j=1}^{L} w_j k_j(\mathbf{x}) = \mathbf{k}^T(\mathbf{x})\,\mathbf{w}$, where $\mathbf{k}(\mathbf{x}) \triangleq (k_1(\mathbf{x}), \dots, k_L(\mathbf{x}))^T$, and $\mathbf{w} \triangleq (w_1, \dots, w_L)^T$. $\mathbf{w}$ must be computed by minimizing the error:

$$\mathcal{E} = E\left\{\big(\varphi_i(\mathbf{x}) - \hat{\varphi}_i(\mathbf{x})\big)^2\right\} \tag{28}$$

Orthogonality principle [20] implies that $E\left\{\mathbf{k}(\mathbf{x})\big(\varphi_i(\mathbf{x}) - \hat{\varphi}_i(\mathbf{x})\big)\right\} = 0$, and by using Property 3 we obtain the following equation which determines $\mathbf{w}$:

$$E\left\{\mathbf{k}(\mathbf{x})\mathbf{k}^T(\mathbf{x})\right\}\mathbf{w} = E\left\{\frac{\partial \mathbf{k}}{\partial x_i}(\mathbf{x})\right\} \tag{29}$$

B. Estimating SFD

Since SFD is the gradient of mutual information, we are mainly interested in estimating SFD, and not only JSF. Therefore, in this section, we consider some methods for SFD estimation.

1) Independent estimations of JSF and MSF: One method for estimating SFD, is to estimate independently JSF and MSF, and then to compute their difference. That is, $\hat{\boldsymbol{\beta}}_{\mathbf{x}}(\mathbf{x}) = \hat{\boldsymbol{\psi}}_{\mathbf{x}}(\mathbf{x}) - \hat{\boldsymbol{\varphi}}_{\mathbf{x}}(\mathbf{x})$. We may apply kernel or MMSE estimation of the joint and marginal score functions.

Example. (Polynomial estimation of SFD) The following estimator is applied successfully in [10] for separating convolutive mixtures of two sources. $\varphi_i(x_1, x_2)$ is estimated by the polynomial $\hat{\varphi}_i(x_1, x_2) = \sum_{j=1}^{7} w_{ij}\, k_j(x_1, x_2)$, where:

$$\begin{aligned} k_1(x_1,x_2) = 1, \quad k_2(x_1,x_2) = x_1, \quad k_3(x_1,x_2) = x_1^2, \quad k_4(x_1,x_2) = x_1^3 \\ k_5(x_1,x_2) = x_2, \quad k_6(x_1,x_2) = x_2^2, \quad k_7(x_1,x_2) = x_2^3 \end{aligned} \tag{30}$$

Consequently:

$$\mathbf{k}(x_1,x_2) = \begin{pmatrix} 1 & x_1 & x_1^2 & x_1^3 & x_2 & x_2^2 & x_2^3 \end{pmatrix}^T \tag{31}$$

$$\frac{\partial}{\partial x_1}\mathbf{k}(x_1,x_2) = \begin{pmatrix} 0 & 1 & 2x_1 & 3x_1^2 & 0 & 0 & 0 \end{pmatrix}^T \tag{32}$$

$$\frac{\partial}{\partial x_2}\mathbf{k}(x_1,x_2) = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 2x_2 & 3x_2^2 \end{pmatrix}^T \tag{33}$$

Then from (29) we have:

$$E\left\{\mathbf{k}(x_1,x_2)\mathbf{k}(x_1,x_2)^T\right\}\mathbf{w}_1 = E\left\{\frac{\partial}{\partial x_1}\mathbf{k}(x_1,x_2)\right\} \tag{34}$$

$$E\left\{\mathbf{k}(x_1,x_2)\mathbf{k}(x_1,x_2)^T\right\}\mathbf{w}_2 = E\left\{\frac{\partial}{\partial x_2}\mathbf{k}(x_1,x_2)\right\} \tag{35}$$

where $\mathbf{w}_1 \triangleq (w_{11}, w_{12}, \ldots, w_{17})^T$ and $\mathbf{w}_2 \triangleq (w_{21}, w_{22}, \ldots, w_{27})^T$. These equations determine $\mathbf{w}_1$ and $\mathbf{w}_2$, which determine $\hat{\varphi}_1(x_1,x_2)$ and $\hat{\varphi}_2(x_1,x_2)$.

Similarly, $\psi_i(x_i)$ is estimated by $\hat{\psi}_i(x_i) = w_1^{(i)} + w_2^{(i)}x_i + w_3^{(i)}x_i^2 + w_4^{(i)}x_i^3$, and the optimal values of the coefficients are obtained from:

$$E\left\{\begin{pmatrix} 1 & x_i & x_i^2 & x_i^3 \end{pmatrix}^T \begin{pmatrix} 1 & x_i & x_i^2 & x_i^3 \end{pmatrix}\right\}\mathbf{w}^{(i)} = E\left\{\begin{pmatrix} 0 & 1 & 2x_i & 3x_i^2 \end{pmatrix}^T\right\} \tag{36}$$

where $\mathbf{w}^{(i)} \triangleq \left(w_1^{(i)}, \ldots, w_4^{(i)}\right)^T$ for $i = 1, 2$.

Finally, the SFD is estimated by using the above estimations, that is, $\hat{\beta}_1(x_1,x_2) = \hat{\psi}_1(x_1) - \hat{\varphi}_1(x_1,x_2)$ and $\hat{\beta}_2(x_1,x_2) = \hat{\psi}_2(x_2) - \hat{\varphi}_2(x_1,x_2)$.

2) Smoothing JSF: The problem of the previous method is that the estimation errors of JSF and MSF are independent. Recall that a gradient based algorithm for minimizing the mutual information stops when SFD (the difference between MSF and JSF) vanishes. However, small estimation errors $E\left\{(\boldsymbol{\varphi}(\mathbf{x}) - \hat{\boldsymbol{\varphi}}(\mathbf{x}))^2\right\}$ and $E\left\{(\boldsymbol{\psi}(\mathbf{x}) - \hat{\boldsymbol{\psi}}(\mathbf{x}))^2\right\}$ does not necessarily results in a small estimation error $E\{(\boldsymbol{\beta}(\mathbf{x}) - \hat{\boldsymbol{\beta}}(\mathbf{x}))^2\}$. In other words, when the MSF and JSF estimation errors are independent, their difference does not necessarily vanish for independent variables. In linear mixtures, the limited degree of freedom of the separating system overcomes this problem and the above estimator works well. On the contrary, nonlinear mixtures require more accurate estimation of the SFD.

Then we suggest to estimate MSF from the estimated JSF. From the Property 4 we know that MSF is the smoothed version of JSF. Therefore, we can estimate MSF as the smoothed version of the estimated JSF. With this trick, the estimation errors in JSF and MSF are no longer independent, and they partially cancel each other when calculating their difference. In other words, in this method, SFD is estimated as the variations of JSF around its mean (see Fig. 1), and hence the separation algorithm tries to minimize these variations.

Practically, following Property 4, $\psi_i(x_i)$ is a regression from x_i to $\varphi_i(\mathbf{x})$, which can be calculated for instance using smoothing splines [21]. The final estimation procedure is summarized in the following steps:

1) From the observed values $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ estimate $\hat{\varphi}_{i,t} = \hat{\varphi}_i(\mathbf{x}_t), t = 1, \ldots, T$ (*e.g.* by kernel estimators).
2) Compute the smoothing spline (or another regressor) which fits on the data $(x_{\,i,t}, \hat{\varphi}_{i,t}), t = 1, \ldots, T$. This spline will be the estimated MSF $\hat{\psi}_i(x_i)$.
3) Estimate SFD by $\hat{\boldsymbol{\beta}}_{\mathbf{x}}(\mathbf{x}) = \hat{\boldsymbol{\psi}}_{\mathbf{x}}(\mathbf{x}) - \hat{\boldsymbol{\varphi}}_{\mathbf{x}}(\mathbf{x})$.

3) Histogram estimation method: Another method for estimating SFD (in two dimensional case) is based on a simple histogram estimation of the PDF of $\mathbf{x}$. Histogram is not a very accurate estimator for PDF, but since we estimate SFD directly, we do not need a very good estimation of PDF. In fact, despite of its simplicity, this estimator works very well for separating instantaneous linear mixtures.

In this method, a histogram is first used for estimating the joint PDF of $\mathbf{x}$. For two-dimensional vectors, let $N(n_1, n_2)$ denote the number of observations in the bin (n_1, n_2), then the histogram estimation of $p_{\mathbf{x}}$ is:

$$p(n_1, n_2) = \frac{N(n_1, n_2)}{T} \tag{37}$$

where T is the number of observations. From there, a histogram estimation of p_{x_1} is obtained by $p_1(n_1) = \sum_{n_2} p(n_1, n_2)$, and then we will have an estimation of $p(x_2|x_1)$:

$$p(n_2|n_1) = \frac{p(n_1, n_2)}{p_1(n_1)} = \frac{N(n_1, n_2)}{N(n_1)} \tag{38}$$

Finally, from (8) we have $\beta_1(x_1, x_2) = (\frac{\partial}{\partial x_1} p(x_2|x_1))/(p(x_2|x_1))$, consequently a histogram estimation of $\beta_1(x_1, x_2)$ is obtained by:

$$\beta_1(n_1, n_2) = \frac{p(n_2|n_1) - p(n_2|n_1 - 1)}{p(n_2|n_1)} \tag{39}$$

$\beta_2(n_1, n_2)$ will be estimated in a similar manner. Note that the value $\boldsymbol{\beta}(n_1, n_2)$ will be assigned to all the points of the bin (n_1, n_2).

Note. $p(n_2|n_1)$ is not defined in the bins where $p_1(n_1) = 0$, but this is not important, because we don't need the value of SFD in these bins, since there is no point in these bins. However, in the bins with smallest n_1, computing $\beta_1(n_1, n_2)$ from (39) requires the value of $p(n_2|n_1 - 1)$ which is not defined. In our simulations, we have used 0 for these values, too.

4) Pham's method: Recently D. T. Pham has proposed [22] a method for estimating the "conditional score function", which appears in separating temporary correlated sources. The conditional score function of the random vector $\mathbf{x} = (x_1, \ldots, x_N)^T$ is defined by:

$$\psi_{x_N|x_{N-1}\cdots x_1}(x_N|x_{N-1}, \ldots, x_1) \triangleq -\nabla \ln p_{x_N|x_{N-1}\cdots x_1}(x_N|x_{N-1}, \ldots, x_1) \tag{40}$$

where $p_{x_N|x_{N-1}\cdots x_1}(x_N|x_{N-1}, \ldots, x_1)$ is the conditional density of x_N given $x_1, \ldots, x_{N-1}$. From the Property 2, it is seen that this is closely related to the SFD of $\mathbf{x}$, and its estimation can be used to estimate SFD. The Pham's method uses cubic splines for this estimation, which results in a fast algorithm.

V. Two general approaches for Mutual information minimization

In this section, we propose two gradient-based approaches for minimizing $I(\mathbf{y})$ with the model of equation (2). These approaches are both based on Theorem 1 and SFD as a non-parametric gradient for mutual information.

A. Gradient approach

In the first approach, Theorem 1 is used to calculate $\partial I(\mathbf{y})/\partial \boldsymbol{\theta}$, and then applying the steepest descent algorithm on the parameter vector:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \mu \frac{\partial I(\mathbf{y})}{\partial \boldsymbol{\theta}} \tag{41}$$

For calculating $\partial I(\mathbf{y})/\partial \boldsymbol{\theta}$ using Theorem 1, one can let $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta} + \boldsymbol{\delta}$ where $\boldsymbol{\delta}$ is a 'small' deviation in the parameters. Then using Theorem 1 the effect of this deviation on the output can be calculated, which leads to the calculation of $\partial I(\mathbf{y})/\partial \boldsymbol{\theta}$.

Example 1. For linear instantaneous mixtures, $\mathbf{y}=\mathbf{B}\mathbf{x}$. Here the separation parameter is the matrix $\mathbf{B}$. Let $\hat{\mathbf{B}}=\mathbf{B}+\mathcal{E}$, where $\mathcal{E}$ is a 'small' matrix. Then the new output is $\hat{\mathbf{y}}=\hat{\mathbf{B}}\mathbf{x}=\mathbf{B}\mathbf{x}+\mathcal{E}\mathbf{x}=\mathbf{y}+\mathcal{E}\mathbf{x}$. Consequently, from Theorem 1:

$$\hat{I}-I=E\left\{\boldsymbol{\beta}_{\mathbf{y}}^T(\mathbf{y})\mathcal{E}\mathbf{x}\right\}=\left\langle\mathcal{E},E\left\{\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})\mathbf{x}^T\right\}\right\rangle \tag{42}$$

Where $\langle .,. \rangle$ stands for the scalar (Euclidean) product of matrices[4]. Finally (42) shows that:

$$\frac{\partial I(\mathbf{y})}{\partial \mathbf{B}}=E\left\{\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})\mathbf{x}^T\right\} \tag{43}$$

and the separation algorithm will be $\mathbf{B}\leftarrow\mathbf{B}-\mu E\left\{\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})\mathbf{x}^T\right\}$, where μ is a small positive constant.

Example 2. In convolutive mixtures, the separating system is:

$$\mathbf{y}(n)=\mathbf{B}_0\mathbf{x}(n)+\mathbf{B}_1\mathbf{x}(n-1)+\cdots+\mathbf{B}_p\mathbf{x}(n-p) \tag{44}$$

To calculate $\partial I(\mathbf{y}(n))/\partial \mathbf{B}_k$, let $\hat{\mathbf{B}}_k=\mathbf{B}_k+\mathcal{E}$, where $\mathcal{E}$ denotes a 'small' matrix. This implies that $\hat{\mathbf{y}}(n)=\mathbf{y}(n)+\mathcal{E}\mathbf{x}(n-k)$, and then by a reasoning similar to the above example, we conclude:

$$\frac{\partial I(\mathbf{y}(n))}{\partial \mathbf{B}_k}=E\left\{\boldsymbol{\beta}_{\mathbf{y}(n)}(\mathbf{y}(n))\mathbf{x}^T(n-k)\right\} \tag{45}$$

However, the above gradient cannot be directly used for separating convolutive mixtures, since some other points must be taken into account (see Section VIII).

B. Minimization-Projection (MP) Approach

Since SFD can be seen as the 'gradient' of mutual information, one may think about the following "steepest descent"-like algorithm for minimizing $I(\mathbf{y})$ in a parametric model $\mathbf{y}=\mathcal{G}(\mathbf{x},\boldsymbol{\theta})$:

$$\mathbf{y}\leftarrow\mathbf{y}-\mu\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y}) \tag{46}$$

where μ is a small positive constant. This algorithm has no local minimum and converges to a random vector $\mathbf{y}$ with statistically independent components (provided that μ is small enough). To show this fact, let $\mathbf{y}_n$ denote the value of $\mathbf{y}$ at the n-th iteration. Then $\mathbf{y}_{n+1}=\mathbf{y}_n-\mu\boldsymbol{\beta}_{\mathbf{y}_n}(\mathbf{y}_n)$, and from Theorem 1:

$$I(\mathbf{y}_{n+1})-I(\mathbf{y}_n)=-\mu E\left\{\left\|\boldsymbol{\beta}_{\mathbf{y}_n}(\mathbf{y}_n)\right\|^2\right\}\leq 0\Rightarrow I(\mathbf{y}_{n+1})\leq I(\mathbf{y}_n) \tag{47}$$

Moreover the equality holds if and only if $\boldsymbol{\beta}_{\mathbf{y}_n}(\mathbf{y}_n)=0$, that is, if the components of $\mathbf{y}_n$ are independent (Property 1). Consequently, the algorithm (46) converges to an independent vector without trapping in any local minimum.

However, after the convergence of the algorithm (46), there may be no particular relation between $\mathbf{y}$ and $\mathbf{x}$. On one hand, the transformation $\mathbf{x}\mapsto\mathbf{y}$ may be non-invertible, and on the other hand, it does not necessarily belong to the desired family $\mathbf{y}=\mathcal{G}(\mathbf{x},\boldsymbol{\theta})$, and hence the independence of $\mathbf{y}$ does not imply the source separation (recall that without structural constraints, output independence does not insure separation). Then, the idea of the Minimization-Projection (MP) approach [23] for overcoming this problem is to replace, at each iteration, the transformation $\mathbf{x}\mapsto\mathbf{y}$ by its "projection" on the desired family. In other words, each iteration of the separating algorithm is composed of the following steps:

[4] The scalar (Euclidean) product of two $p\times q$ matrices $\mathbf{M}$ and $\mathbf{N}$ is defined by $\langle\mathbf{M},\mathbf{N}\rangle\triangleq\sum_{i,j}m_{ij}n_{ij}$. Moreover, it can be easily seen that for vectors $\mathbf{x}$ and $\mathbf{y}$ and matrix $\mathbf{A}$ we have $\mathbf{x}^T\mathbf{A}\mathbf{y}=\langle\mathbf{A},\mathbf{x}\mathbf{y}^T\rangle$.

- Minimization:
  1. $\mathbf{y} \leftarrow \mathbf{y} - \mu \boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})$.
- Projection:
  2. $\boldsymbol{\theta}_0 = \mathrm{argmin}_{\boldsymbol{\theta}} E\{\|\mathbf{y} - \mathcal{G}(\mathbf{x}; \boldsymbol{\theta})\|^2\}$.
  3. $\mathbf{y} = \mathcal{G}(\mathbf{x}, \boldsymbol{\theta}_0)$.

Note that the dependence of the algorithm to the separating system appears only in the projection step: the minimization step is the same for all separating models.

Remark. The minimization part of the MP approach has no local minimum, and the outputs are directly manipulated regardless of the structure of the separating system. Consequently, it can be conjectured that it is less probable that this approach traps in a local minimum compared to the gradient approach, in which a steepest descent algorithm is applied on each parameter of the separating system. In other words, the MP approach is expected to have better convergence behavior than the gradient approach.

VI. Some separable models for Blind Source Separation

In the following sections, we will use the presented gradient and MP approaches for Blind Source Separation. In BSS, a parametric model for the separating system is required, because general non-linear mixtures are not separable [2] (that is, the output independence does not insure source separation). Four separable models, for which, separation algorithms will be addressed in the following sections are:

1) Linear Instantaneous Mixtures: This is the simplest case, in which the mixing system is $\mathbf{x} = \mathbf{A}\mathbf{s}$ and the separating system is $\mathbf{y} = \mathbf{B}\mathbf{x}$, where $\mathbf{A}$ and $\mathbf{B}$ are constant (and regular) matrices. For these mixtures, it is shown [3] that the independence of the outputs insures source separation, up to a scale and a permutation indeterminacy (provided that there is at most one Gaussian source).

2) Convolutive mixtures: In convolutive mixtures, the mixing matrix composed of finite order Linear Time-Invariant (LTI) filters instead of scalars. For these mixtures, the separation system is:

$$\mathbf{y}(n) = [\mathbf{B}(z)]\,\mathbf{x}(n) = \sum_{k=0}^{p} \mathbf{B}_k \mathbf{x}(n-k) \tag{48}$$

As it has been proved in [24], convolutive mixtures are separable, too: if $\mathbf{B}(z)$ is determined to produce statistically independent outputs, then the sources are separated.

In convolutive mixtures, the scale indeterminacy of instantaneous mixtures extends to a *filtering indeterminacy*. However, it can be shown [25] that the effect of each source on each sensor (that is, what a sensor would receive if all other sources were zero), can be found after the source separation.

In convolutive mixtures, the independence of outputs cannot be reduced to instantaneous independence of $y_1(n)$ and $y_2(n)$ [10]. In effect, in convolutive mixtures, y_1 and y_2 must be treated as stochastic processes, not random variables. Recall that two stochastic processes y_1 and y_2 are independent if the set of random variables $\{y_1(n_1), \dots, y_1(n_k)\}$ is independent from the set $\{y_2(n'_1), \dots, y_2(n'_k)\}$, for every choice of k and $n_1, \dots, n_k, n'_1, \dots, n'_k$ [20]. This independence cannot be deducted from the instantaneous independence of $y_1(n)$ and $y_2(n)$.

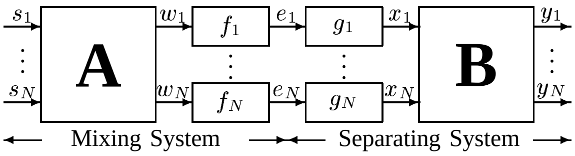

Fig. 2. The mixing-separating system for PNL mixtures.

However, for separating convolutive mixtures, it is sufficient that $y_1(n)$ and $y_2(n-m)$ be independent for all n's and all m's. This comes from the fact that this independence results in cancelling output bi-spectra, which is shown to be sufficient for separating convolutive mixtures [24]. Consequently, as proposed in [26] the separation criterion may be chosen:

$$J = \sum_{m=-M}^{+M} I\big(y_1(n), y_2(n-m)\big) \tag{49}$$

where $M = 2p+1$, and p is the degree of the separating filter ($\mathbf{B}(z)$). This criterion is computationally expensive. To reduce its computational cost, we use $I(y_1(n), y_2(n-m))$ as the separation criterion, but at each iteration a different random m is chosen form $\{-M, \dots, M\}$.

3) Post Non-Linear (PNL) mixtures: Post Non-Linear (PNL) mixtures have been first considered by Taleb and Jutten [16], as a (practically important) special case of non-linear mixtures which is separable. The PNL mixing-separating system is shown in Fig. 2. This model corresponds to a linear mixing system followed by non-linear sensors. For this model, it is shown that the independence of outputs insures that $g_i = f_i^{-1}$ and $\mathbf{B}$ is a separating matrix [16] up to scale and permutation indeterminacies like in linear mixtures (see [27] for a simple geometric proof for bounded sources). In other words, PNL mixtures are separable: independence is sufficient for estimating the separating structure and restoring the sources.

4) Convolutive PNL (CPNL) mixtures: This model is a generalization of PNL models, in which the mixing and separating matrices are convolutive, *i.e.* $\mathbf{A}$ and $\mathbf{B}$ (with scalar entries) are replaced by filter matrices $\mathbf{A}(z)$ and $\mathbf{B}(z)$ (where entries are filters) in Fig. 2. The separability of this model can be deduced from the separability of PNL mixtures [28], for finite impulse response filters.

VII. Application to linear instantaneous mixtures

Here, as an illustration of the "gradient" and "MP" approaches, we utilize them for separating linear instantaneous mixtures. All of the algorithms and programs of this section are available as a MATLAB package with a graphical interface at:

`http://www.lis.inpg.fr/pages_perso/bliss/deliverables/d20.html`

A. Gradient approach

For these mixtures, the gradient of $I(\mathbf{y})$ with respect to $\mathbf{B}$ has been obtained in equation (43). However, in linear instantaneous mixtures, it is preferable to use the equivariant algorithm [29], [30], which results in a separation quality independent of the mixing matrix. In equivariant algorithm, instead of $\frac{\partial I}{\partial \mathbf{B}}$, the natural [29] (or relative [30]) gradient is used:

$$\nabla_{\mathbf{B}} I \triangleq \frac{\partial I}{\partial \mathbf{B}} \mathbf{B}^T = E\left\{\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})\, \mathbf{y}^T\right\} \tag{50}$$

```
• Initialization: $\mathbf{B} = \mathbf{I}$ and $\mathbf{y} = \mathbf{x}$.
• Loop:
    1) Estimate $\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})$ (e.g. by histogram method).
    2) $\nabla_{\mathbf{B}} I = E\left\{\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})\,\mathbf{y}^T\right\}$
    3) $\mathbf{B} \leftarrow (\mathbf{I} - \mu \nabla_{\mathbf{B}} I)\,\mathbf{B}$.
    4) $\mathbf{y} = \mathbf{B}\mathbf{x}$.
    5) Normalization:
        – Let $y_i = y_i/\sigma_i$, where $\sigma_i^2$ is the energy of $y_i$.
        – Divide the $i$-th raw of $\mathbf{B}$ by $\sigma_i$.
• Repeat until convergence.
```

Fig. 3. Gradient algorithm for separating linear instantaneous mixtures.

and the updating algorithm for $\mathbf{B}$ is:

$$\mathbf{B} \leftarrow (\mathbf{I} - \mu \nabla_{\mathbf{B}} I)\,\mathbf{B} \tag{51}$$

where $\mathbf{I}$ denotes the identity matrix.

The scale indeterminacy must also be taken into account. The algorithm (51), which is based on output independence, imposes no restriction on the output energies. Moreover, $\mathbf{B} = \mathbf{0}$ is a trivial solution of the algorithm, which must be prevented. To solve this problem, the output energies may be normalized at each step of the algorithm. This results in the algorithm of Fig. 3 for separating linear instantaneous mixtures.

Experimental results. To examine the performance of the algorithm, we use two uniform random sources with zero means and unit variances. The mixing matrix is:

$$\mathbf{A} = \begin{pmatrix} 1 & 0.7 \\ 0.5 & 1 \end{pmatrix} \tag{52}$$

and the parameters of the separating algorithm are: (a) Histogram estimation of SFD (using a 10×10 histogram), (b) 500 point data block, and (c) $\mu = 0.1$. The experiment has been repeated 100 times (for 100 different realizations of the sources).

In this paper, for measuring the quality of separation, we use two performance indices. The first, is the mean of output Signal to Noise Ratios (SNR). At each output, the SNR is defined by (assuming there is no permutation):

$$\mathrm{SNR}_i = 10 \log_{10} \frac{E\left\{s_i^2\right\}}{E\left\{(s_i - y_i)^2\right\}} \tag{53}$$

Then, as a performance index, we use the mean of output SNR's, that is, $\mathrm{SNR} \triangleq (\mathrm{SNR}_1 + \mathrm{SNR}_2)/2$. This is done for having just a single number as the performance index. Moreover, it must noted that we use averaged SNR's over several (say 100) runs of the algorithm, for which SNR_1 and SNR_2 are essentially equal.

The second performance index is the number of iterations required for the SNR to reach at 90% of its final value.

In this experiment, averaged output SNR, taken over 100 runs of the algorithm was 32.4dB, and the number of iterations was 31 (the results of the experiments of the paper are collected in the Table I at the end of paper).

As it can be seen in this experiment, despite of the simplicity of the SFD estimator (a 10×10 histogram), a good separation performance has been obtained. This fact can be explained as follows. First note that (50) can be rewritten as (see the corollary 1 of the Property 3):

$$\nabla_{\mathbf{B}} I = E\left\{\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})\,\mathbf{y}^T\right\} = E\left\{\boldsymbol{\psi}_{\mathbf{y}}(\mathbf{y})\,\mathbf{y}^T\right\} - E\left\{\boldsymbol{\varphi}_{\mathbf{y}}(\mathbf{y})\,\mathbf{y}^T\right\} = E\left\{\boldsymbol{\psi}_{\mathbf{y}}(\mathbf{y})\,\mathbf{y}^T\right\} - \mathbf{I} \tag{54}$$

```
• Initialization: $\mathbf{y} = \mathbf{x}$.
• Loop:
    1) $\mathbf{y} \leftarrow \mathbf{y} - \mu \boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})$.
    2) Remove the DC of each output, and normalize its energy to 1.
    3) $\mathbf{B} = E\left\{\mathbf{y}\mathbf{x}^T\right\}\left(E\left\{\mathbf{x}\mathbf{x}^T\right\}\right)^{-1}$
    4) $\mathbf{y} = \mathbf{B}\mathbf{x}$.
• Repeat until convergence.
```

Fig. 4. Minimization-Projection algorithm for separating linear instantaneous mixtures.

which is the equation used in the previous methods of separating linear instantaneous mixtures by mutual information minimization [11]. Practically, this equation is simpler than (50), because it only requires estimation of marginal score functions. However, the separation information (independence) is contained in the averaged SFD, as the gradient of the mutual information, and the separating algorithms will stop when it vanishes. Moreover, SFD is the difference of two terms. In (54), one of these terms is theoretically computed. Hence, a good estimation of the other term is required for source separation (because the difference of the two terms must vanish for the convergence of the algorithm). But in our method which is based on the direct use of the SFD, since it is directly estimated, a good estimation of $\mathbf{B}$ can be achieved even with a coarse estimation of the SFD (*e.g.* a simple histogram).

Another advantage of this method (based on SFD) is that it can be readily generalized to more complicated mixtures (as it will be shown in the following sections), while the use of (54) is restricted to linear instantaneous mixtures. The drawback of the methods based on SFD is the necessity of the estimation of a multi-variate density, which is quite difficult, and requires a high computation cost and a lot of data, when the dimension (*i.e.* the number of sources) increases. Practically, this method is applicable only for a small number of sources (at most 3 or 4).

B. Minimization-Projection approach

In the MP approach, the minimization stage (step 1) does not depend on the separating model. The projection stage for linear instantaneous mixtures requires calculating the matrix $\mathbf{B}$ which minimizes $E\left\{\|\mathbf{y} - \mathbf{B}\mathbf{x}\|^2\right\}$ (step 2), and then replacing $\mathbf{y}$ by $\mathbf{B}\mathbf{x}$ (step 3). The solution of step 2 is given by the following well-known lemma.

Lemma 1: The matrix $\mathbf{B}$ which minimizes $E\left\{\|\mathbf{y} - \mathbf{B}\mathbf{x}\|^2\right\}$ is:

$$\mathbf{B}_{\text{opt}} = E\left\{\mathbf{y}\mathbf{x}^T\right\}\left(E\left\{\mathbf{x}\mathbf{x}^T\right\}\right)^{-1} \tag{55}$$

Taking into account the scale indeterminacy, the final MP algorithm for separating linear instantaneous mixtures is given in Fig. 4.

Experimental results. We repeat the experiment of Section VII-A using the MP approach. The source signals are two uniform random signals with zero means and unit variances. The mixing matrix is the same as (52). The parameters of the separating algorithm are like Section VII-A: (a) Histogram estimation of SFD, (b) 500 point data block, and (c) $\mu = 0.1$. The averaged SNR's taken over 100 runs of the algorithm, and the number of required iterations for convergece are given in the Table I.

Here, for linear instantaneous mixtures and the histogram estimation of SFD, the MP approach shows no advantages or disadvantages to the gradient approach. However, as our experiences shows for more complicated mixtures (and for other estimation methods of SFD), the MP approach has, in general, better convergence behavior (as discussed at the end of Section V-B) and converges in fewer iterations, while the gradient approach leads to slightly higher output SNRs.

VIII. Application to convolutive mixtures

In this section, we show how the gradient and the MP approaches can be used in separating convolutive mixtures of two sources.

A. *Gradient approach*

To use the gradient approach, $\frac{\partial}{\partial \mathbf{B}_k} I(y_1(n), y_2(n-m))$ must be first calculated. Although this gradient is directly calculated in [10], here, we recalculate it using the general approach of Section V-A and Theorem 1, to show how this general approach can be used in convolutive mixtures.

For calculating this gradient, we first define the following notation for any signal $\mathbf{y}(n) = (y_1(n), y_2(n))^T$:

$$\mathbf{y}^{(m)}(n) \triangleq \begin{pmatrix} y_1(n) \\ y_2(n-m) \end{pmatrix} \tag{56}$$

In other words, $\mathbf{y}^{(m)}(n)$ shows a shift operation on the second component of $\mathbf{y}(n)$ with respect to the first one.

Now, the gradient of $I\left(\mathbf{y}^{(m)}(n)\right)$ must be calculated. Let $\hat{\mathbf{B}}_i = \mathbf{B}_i$ for $i \neq k$ and $\hat{\mathbf{B}}_k = \mathbf{B}_k + \mathcal{E}$, where $\mathcal{E}$ is a "small" matrix. Then, from (48) we conclude that $\hat{\mathbf{y}}(n) \triangleq [\hat{\mathbf{B}}(z)]\mathbf{x}(n) = \mathbf{y}(n) + \mathcal{E}\mathbf{x}(n-k)$. Denoting $\boldsymbol{\Delta}(n) = \mathcal{E}\mathbf{x}(n-k)$, we will have $\hat{\mathbf{y}}^{(m)}(n) = \mathbf{y}^{(m)}(n) + \boldsymbol{\Delta}^{(m)}(n)$, and hence from Theorem 1:

$$\begin{aligned} I\left(\hat{\mathbf{y}}^{(m)}(n)\right) - I\left(\mathbf{y}^{(m)}(n)\right) &= E\left\{\boldsymbol{\beta}_{\mathbf{y}^{(m)}}(n)^T \boldsymbol{\Delta}^{(m)}(n)\right\} = E\left\{\beta_{\mathbf{y}^{(m)},1}(n)\Delta_1^{(m)}(n)\right\} \\ &+ E\left\{\beta_{\mathbf{y}^{(m)},2}(n)\Delta_2^{(m)}(n)\right\} = E\left\{\beta_{\mathbf{y}^{(m)},1}(n)\Delta_1(n)\right\} + E\left\{\beta_{\mathbf{y}^{(m)},2}(n)\Delta_2(n-m)\right\} \\ &= E\left\{\beta_{\mathbf{y}^{(m)},1}(n)\Delta_1(n)\right\} + E\left\{\beta_{\mathbf{y}^{(m)},2}(n+m)\Delta_2(n)\right\} \end{aligned}$$

where $\boldsymbol{\beta}_{\mathbf{y}^{(m)}}(n)$ stands for $\boldsymbol{\beta}_{\mathbf{y}^{(m)}}\left(\mathbf{y}^{(m)}(n)\right)$. Note that in writing the above equations, the sources are assumed to be stationary. Now, we define:

$$\boldsymbol{\beta}_m(n) \triangleq \boldsymbol{\beta}_{\mathbf{y}^{(m)}}^{(-m)}(n) = \begin{pmatrix} \boldsymbol{\beta}_{\mathbf{y}^{(m)},1}(n) \\ \boldsymbol{\beta}_{\mathbf{y}^{(m)},2}(n+m) \end{pmatrix} \tag{57}$$

In other words, for obtaining $\boldsymbol{\beta}_m(n)$, the second component of $\mathbf{y}(n)$ must be shifted m times, then after calculating its SFD, the second component of SFD must be shifted back m times:

$$\begin{pmatrix} y_1(n) \\ y_2(n) \end{pmatrix} \xrightarrow{\text{Shift}} \begin{pmatrix} y_1(n) \\ y_2(n-m) \end{pmatrix} \xrightarrow{\text{SFD}} \begin{pmatrix} \beta_1^\star(n) \\ \beta_2^\star(n) \end{pmatrix} \xrightarrow{\text{Shift back}} \begin{pmatrix} \beta_1^\star(n) \\ \beta_2^\star(n+m) \end{pmatrix} \triangleq \boldsymbol{\beta}_m(n)$$

By using this notation, we can write:

$$\hat{I} - I = E\left\{\boldsymbol{\beta}_m(n)^T \boldsymbol{\Delta}(n)\right\} = E\left\{\boldsymbol{\beta}_m(n)^T \mathcal{E}\mathbf{x}(n-k)\right\} = \left\langle \mathcal{E}, E\left\{\boldsymbol{\beta}_m(n)\, \mathbf{x}(n-k)^T\right\}\right\rangle$$

and finally:

$$\frac{\partial}{\partial \mathbf{B}_k} I\big(y_1(n), y_2(n-m)\big) = E\left\{\boldsymbol{\beta}_m(n)\, \mathbf{x}(n-k)^T\right\} \tag{58}$$

```
• Initialization:
    1) $\mathbf{B}_0 = \mathbf{I}$.
    2) For $k = 1 \dots p$, $\mathbf{B}_k = \mathbf{0}$.
    3) $\mathbf{y}(n) = \mathbf{x}(n)$.
• Loop:
    1) Choose a random $m$ from the set $\{-M, \dots, +M\}$.
    2) Estimate $\boldsymbol{\beta}^\star(n)$, the SFD of $\big(y_1(n), y_2(n-m)\big)^T$.
    3) Let $\boldsymbol{\beta}_m(n) = (\beta_1^\star(n), \beta_2^\star(n+m))^T$.
    4) For $k = 0 \dots p$ :
           $\mathbf{B}_k \leftarrow \mathbf{B}_k - \mu E\left\{\boldsymbol{\beta}_m(n)\,\mathbf{x}(n-k)^T\right\}$
    5) Calculate $\mathbf{y}(n) = [\mathbf{B}(z)]\mathbf{x}(n)$.
    6) Normalization:
        – Let $y_i = y_i/\sigma_i$, where $\sigma_i^2$ is the energy of $y_i$.
        – Divide the $i$-th raw of $\mathbf{B}(z)$ by $\sigma_i$.
• Repeat until convergence.
```

Fig. 5. Gradient algorithm for separating convolutive mixtures.

Now, taking into account the scale indeterminacy, the separation algorithm is obtained as shown in Fig. 5.

Experimental results. Here, an experimental result with the algorithm of Fig. 5 is presented. Because of the filtering indeterminacy, instead of (53), the SNR is defined here as:

$$\mathrm{SNR}_i = 10 \log_{10} \frac{E\left\{y_i^2\right\}}{E\left\{y_i^2|_{s_i=0}\right\}} \tag{59}$$

where $y_i|_{s_i=0}$ stands for what is at the i-th output, where the i-th input is zero (assuming there is no permutation). By using this definition, SNR will be a measure of "separation", that is, a high SNR means that there is not a large leakage from the other sources to this output. However, it can be a filtered version of the actual source, and a post-processing can be done for recovering the effect of this source on each sensor [25] (as mensioned at the begining of this section).

Now, we mix two uniform random sources with zeros means and unit variances. The mixing system is:

$$\mathbf{A}(z) = \begin{pmatrix} 1 + 0.2z^{-1} + 0.1z^{-2} & 0.5 + 0.3z^{-1} + 0.1z^{-2} \\ 0.5 + 0.3z^{-1} + 0.1z^{-2} & 1 + 0.2z^{-1} + 0.1z^{-2} \end{pmatrix} \tag{60}$$

The separation parameters are: Second order filters ($p = 2$), $M = 2p + 1 = 5$, $\mu = 0.3$, 500 samples data block, and the Pham's method [22] for estimating SFD. The averaged SNR's (taken over 100 repetitions of the experiment), and the number of required iterations for convergence are given in the Table I.

B. Minimization-Projection Approach

In this section, an algorithm based on the MP approach (Section V-B) is developed for separating convolutive mixtures. The minimization step of this algorithm (Equation (46)) does not depend on the separating model. The projection step for convolutive mixtures consists of first finding the filter $\mathbf{B}(z)$ which minimizes the error $E\left\{\|\mathbf{y}(n) - [\mathbf{B}(z)]\,\mathbf{x}(n)\|^2\right\}$, and then replacing $\mathbf{y}(n)$ by $[\mathbf{B}(z)]\mathbf{x}(n)$.

After doing some algebric calculations, it can be found that the filter $\mathbf{B}(z)$ which minimizes $E\left\{\|\mathbf{y}(n) - [\mathbf{B}(z)]\,\mathbf{x}(n)\|^2\right\}$ is given by the following system of linear equations (see [26] for details):

$$\sum_{j=0}^{p} \mathbf{B}_j \mathbf{R}_{\mathbf{xx}}(j,k) = \mathbf{R}_{\mathbf{yx}}(0,k) \quad , \quad k = 1, \dots, p \tag{61}$$

```
• Initialization: $\mathbf{y}(n) = \mathbf{x}(n)$.
• Loop:
    1) Choose a random $m$ from the set $\{-M, \dots, +M\}$.
    2) Estimate $\boldsymbol{\beta}_{\mathbf{y}^{(m)}}$, the SFD of $\big(y_1(n), y_2(n-m)\big)^T$.
    3) Update the outputs by:
           $\mathbf{y}^{(m)} \leftarrow \mathbf{y}^{(m)} - \mu \boldsymbol{\beta}_{\mathbf{y}^{(m)}}(\mathbf{y}^{(m)})$
    4) Remove the DC of each output, and normalize its energy.
    5) Compute the matrices $\mathbf{B}_k$, $k = 0, \dots, p$, from (61).
    6) Let $\mathbf{y}(n) = [\mathbf{B}(z)]\,\mathbf{x}(n)$.
• Repeat until convergence.
```

Fig. 6. Projection algorithm for separating convolutive mixtures.

where:

$$\mathbf{R}_{\mathbf{xx}}(j,k) \triangleq E\left\{\mathbf{x}(n-j)\mathbf{x}^T(n-k)\right\} \tag{62}$$

$$\mathbf{R}_{\mathbf{yx}}(j,k) \triangleq E\left\{\mathbf{y}(n-j)\mathbf{x}^T(n-k)\right\} \tag{63}$$

Finally, taking into account the scale indeterminacy, the MP algorithm for separating convolutive mixtures is obtained as shown in Fig. 6.

Experimental results. The experiment of Section VIII-A is repeated here using the MP method. The separation parameters are: Second order filters ($p = 2$), $M = 2p+1 = 5$, $\mu = 0.1$, 500 samples data block, and the Pham's method [22] for estimating SFD. The averaged SNR's (taken over 100 runs of the algorithm), and the number of required iterations for convergence are given in Table I.

A look at the Table I shows that the quality of the projection approach is slightly less than the gradient approach. However, as discussed at the end of section V-B, it has better convergence behavior.

IX. Application to PNL mixtures

In this section, we consider the application of the general mutual information minimization approaches of Section V (gradient and MP approaches) in separating PNL mixtures (Fig. 2).

A. Gradient approach

The gradient approach for separating PNL mixtures has been first reported in [31]. The parameters of the separating system (Fig. 2) are the matrix $\mathbf{B}$ and the functions g_i's, and for using the gradient approach, the gradients of $I(\mathbf{y})$ must be calculated with respect to these parameters.

The gradient with respect to $\mathbf{B}$ can be obtained by repeating the calculations done in Example 1 of Section V-A, and hence (43) gives the desired gradient (or (50) for the relative gradient).

For calculating the gradients of $I(\mathbf{y})$ with respect to the functions g_i, two different approaches may be used. The first is a 'parametric' approach, in which, a parametric model for these functions is assumed, and then the gradients of $I(\mathbf{y})$ with respect to these parameters are calculated. Finally the steepest descent gradient algorithm is applied on each parameter of the model. For example, neural networks and polynomial models have already been used for modeling g_i's [16], [32].

Another approach, which is used in this paper, is 'non-parametric' [33], [31]. In this approach, no parametric model is assumed for g_i's, and the 'gradient' of the criterion with respect to these functions, which is itself a function, is directly calculated. To clarify the idea, suppose that a cost function $\mathcal{C}$ is to be minimized with respect to a function $g(x)$. Moreover, suppose that for any 'small' function $\epsilon(x)$ we have:

$$\mathcal{C}(g+\epsilon)-\mathcal{C}(g)=\int_{-\infty}^{+\infty}\epsilon(x)\,f(x)\,p(x)\,dx+o(\epsilon) \tag{64}$$

where $p(x)$ is a non-negative function and $o(\epsilon)$ stands for higher order terms. Then, it can be said that the gradient of $\mathcal{C}$ with respect to the function $g(x)$ is the function $f(x)p(x)$. Equivalently, it can be said that the gradient of $\mathcal{C}$ with respect to g via the weighting function $p(x)$ is $f(x)$. The main point here is that a little movement in the opposite direction of these gradients (that is, $g \leftarrow g - \mu f p$ or $g \leftarrow g - \mu f$) insures a reduction in the cost function.

Now, for calculating the gradients of $I(\mathbf{y})$ with respect to the functions g_i, suppose there is a small perturbation in these functions of the form $\hat{g}_i = g_i + \epsilon_i \circ g_i$, where ϵ_i denotes a 'small' function. This is equivalent to $\hat{x}_i = x_i + \epsilon_i(x_i)$. Using Theorem 1 and after doing some algebric calculations (see [31] for details), we will have:

$$I(\hat{\mathbf{y}})-I(\mathbf{y})=\sum_i\int_{-\infty}^{\infty}\epsilon_i(x)E\left\{\alpha_i(\mathbf{y})\mid x_i=x\right\}p_{x_i}(x)dx \tag{65}$$

where:

$$\boldsymbol{\alpha}(\mathbf{y}) \triangleq \mathbf{B}^T\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y}) \tag{66}$$

From the above equation, we conclude that the (natural) gradient of $I(\mathbf{y})$ with respect to g_i via the weighting function p_{x_i} is:

$$(\nabla_{g_i}I)(x)=E\left\{\alpha_i(\mathbf{y})\mid x_i=x\right\} \tag{67}$$

We emphasize again thet $\nabla_{g_i}I$ is itself a function.

In PNL mixtures, there are scale indeterminacies on both x_i and y_i signals, and a mean (DC) indeterminacy on x_i [16]. Moreover, when using nonparametric methods, a smoothness constraint is necessary on functions g_i [31]. Without any smoothness constraint, the value of g_i at each data point acts as a parameter, and the number of parameters will be equal to the number of data points, which is mathematically ill-posed. To insure this smoothness, smoothing splines [21] may be used [31]: at each iteration, g_i is the smoothing spline which has the best fit on the data points (x_i, α_i).

Taking into account these indeterminacies, the separation algorithm is obtained as shown in Fig. 7.

Experimental results. As an experiment, two uniform random sources with zero means and unit variances are mixed. The mixing system composed of:

$$\mathbf{A}=\begin{pmatrix}1 & 0.5\\ 0.5 & 1\end{pmatrix},\quad f_1(x)=f_2(x)=0.1x+\tanh(2x) \tag{68}$$

Then, we have used the separation algorithm of Fig. 7 with 1000 data samples, kernel estimation of SFD (by Gaussian kernels), and $\mu_1=\mu_2=0.2$.

The separation quality is measured by mean of output SNR's, where each output SNR is defined in (53). The averaged SNR's taken over 100 runs of the algorithm, and the number of required iterations for convergence are given in the Table I.

```
• Initialization:
    1) $\mathbf{B} = \mathbf{I}$
    2) $\mathbf{x} = \mathbf{e}$
    3) $\mathbf{y} = \mathbf{x}$
• Loop:
    1) Estimate $\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})$ (SFD of $\mathbf{y}$).
    2) Let $\boldsymbol{\alpha}_{\mathbf{y}}(\mathbf{y}) \triangleq \mathbf{B}^T \boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})$.
    3) For $i = 1, \dots, N$, estimate $(\nabla_{g_i} I)(x_i)$, by fitting a smoothing spline on $(x_i, \alpha_i)$.
    4) For $i = 1, \dots, N$, modify $x_i$ by $x_i \leftarrow x_i - \mu_2 (\nabla_{g_i} I)(x_i)$.
    5) Let $g_i$ be the smoothing spline which fits on the $(e_i, x_i)$ points.
    6) For $i = 1, \dots, N$, normalize $x_i$ by $x_i \leftarrow \frac{x_i - \hat{\mu}_{x_i}}{\hat{\sigma}_{x_i}}$.
    7) Absorb the effect of the above normalization in $\mathbf{B}$:
         $$\mathbf{B} \leftarrow \mathbf{B} \begin{pmatrix} \hat{\sigma}_{x_1} & & 0 \\ & \ddots & \\ 0 & & \hat{\sigma}_{x_N} \end{pmatrix}$$
    8) Estimate $\nabla_{\mathbf{B}} I$, using (50).
    9) Modify $\mathbf{B}$ by $\mathbf{B} \leftarrow (\mathbf{I} - \mu_1 \nabla_{\mathbf{B}} I)\mathbf{B}$.
   10) Compute outputs by $\mathbf{y} = \mathbf{B}\mathbf{x}$.
   11) Normalization:
         – Let $y_i = y_i / \sigma_i$, where $\sigma_i^2$ is the energy of $y_i$.
         – Divide the $i$-th raw of $\mathbf{B}$ by $\sigma_i$.
• Repeat until convergence
```

Fig. 7. The gradient approach for separating PNL mixtures.

B. Minimization-Projection Approach

In this section, an algorithm for separating PNL mixtures is developed based on the MP approach (Section V-B). As usual, the minimization step of this algorithm (Equation (46)) is the same for all mixing-separating models. The projection step for PNL mixtures (see Fig. 2), consists in first finding functions g_i and matrix $\mathbf{B}$ which minimize $E\left\{\|\mathbf{y} - \mathbf{B}\mathbf{g}(\mathbf{e})\|^2\right\}$, and then replacing $\mathbf{y}$ by $\mathbf{B}\mathbf{g}(\mathbf{e})$, where $\mathbf{g}(\mathbf{e}) \triangleq (g_1(e_1), \dots, g_N(e_N))^T$.

For finding functions g_i and matrix $\mathbf{B}$ which optimally map $\mathbf{e}$ to $\mathbf{y}$, an iterative algorithm is proposed in [23]. It is based on the fact that the invertibility of g_i's implies their monotonicity, and without loss of generality they can be assumed ascending. If $\mathbf{x}$ was known, then the matrix $\mathbf{B}$ which optimally maps $\mathbf{x}$ to $\mathbf{y}$ were given by (55). Knowing $\mathbf{B}$, $\mathbf{x}$ can be re-calculated using $\mathbf{x} = \mathbf{B}^{-1}\mathbf{y}$. It automatically defines a set of g_i's, which are not necessarily ascending. To make g_i ascending, we change the order of the values $x_i(1), x_i(2), \dots, x_i(T)$ (T is the length of data block) in such a way that for any $e_i(k_1) > e_i(k_2)$ we have $x_i(k_1) > x_i(k_2)$. This may be better explained by its MATLAB code:

```
[temp, index_i] = sort(e_i);
x_i(index_i) = sort(x_i);
```

It defines a new $\mathbf{x}$, and the above process is repeated. This results in the algorithm of Fig. 8 for finding the optimal mapping. It is experimentally shown in [23] that this algorithm converges very fast, typically in 2 to 5 iterations.

Having this algorithm for finding the projected mapping, the final MP algorithm for separating PNL mixtures can be easily obtained. However, two modifications is done in the algorithm of Fig. 8: (a) instead of initializing by $\mathbf{x} = \mathbf{e}$, the value of $\mathbf{x}$

```
• Initialization: $\mathbf{x} = \mathbf{e}$.
• Loop:
    1) Let $\mathbf{B} = E\left\{\mathbf{y}\mathbf{x}^T\right\}\left(E\left\{\mathbf{x}\mathbf{x}^T\right\}\right)^{-1}$.
    2) Let $\mathbf{x} = \mathbf{B}^{-1}\mathbf{y}$.
    3) For $i = 1, \ldots, N$, change the order of $x_i(k)$ such that the function $x_i = g_i(e_i)$ become ascending (see the text).
• Repeat until convergence
```

Fig. 8. Finding the projected PNL mapping.

```
• Initialization: $\mathbf{y} = \mathbf{x} = \mathbf{e}$.
• Loop:
    1) Estimate $\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})$, the SFD of $\mathbf{y}$.
    2) Modify the outputs by $\mathbf{y} \leftarrow \mathbf{y} - \mu\boldsymbol{\beta}_{\mathbf{y}}(\mathbf{y})$.
    3) For $k = 1, \ldots, K$, do:
        a) Let $\mathbf{B} = E\left\{\mathbf{y}\mathbf{x}^T\right\}\left(E\left\{\mathbf{x}\mathbf{x}^T\right\}\right)^{-1}$.
        b) Let $\mathbf{x} = \mathbf{B}^{-1}\mathbf{y}$.
        c) For $i = 1, \ldots, N$, change the order of $x_i(k)$ such that the function $x_i = g_i(e_i)$ be ascending.
    4) For $i = 1, \ldots, N$, remove the mean of $x_i$ and normalize its energy.
    5) For $i = 1, \ldots, N$, let $g_i$ be the smoothing spline which fits on $(e_i, x_i)$.
    6) Let $\mathbf{y} = \mathbf{B}\mathbf{g}(\mathbf{e})$.
    7) For $i = 1, \ldots, N$, remove the mean of $y_i$ and normalize its energy.
• Repeat until convergence
```

Fig. 9. Minimization-Projection algorithm for separating PNL mixtures.

obtained in the previous iteration of MP algorithm is used, which is a better initial estimation of $\mathbf{x}$. (b) we do not wait for the projection algorithm of Fig. 8 to converge (even, it is possible that it does not converge, when the outputs cannot expressed as $\mathbf{B}\mathbf{g}(\mathbf{e})$). Instead, we simply repeat the loop for some fixed number of iterations, say 5 or 10 times (even 1 iteration seems sufficient in many cases, because the whole MP algorithm is itself iterative, and the value of $\mathbf{x}$ in the previous iteration is used as initial value in the current iteration).

Finally taking into account the indeterminacies (similir to what is done for the gradient approach), the final separation algorithm is obtained as shown in Fig. 9.

Experimental results. Now we repeat the experiment of Section IX-A using the MP algorithm with parameters: $\mu = 0.1$, 1000 samples data block, Pham's estimation of SFD, and $K = 5$. The averaged SNR's taken over 100 runs of the algorithm, and the number of required iterations for convergece are given in the Table I.

X. Application to CPNL mixtures

We terminate the applications of gradient and MP approaches by using them in separating Convolutive Post Non-Linear (CPNL) mixtures of two sources. In CPNL mixtures, the separation system composed of the linear FIR filter (48) and non-linear functions g_i. Consequently, the separation algorithms can be obtained by combining the algorithms obtained for convolutive and PNL mixtures.

```
• Initialization:
    1) $\mathbf{B}(z) = \mathbf{I}$, that is: $\mathbf{B}_0 = \mathbf{I}$ and for $k = 1 \dots p$, $\mathbf{B}_k = \mathbf{0}$.
    2) $\mathbf{x}(n) = \mathbf{e}(n)$.
    3) $\mathbf{y}(n) = \mathbf{x}(n)$.
• Loop:
    1) Choose a random $m$ from the set $\{-M, \dots, +M\}$.
    2) Compute $\boldsymbol{\beta}^\star(n)$, the SFD of $\big(y_1(n), y_2(n-m)\big)^T$.
    3) Let $\boldsymbol{\beta}_m(n) = (\beta_1^\star(n), \beta_2^\star(n+m))^T$.
    4) Estimate $(\nabla_{g_i} I)(x_i)$, by fitting a smoothing spline on $(x_i, \alpha_i)$.
    5) Modify $x_i$: $x_i \leftarrow x_i - \mu_1(\hat{\nabla}_{g_i} I)(x_i)$.
    6) Let $g_i$ be the smoothing spline which fits on the $(e_i, x_i)$ points.
    7) Normalize $x_i$: $x_i \leftarrow (x_i - \hat{\mu}_{x_i}) / \hat{\sigma}_{x_i}$.
    8) Absorb the effect of the above normalization in $\mathbf{B}(z)$:
           $$\mathbf{B}(z) \leftarrow \mathbf{B}(z) \begin{pmatrix} \hat{\sigma}_{x_1} & & 0 \\ & \ddots & \\ 0 & & \hat{\sigma}_{x_N} \end{pmatrix}$$
    9) For $k = 0 \dots p$ :
           $$\mathbf{B}_k \leftarrow \mathbf{B}_k - \mu_2 \hat{E}\left\{\boldsymbol{\beta}_m(n)\,\mathbf{x}(n-k)^T\right\}$$
    10) Calculate $\mathbf{y}(n) = [\mathbf{B}(z)]\mathbf{x}(n)$.
    11) Normalization:
        – Let $y_i = y_i / \sigma_i$, where $\sigma_i^2$ is the energy of $y_i$.
        – Divide the $i$-th raw of $\mathbf{B}(z)$ by $\sigma_i$.
• Repeat until convergence.
```

Fig. 10. Gradient algorithm for separating CPNL mixtures.

A. Gradient Approach

Similar to convolutive mixtures, the separation criterion is $I(y_1(n), y_2(n-m)))$, where m is randomly chosen at each iteration from $-(2p+1)$ to $2p+1$. To use the gradient approach in separating CPNL mixtures, the gradient of this criterion must be calculated with respect to both $\mathbf{B}_k$'s and g_i's.

Following the same calculations done in Section VIII-A, the gradient with respect to $\mathbf{B}_k$'s is given by (58). The gradient with respect to functions g_i can be obtained using a method similar to what is done for PNL mixtures at Section IX-A. The final result is given by (67), where in this case $\boldsymbol{\alpha}$ is [28]:

$$\boldsymbol{\alpha}(n) \triangleq \sum_{k=0}^{p} \mathbf{B}_k^T \boldsymbol{\beta}_m(n+k) = \left[\mathbf{B}^T\left(\frac{1}{z}\right)\right] \boldsymbol{\beta}_m(n) \tag{69}$$

Taking into account the DC and scale indeterminacies, the final separation algorithm is obtained as shown in Fig. 10.

Experimental results. As an experiment, two uniform radom sources with zero means and unit variances are mixed by a system composed of the mixing matrix (60) and sensor nonlinearities given by (68). Each output SNR is defined as (59). Moreover, there are a lot of parameters to be estimated and to not encounter "over-learning", a new data set is used at each iteration. Here, the main parameters are: observation blocks of 2000 samples, $p = 2$, $M = 5$ and $\mu = 0.2$. The averaged SNR's taken over 100 runs of the algorithm, and the number of required iterations for convergence are given in the Table I.

```
- Initialization: $\mathbf{y} = \mathbf{x} = \mathbf{e}$.
- Loop:
    1) Choose a random $m$ from the set $\{-M, \ldots, M\}$.
    2) Estimate $\boldsymbol{\beta}_{\mathbf{y}^{(m)}}(\mathbf{y})$, the SFD of $\mathbf{y}^{(m)}$.
    3) Modify the outputs by $\mathbf{y}^{(m)} \leftarrow \mathbf{y}^{(m)} - \mu\boldsymbol{\beta}_{\mathbf{y}^{(m)}}(\mathbf{y}^{(m)})$.
    4) For $k = 1, \ldots, K$, do:
        a) Find the best $\mathbf{B}(z)$ which maps $\mathbf{x}(n)$ to $\mathbf{y}(n)$, using (61).
        b) Compute $\mathbf{x}(n) = [\text{inv}(\mathbf{B}(z))]\,\mathbf{y}(n)$ using the recursive equation (70).
        c) For $i = 1, 2$, change the order of $x_i(k)$ such that the function $x_i = g_i(e_i)$ be ascending.
    5) For $i = 1, 2$, remove the DC of $x_i$ and normalize its energy.
    6) For $i = 1, 2$, let $g_i$ be the smoothing spline which fits on $(e_i, x_i)$.
    7) Let $\mathbf{y}(n) = [\mathbf{B}(z)]\,\mathbf{g}\big(\mathbf{e}(n)\big)$.
    8) For $i = 1, 2$, remove the DC of $y_i$ and normalize its energy.
- Repeat until convergence
```

Fig. 11. Minimization-Projection algorithm for separating CPNL mixtures.

TABLE I

SUMMARY OF THE RESULTS OF THE EXPERIMENTS OF THE PAPER. SNR'S ARE AVERAGED OVER 100 EXPERIMENTS.

Mixture type	Algorithm	SNR (in dB)	Number of iterations
Linear Instantaneous	Gradient	32.4	31
	MP	31.7	22
Convolutive	Gradient	25.4	86
	MP	20.4	108
PNL	Gradient	21.4	22
	MP	18.7	19
CPNL	Gradient	24	1090
	MP	24.7	328

B. Minimization-Projection Approach

The MP approach for separating CPNL mixtures can be obtained by directly combining the corresponding approaches for separating convolutive and PNL mixtures. However, step (2) of the projection algorithm of Fig. 8 (for finding the optimum mapping) must be replaced by $\mathbf{x}(n) = [\text{inv}(\mathbf{B}(z))]\,\mathbf{y}(n)$. This inverse function can be found using the recursive equation:

$$\mathbf{x}(n) = \mathbf{B}_0^{-1}\big[\mathbf{y}(n) - \mathbf{B}_1\mathbf{x}(n-1) - \cdots - \mathbf{B}_p\mathbf{x}(n-p)\big] \qquad (70)$$

Consequently, the final MP separation algorithm is obtained as shown in Fig. 11.

Experimental results. The same experiment of Section X-A is repeated here, but with block lengths of 1000 samples and $\mu = 0.1$. The averaged SNR's taken over 100 runs of the algorithm, and the number of required iterations for convergece are given in the Table I.

XI. Conclusion

In this paper, we introduced the concept of SFD, which can be considered as a non-parametric 'gradient' of the mutual information. We also showed that SFD is a powerful tool for minimizing mutual information of a parametric model, and we proposed two general approaches called gradient and Minimization-Projection (MP) approaches. These approaches give a unifying view and powerful tools for separating any complicated (yet separable) mixing model (for example, up to our best knowledge there is currently no other method for separating CPNL mixtures). Finally, we showed how these approaches can be used for separating sources in four mixing models: linear instantaneous, linear convolutive, PNL, and CPNL mixtures.

Moreover using the properties of SFD, it was shown that mutual information has no local minimum (Theorem 2).

In general, because of direct manipulation of outputs in the MP method, and since mutual information has no local minimum, this method has better convergence behavior (faster convergence and lowest risk of a local minimum) than gradient method, especially when the number of parameters are large (*e.g.* convolutive mixtures with long separating filters). Moreover, in the gradient-based algorithms, it is very tricky to select parameters which insures the algorithm convergence. On the contrary, the convergence of MP algorithms is very robust with respect to the parameters. However, if convergence is achieved, the gradient approach may result in a slightly better separation performance. Finally, for linear instantaneous mixtures, gradient approach may be simplified (as done in (54)) such that it only uses marginal distribution estimations, which is a great advantage for large number of sources.

The limitation of methods based on mutual information minimization with SFD is the requirement of multi-variate PDF estimations which is quite difficult and requires a large number of data when the number of variables grows. Practically, they are currently limited to 3 or 4 sources.

References

[1] A. Hyvärinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, John Wiely & Sons, 2001.

[2] C. Jutten, M. Babaie-Zadeh, and S. Hosseini, "Three easy ways for separating nonlinear mixtures?," *Signal Processing*, vol. 84, no. 2, pp. 217–229, February 2004.

[3] P. Comon, "Independent component analysis, a new concept?," *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[4] L. B. Almeida, "MISEP – linear and nonlinear ica based on mutual information," *Journal of Machine Learning Research*, vol. 4, pp. 1297–1318, December 2003.

[5] H. Valpola and J. Karhunen, "An unsupervised ensemble learning method for nonlinear dynamic state-space models," *Neural Computation*, vol. 14, no. 11, pp. 2647–2692, 2002.

[6] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley Series in Telecommunications, 1991.

[7] J.-F. Cardoso, "Blind signal separation: statistical principles," *Proceedings IEEE*, vol. 9, pp. 2009–2025, 1998.

[8] D. T. Pham, "Mutual information approach to blind separation of stationary sources," in *Proceedings of ICA'99*, Aussois, France, January 1999, pp. 215–220.

[9] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, "Differential of mutual information function," *IEEE Signal Processing Letters*, vol. 11, no. 1, pp. 48–51, January 2004.

[10] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, "Separating convolutive mixtures by mutual information minimization," in *Proceedings of IWANN'2001*, Granada, Spain, Juin 2001, pp. 834–842.

[11] A. Taleb and C. Jutten, "Entropy optimization, application to blind source separation," in *Proceedings of ICANN'97*, Lausanne, Switzerland, October 1997, pp. 529–534.

[12] D. D. Cox, "A penalty method for nonparametric estimation of the logarithmic derivative of a density function," *Ann. Instit. Statist. Math.*, vol. 37, pp. 271–288, 1985.

[13] D. T. Pham, “Mutual information approach to blind separation of stationary sources,” *IEEE Transactions on Information Theory*, vol. 48, no. 7, pp. 1–12, July 2002.

[14] E. G. Learned-Miller and J. W. Fisher III, “ICA using spacings estimates of entropy,” *Journal of Machine Learning Research*, vol. 4, pp. 1271–1295, December 2003.

[15] F. Vrins and M. Verleysen, “On the entropy minimization of a linear mixture of variables for source separation,” In this issue of *Signal Processing*.

[16] A. Taleb and C. Jutten, “Source separation in post nonlinear mixtures,” *IEEE Transactions on Signal Processing*, vol. 47, no. 10, pp. 2807–2820, 1999.

[17] W. Härdle, *Smoothing techniques with implementation in S*, Springer-Verlag, 1991.

[18] B. W. Silverman, *Density estimation for statistics and data analysis*, Chapman and Hamm, 1986.

[19] K. Fukunaga, *Introduction to statistical pattern recognition*, New York: Academic Press, 1972.

[20] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, 2002.

[21] R. L. Eubank, *Spline smoothing and nonparanetric regression*, Dekker, 1988.

[22] D. T. Pham, “Fast algorithm for estimating mutual information, entropies and score functions,” in *Proceedings of ICA2003*, Nara, Japan, April 2003, pp. 17–22.

[23] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, “Minimization-projection (MP) approach for blind source separation in different mixing models,” in *ICA2003*, April 2003, pp. 1083–1088.

[24] D. Yellin and E. Weinstein, “Criteria for multichannel signal separation,” *IEEE Trans. Signal Processing*, vol. 42, pp. 2158–2168, August 1994.

[25] C. Simon, *Séparation aveugle des sources en mélange convolutif*, Ph.D. thesis, Université de Marne la Vallée, Novembre 1999, (in French).

[26] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, “A minimization-projection (MP) approach for blind separating convolutive mixtures,” in *Proceedinds of ICASSP’04*, Montreal (Canada), May 2004, vol. 5, pp. 533–536.

[27] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, “A geometric approach for separating Post Non-Linear mixtures,” in *EUSIPCO*, Toulouse, France, September 2002, vol. II, pp. 11–14.

[28] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, “Blind separating Convolutive Post-Nonlinear mixtures,” in *Proceedings of ICA2001*, San Diego (Ca, USA), December 2001, pp. 138–143.

[29] J.-F. Cardoso and B. Laheld, “Equivariant adaptive source separation,” *IEEE Trans. on SP*, vol. 44, no. 12, pp. 3017–3030, December 1996.

[30] S. I. Amari, “Natural gradient works efficiently in learning,” *Neural Computation*, vol. 10, pp. 251–276, 1998.

[31] M. Babaie-Zadeh, C. Jutten, and K. Nayebi, “Using multivariate score functions in source separation: Application to post non-linear mixtures,” *Scientia-Iranica*, vol. 9, no. 4, pp. 409–418, 2002.

[32] J. Sole, C. Jutten, and A. Taleb, “Parametric approach to blind deconvolution of nonlinear channels,” *Neurocomputing*, vol. 48, pp. 339–355, 2002.

[33] A. Taleb and C. Jutten, “Batch algorithm for source separation in postnonlinear mixtures,” in *Proceedings of ICA’99*, Aussois, France, January 1999, pp. 155–160.