Noname manuscript No.
(will be inserted by the editor)

A Fast Gradient Approximation for Nonlinear Blind Signal Processing

Jordi Solé-Casals · Cesar F. Caiafa

Received: date / Accepted: date

Abstract When dealing with nonlinear blind processing algorithms (deconvolution or post-nonlinear source separation) complex mathematical estimations must be done giving as a result very slow algorithms. This is the case, for example, in speech processing, spike signals deconvolution or microarray data analysis. In this paper, we propose a simple method to reduce computational time for the inversion of Wiener systems or the separation of post-nonlinear mixtures, by using a linear approximation in a minimum-mutual information algorithm. Simulation results demonstrate that linear spline interpolation is fast and accurate, obtaining very good results (similar to those obtained without approximation) while computational time is dramatically decreased. On the other hand, cubic spline interpolation also obtains similar good results, but due to its intrinsically complexity the global algorithm is much more slow and hence not useful for our purpose.

Keywords blind deconvolution · blind source separation · minimum mutual information methods · Wiener systems

This work has been in part supported by the MINCYT-MICINN Research Program 2009-2011 (Argentina-Spain): *Desarrollo de Herramientas de Procesado de Señales para el Análisis de Datos Bioinformáticos* (Ref. AR2009-0010), by the University of Vic under the grant R0904 and by CONICET under the grant PIP 2012-2014, number 11420110100021.

Jordi Solé-Casals
Digital Technologies Group
University of Vic, Sagrada Famlia 7, 08500, Vic, SPAIN.
Tel.: +34 93 881 55 19
Fax: +34 93 885 69 00
E-mail: jordi.sole@uvic.cat

Cesar F. Caiafa
Instituto Argentino de Radioastronomía (CCT La Plata, CONICET)
C.C.5, (1894) Villa Elisa, Buenos Aires, ARGENTINA.
Tel.: +54 221 482 4903
Fax: +54 221 425 4909
E-mail: ccaiafa@iar.unlp.edu.ar
C. F. Caiafa is also within Facultad de Ingeniería - UBA, Av. Paseo Colón 850. 4to. piso, Ala sur (1063), Capital Federal, ARGENTINA

1 Introduction

Blind Signal Processing (BSP) comprises a family of unsupervised adaptive filtering algorithms designed to be applied to observed signals without much information about the hidden source signals and the system coefficients. This is the case in Blind Deconvolution (BD) [5], where one searches for an estimation of an input signal based only on its filtered version (output signal) or one just want to identify the linear system (System Identification (SI)); or in Blind Source Separation (BSS) [6, 1] where the objective is to estimate source signals from a set of their linear mixtures. Although the blind assumption represents a challenge for the algorithm design, these problems have been intensively considered in the last thirty years and, today, we have available a wide family of algorithms that solve the problem in its simplest version which is called Linear Blind Signal Processing (L-BSP), where the involved systems are linear, i.e. observed signals are obtained by a matrix operation and/or linear filtering applied to the source signals (instantaneous as well as convolutive cases).

More recently, an even more difficult problem has been considered by few researchers by including unknown nonlinear distortions at some stage in the linear system. These new models have motivated researchers to develop Nonlinear Blind Signal Processing (NL-BSP) algorithms as it is the case of Nonlinear Blind Deconvolution (NL-BD) [13, 14, 12, 9] and Post-Nonlinear Blind Source Separation (PNL-BSS) [10, 11] which are the subject of study in this paper, as a continuation of our previous work [8].

Under the assumption of independent and identically distributed (iid) samples in NL-BD or source independence in NL-BSS, it is known that algorithms based on information theoretic criteria are able to recover the desired signals. These algorithms pose the separation as an optimization problem where the Mutual Information (MI) between samples in the estimated source signal (BD) [12, 9], or between estimated sources (NL-BSS)[10, 11], is minimized as a function of the inverse system coefficients.

When dealing with MI minimization algorithms in an NL-BSP context, the score function (see its definition in the next section) appears as an important parameter to be estimated. In these cases, due to the nonlinear part that must be compensated, the gradient equations that come out in the optimization procedure are much more complex than in the pure linear case (L-BSP). More specifically, when a nonlinear distortion is considered, the gradient of MI has an expectation form where the score function appears jointly with other functions that depend on the inverse system parameters [12, 9–11]. Those kinds of calculations are very time consuming, as many different terms are involved into the same equation. In this work we present a simple method to overcome this problem. The score function and the expectation term are calculated only at N equally spaced points covering the samples range. After that, we interpolate the result over the whole range of the domain.

The paper is organized as follows: in section 2, mathematical models, assumptions and notations are defined; in section 3, the minimum mutual information algorithms are presented for the case of the blind deconvolution problem (PNL problem is equivalent); in section 4, the interpolation method to reduce the complexity of the algorithms is introduced; in section 5, several simulation results are

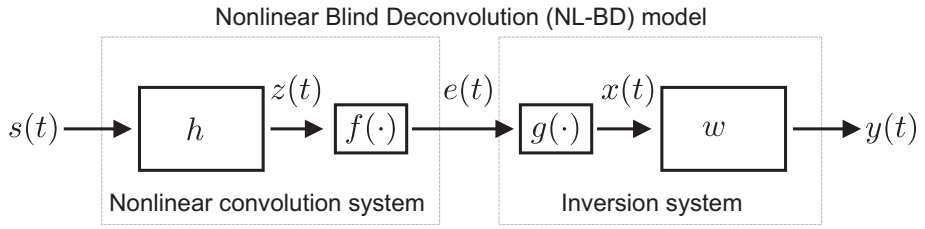

Fig. 1 The NL-BD model. The unknown nonlinear convolution system and the inversion system are shown in the left and in the right, respectively.

presented for the case of nonlinear deconvolution (PNL problem is equivalent), and finally, in section 6, discussion and main conclusions are outlined.

2 Models, Assumptions and Notation

The NL-BD model is shown in Fig. 1. Following the same notation used in [12, 8], we assume that the input of the system $\mathcal{S} = \{s(t)\}$ is an unknown non-Gaussian i.i.d. process. The linear filter

$$h = [h(-L), ..., h(-1), h(0), h(1), ..., h(L)],$$

is assumed to be unknown and invertible, i.e., h^{-1} exists such that $h^{-1} * h = h * h^{-1} = \delta_0$ (the Dirac impulse at $t = 0$), and h may have causal as well as anticausal parts that can be, eventually, of infinite length ($L = \infty$). Here we use "$*$" to denote the discrete convolution. On the other hand, the nonlinear distortion (memoryless) is defined by a nonlinear function $f : \mathbb{R} \to \mathbb{R}$ which is also assumed unknown and invertible. In this work, it is assumed that all involved stochastic processes $x(t)$, $y(t)$, etc., are wide sense stationary and ergodic, i.e. expectations can be computed by averaging over time, for example, for the case of the mean we have that $\mu = \mathbb{E}[x(t)] \approx \frac{1}{N}\sum_{n=1}^{N} x(t_n)$.

In order to estimate $s(t)$ by only observing the system output $e(t)$, we will use an inverse structure composed of the same kind of subsystems: a memoryless function g and a linear filter w (Fig. 1, right). The nonlinear part g is devoted to the compensation of the distortion f without access to its input, while the linear part w is a linear deconvolution filter. Such a system (Wiener system but in reverse order) is known as a Hammerstein system [3].

Additionally, we define the *cross-correlation function* between $x(t)$ and $y(t)$ as $\gamma_{x,y}(t) = \mathbb{E}[x(\tau - t)y(\tau)]$ and the *score function* of signal $y(t)$ as $\psi_y = (\log p_y)'(u) = p'_y(u)/p_y(u)$ where $p_y(u)$ is the *probability density function* of $y(t)$. It is noted that, in our case, the *score function* is independent of t since $y(t)$ is stationary.

The case of having multiple nonlinear observations of mixed input signals (PNL-BSS) can be treated equivalently to the case of the NL-BD problem. In Appendix A, we introduce the mathematical model and derive the algorithm for that case.

3 Minimum Mutual Information Algorithms

3.1 The Min-MI Nonlinear Blind Deconvolution Algorithm (Min-MI NL-BD)

It is known that the inverse system, composed of function g and the linear filter w, can be estimated by minimizing the mutual information (MI) of the output $\mathcal{Y} = \{y(t)\}$. In [12], the Min-MI NLBD (Minimum-Mutual Information Nonlinear Blind Deconvolution) algorithm was derived. The Min-MI NLBD algorithm requires to compute a special *perturbation signal* $\varepsilon(x(t))$ which is needed to compensate the nonlinear distortion. The *perturbation signal* is defined as follows:

$$\varepsilon(x(t)) = \mathbb{E}\left[\psi_y(y)(w * r(x - x(t))) + r'(x - x(t))\right], \tag{1}$$

where, the functions $r(x)$ and $r'(x)$ are chosen as the *ramp* function and its derivative respectively, i.e. $r(x) = xH(x)$ and $r'(x) = H(x)$ with $H(x)$ being the Heaviside unit step function ($H(x) = 1$ if $x \geq 0$ and $H(x) = 0$ if $x < 0$).

Additionally, the deconvolution algorithm requires to compute the *cross-correlation* between the current output and its *score function*, i.e.

$$\gamma_{y,\psi_y}(t) = \mathbb{E}[y(\tau - t)\psi_y(y(\tau))], \tag{2}$$

which can be interpreted as a high-order correlation function of $y(t)$. In Algorithm 1, the steps of the Min-MI NL-BD Algorithm are shown.

Algorithm 1 `Min-MI NL-BD` **[12]**

INPUT: Maximum Number of Iterations $Niter$ and output signal $e(t)$
OUTPUT: Estimated input signal $\widehat{s}(t)$
1: $i = 1$; $w = \delta_0$; Inicialization
2: $y(t) = (w * e)(t)$;
3: **while** $i \leq Niter$ and NOT convergence is reached **do**
4: Estimate the *score function* $\psi_y(y(t))$ (eq. (4))
5: Compute the *perturbation signal* $\varepsilon(x(t))$ (eq. (1))
6: $x(t) \leftarrow x(t) + \mu_1\varepsilon(x(t))$; Nonlinear compensation[2]
7: $x(t) \leftarrow (x(t) - m_x)/\sigma_x$; Normalization[3]
8: Estimate the *cross-correlation* $\gamma_{y,\psi_y}(t)$ (eq. (2))
9: $w \leftarrow w + \mu_2\gamma_{y,\psi_y}$; Deconvolution Filter update[2]
10: $y(t) = (w * x)(t)$; Current Estimation of source signal
11: $i = i + 1$;
12: **end while**
13: $\widehat{s}(t) = y(t)$;

To compute equations (1) and (2) the *score function* $\psi_y(y)$ should be estimated in some manner since it is not known a priori. Here, we use a nonparametric approach based on Parzen windows [4, 2, 7]. This kernel density estimator is easy to implement and has a very flexible form. Formally, we estimate the *probability density function* $p_y(y)$ by:

$$\widehat{p}_y(y) = \frac{1}{BT}\sum_{\tau=1}^{T} K\left(\frac{y - y(\tau)}{B}\right), \tag{3}$$

and ψ_y by:

$$\widehat{\psi}_y(y) = \frac{\sum_{\tau=1}^{T} K'\left(\frac{y-y(\tau)}{B}\right)}{\sum_{\tau=1}^{T} K\left(\frac{y-y(\tau)}{B}\right)}, \tag{4}$$

where T is the number of available samples (the signal length in our case), $K(u)$ is a kernel (window) function that must obey certain properties and B is related to the width of the window. In our experiments we used Gaussian kernels, however many other kernel shapes can be good candidates. A "quick and dirty" method for the choice of the bandwidth consists in using the rule of thumb $B = 1.06\sigma_y T^{-1/5}$, which is based on the minimum asymptotic mean integrated error criterion [7].

From equation (4) it is easy to see that the computation of the score function at a specific point y_0, i.e. $\widehat{\psi}_y(y_0)$, requires to evaluate the Kernel T times and sum over T terms then giving a complexity of order $\mathcal{O}(T)$. Since the equations (1)-(2) require the score function to be computed at every available signal point $y(t)$ $(t = 1, 2, ..., T)$, the total complexity in the score function estimation is $\mathcal{O}(T^2)$ and is the same complexity that we will find computing the perturbation signal $\varepsilon(x(t))$ (1) and estimating the cross-correlation $\gamma_{y,\psi_y}(t)$ (2).

4 Complexity reduction by interpolation of measures

By assuming signals to be ergodic processes, we are able to compute expectations by averaging over time samples. Therefore, equations (1) and (2) can be estimated, respectively, by:

$$\varepsilon(x(t)) \approx \frac{1}{T}\sum_{\tau=1}^{T}\left[\widehat{\psi}_y(y(\tau))(w * r(x - x(t)))(\tau) + r'(x - x(t))(\tau)\right], \tag{5}$$

$$\gamma_{y,\psi_y}(t) \approx \frac{1}{T}\sum_{\tau=1}^{T} y(\tau - t)\widehat{\psi}_y(y(\tau)). \tag{6}$$

The score function (4) and the perturbation (5) must be evaluated at T points (one per signal sample). In order to reduce the quadratic complexity to linear complexity we propose to create a grid of N amplitude values covering the ranges of the variables by using evenly-spaced points (quantization) which give us complexity $\mathcal{O}(NT)$. Finally, in order to have an approximation of these measures at every time sample $t = 1, 2, ..., T$, we use the simplest form of a spline interpolation: the linear interpolation, i.e., data points are graphically connected by straight lines resulting in a polygon. This interpolation technique has linear complexity ($\mathcal{O}(NT)$) which means that the total complexity for computing these measures can be reduced from quadratic order to linear order in terms of the number of samples T. We should also note that the estimation of equation (6) is a convolution, which also has formally quadratic complexity $\mathcal{O}(T^2)$ but can be implemented in a fast way by using the classical Fast Fourier Transform (FFT) implementation reaching to a subquadratic complexity of order $\mathcal{O}(T \log_2(T))$.

[2] μ_1 and μ_2 are small scalars, typically $0.1 \sim 0.2$

[3] m_x and σ_x are the mean and the standard deviation of the signal $x(t)$ respectively

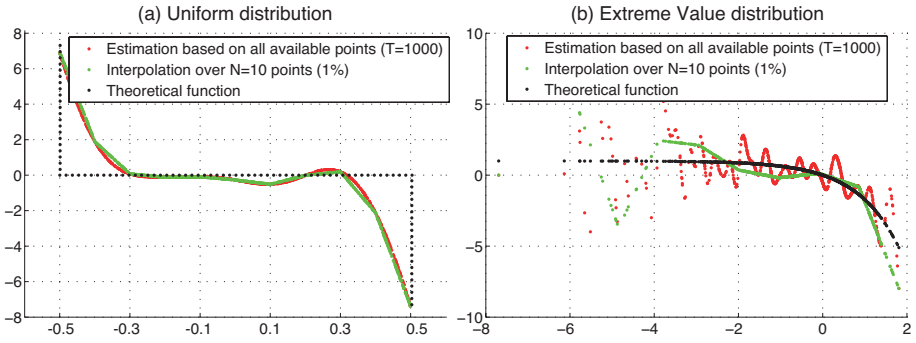

Fig. 2 Score function estimation for an uniformly distributed signal (left) and Extreme Value distribution (right). Theoretical score function is plotted as a reference.

To illustrate our approach, we show in Fig. 2 two examples of the score function estimation for: (a) an uniformly distributed signal whose pdf and score function are $p_y(u) = 1$ and $\Psi_y(u) = 0$ for $u \in [-0.5, 0.5]$, respectively; and (b) using the Extreme Value distribution, whose pdf and score functions are $p_y(u) = e^u e^{-e^u}$ and $\Psi_y(u) = 1 - e^u$, respectively. From this figure, it is clear that the estimation of the score function on few points ($N = 10$) can be used to interpolate it to all the samples with minimal error of approximation, in fact, in the case of the Extreme Value distribution the interpolated score function is more accurate than the one estimated over all the available points ($T = 1000$).

5 Simulations and Results

In the following sections, we evaluate the computational time and performance of our accelerated implementation of the Min-MI NL-BD algorithm and we compare it to the original implementation which estimates the perturbation signal and the score function in all the available samples. We consider several cases with different filter h lengths. We have generated i.i.d signals using the uniform distribution in the $[-1, 1]$ interval and we applied a normalization step in order to guarantee to have unit-variance signals. We have selected the nonlinearities such that they significantly distort the input signals in this range. For the inverse filter we consider symmetric filters defined by $w = [w(-M), ..., w(-1), w(0), w(1), ..., w(M)]$ with a total number of $2M + 1$ coefficients and we evaluate the performance of the algorithm as a function of the parameter M.

5.1 Computational Time Reduction Evaluation

In this section, we compare the computational time of the exact algorithm to the approximated version by considering an i.i.d. random sequence $s(t)$ as input, filtered by the non-minimum phase FIR filter

$$h = [-0.0082, 0, -0.1793, 0, 0.6579, 0, -0.1793, 0, -0.0082], \tag{7}$$

and then distorted with the nonlinear function $f(u) = 0.1u + \tanh(5u)$.

The algorithms were tested with sample sizes $T = 200, 400, 800, 1000$. For the approximated algorithm, different numbers of N equally spaced points were considered (for each T) and used to approximate equations (4) and (5). In all the cases the length of the filter w was arbitrarly set to 21 ($M = 10$), i.e. with the same length for the causal and anti-causal parts. In Table 1 the cases of N from 50% to 1% are detailed in percentage of diminishing computational time: $100 - (T_a/T_e) \times 100$ where T_a is the computational time of the approximate method and T_e is the computational time of the exact method. We can see how the computational time decreases using our approximate method, from 30% to 90% of the time needed for the exact method, approximately.

Table 1 Percentage of diminishing time for all cases of T for the interesting cases of N. All the percentage values are taken in reference to the exact case.

	Exact Case	$N = 50\%$	$N = 20\%$	$N = 10\%$	$N = 2\%$	$N = 1\%$
$T = 200$	100%	28.59%	49.27%	56.43%	62.23%	63.07%
$T = 400$	100%	33.77%	63.36%	72.45%	80.09%	81.09%
$T = 800$	100%	36.85%	70.82%	81.59%	90.5%	91.61%
$T = 1000$	100%	36.5%	72.05%	83.26%	92.45%	93.61%

Table 2 Obtained mean Signal to Noise Ratio (SNR) in dB for all cases of T for interesting cases of N.

	Exact Case	$N = 50\%$	$N = 20\%$	$N = 10\%$	$N = 2\%$	$N = 1\%$
$T = 200$	$11.84dB$	$11.70dB$	$11.48dB$	$11.62dB$	$8.43dB$	$5.31dB$
$T = 400$	$14.53dB$	$14.53dB$	$14.52dB$	$14.53dB$	$14.33dB$	$12.35dB$
$T = 800$	$17.48dB$	$17.48dB$	$17.48dB$	$17.52dB$	$17.86dB$	$18.38dB$
$T = 1000$	$18.96dB$	$18.95dB$	$18.96dB$	$18.99dB$	$19.16dB$	$19.21dB$

But this enormous reduction on the computational time should not affect the performance of the results. In order to evaluate the average performance, we computed the mean values of SNR. The SNR can be directly measured with the output signal to noise ratio $\sigma_s^2/\sigma_n^2 = \mathbb{E}[y^2(t)]/\mathbb{E}[(s(t)-y(t))^2]$, where σ_n^2 is the error power and σ_s^2 is the estimated signal power. In Figure 3 and Table **??**, the mean SNR versus the number of points N is shown for $T = 200, 400, 800, 1000$. Dashed line corresponds to the exact algorithm and solid line corresponds to the approximate algorithm run over N points, ranging from 50% to 1% of T.

SNR values for the exact case and for the approximate cases are almost the same for N at 50%, 20% and 10% of the T points considered in each case. For $T = 200$, if the number of points N is decreased to a 2% or less, the performance also decreases and the results are not good enough. On the other side, for larger datasets with $T = 800$ and $T = 1000$, the performance is not deteriorated even if the number of points is decreased to 1%. If we focus on the cases corresponding to N in 50%, 20% and 10% of the points in T, where we maintain SNR values, we observe that the reduction in computational time is within 30%- 80% approximately, but this reduction is higher for larger datasets ($T = 800$ and $T = 1000$) and N at 2% and 1% of the T points, where we achieve reductions over 90%.

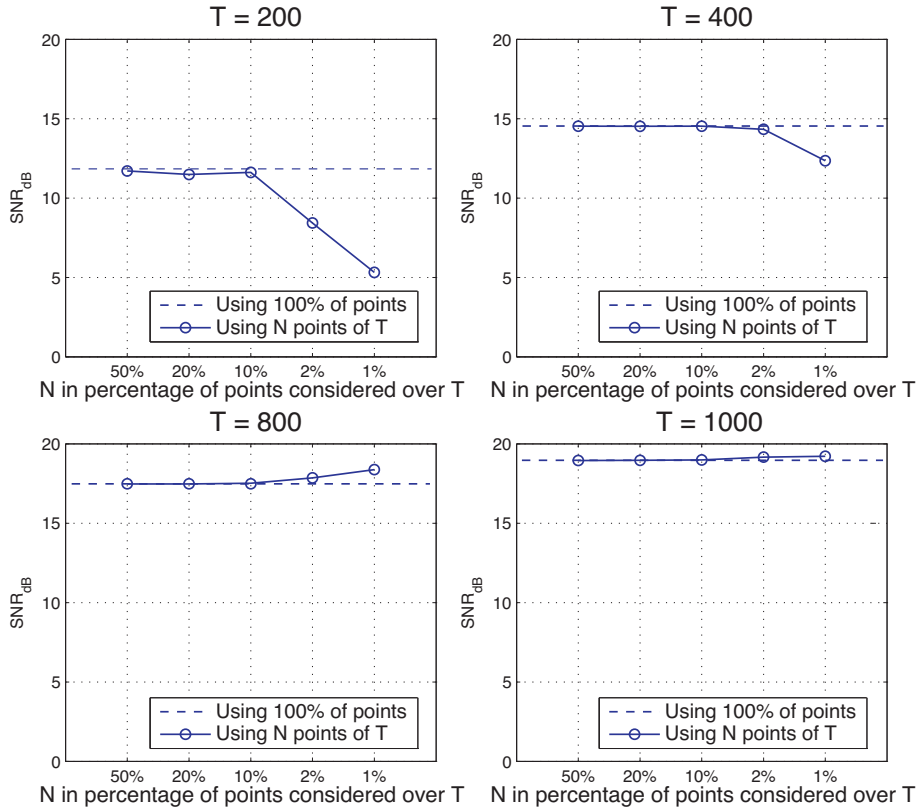

Fig. 3 Signal to Noise Ratio (SNR) versus percentage of points considered over T for the cases $T = 200$, $T = 400$, $T = 800$ and $T = 1000$

5.2 Performance Evaluation Under Various Conditions

Here we present a complete Monte Carlo analysis of the obtained performance by considering various filter h lengths and different nonlinearities. In particular, we compare the mean SNR obtained with our approximate algorithm by averaging the results of 50 realizations. We consider the following filters with 3, 6 and 9 coefficients: $h = [1, 0.2, -0.2]$ (case 1), $h = [1, -0.0394, 0.3419, -0.0523, 0.0361, -0.0104]$ (case 2) and $h = [1, -0.0919, 0.2282, -0.1274, 0.1408, -0.0189, 0.0173, -0.0072, 0.0038]$ (case 3); and two types of nonlinearities: $f(u) = 0.1u + \tanh(5u)$ and $f(u) = u^3$. In all the realizations we have considered a total number of $T = 1000$ samples but our algorithm computes explicitly the perturbation signal and the score function only on a grid with $N = 10$ points, which represents 1% of the available points, and linearly interpolates them to the rest o the points. In Fig. 4 we show the used filters h, their corresponding inverse filters w (theoretical and estimated by our algorithm) and the convolution $w * h$ using our estimated inverse filter w.

In Fig. 5 we show two different nonlinear functions f (scatter plot of z versus e), the corresponding estimated inverse function g (scatter plot of e versus x) and the composition of both functions, $g \circ f$ (scatter plot of z versus x) (see Fig. 1). We can clearly observe that the algorithm is able to inverse the nonlinear functions

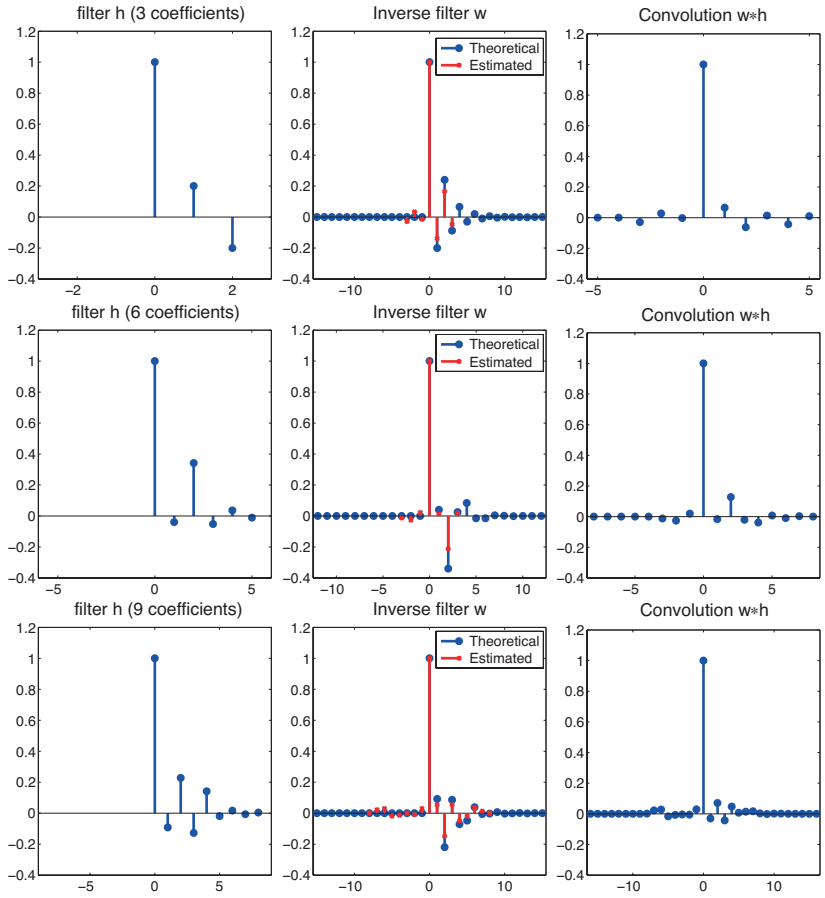

Fig. 4 Illustration of filters h, inverse filters w and convolution $w * h$ for 3 (case 1), 6 (case 2) and 9 (case 3) non-zero coefficients. The selected number of nonzero coefficients for the inverse filters are: $9(M = 4)$ for case 1, $9(M = 4)$ for case 2 and $19(M = 9)$ for case 3, which correspond to the optimal values. The obtained SNRs are: 19.06dB for case 1, 18.10dB for case 2 and 18.15dB for case 3.

and hence cancel the effect of the distortion introduced by f in the convolution system.

Finally, in Table 3, the obtained SNRs averaged over 50 realizations are shown together with their associated standard deviations. From Fig. 4 and Table 3 it is clear that our algorithm is able to identify correctly the Wiener system recovering the input signals correctly.

5.3 Performance Versus Length of the Inverse Filter

In order to illustrate which is the effect of a wrong selection of of the inverse filter w length, we performed some experiments using the same filters of of the

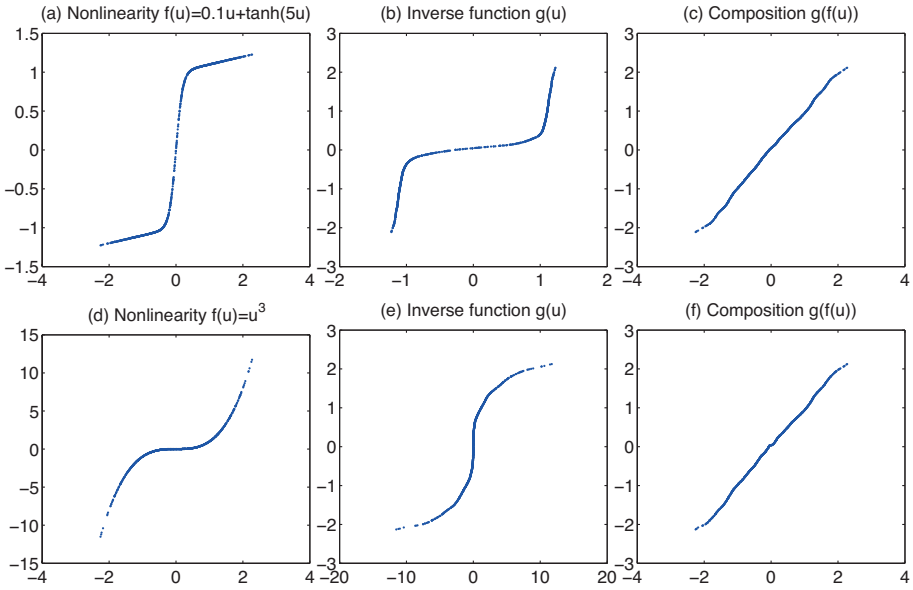

Fig. 5 Scatter plots of two different nonlinearities $f(u)$ (left), its estimated inverse functions $g(u)$ (center) and the composition of both nonlinear functions $g(f(u))$ (right), that theoretically must be a straight line.

Table 3 Monte Carlo (MC) analysis (50 realizations): sample points $T = 1000$, number of points in the interpolation grid $N = 10$ (1%) and the number of nonzero coefficients for the inverse filters are: $9(M = 4)$ for case 1, $9(M = 4)$ for case 2 and $19(M = 9)$ for case 3 which correspond to the optimal values.

	$f(u) = 0.1u + \tanh(5u)$	$f(u) = u^3$
Case 1 (3 coefficients)	19.98dB±1.22dB	19.03dB±0.95dB
Case 2 (6 coefficients)	18.48dB±1.27dB	17.88dB±1.07dB
Case 3 (9 coefficients)	18.63dB±1.10dB	18.14dB±0.92dB

previous subsection (3, 6 and 9 coefficients), whose impulse response can be seen in Fig. 4. Using an i.i.d. signal of $T = 1000$ samples and a nonlinear function $f(u) = 0.1u + \tanh(5u)$, we have calculated the SNR of the system for different values of M (from 0 to 20) and for each one of the proposed filters h. In Fig. 6 we can see the mean SNR obtained after 10 repetitions for each case. We observe that, using very few coefficients does not allow the algorithm to recover the input signal correctly. The optimal number of coefficients is found to be 11 ($M = 5$) for case 1 and case 2, and 17 ($M = 8$) for case 3. If we use more coefficients, the SNR degrades as these extra coefficients have low values but are not exactly zero as they should be. Since the convolutional filter h is unknown, we don't have any *a priori* information about how long the inverse filter w must be. Therefore, a practical solution can be to check the results for different filter lengths from 10 to 20 coefficients and adjust this parameter according to the results, taking into account that having less coefficients that the optimal value is worst that taking

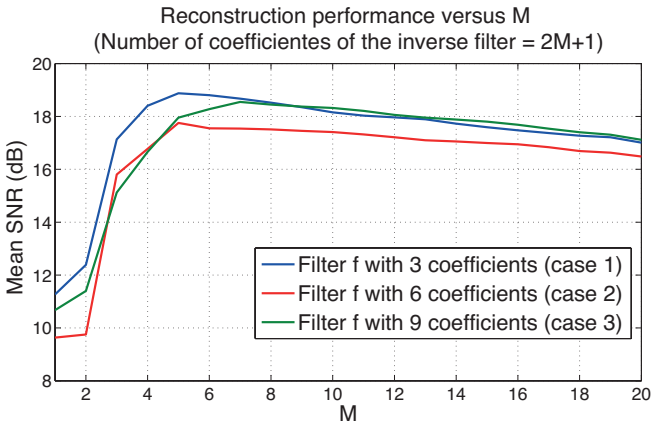

Fig. 6 Mean SNR values for three different convolution filters h versus the length $(2M+1)$ of the corresponding estimated inverse filter w.

more than the optimal, as can be seen in Fig. 6, where the slope of the curve decays slowly after the optimal value of M.

5.4 Higher Order Interpolation Analysis

Linear interpolation is the simplest interpolation method. One can expect better results if more a accurate interpolation method is used. Cubic spline interpolation was investigated as a possible alternative interpolation strategy. In this case, a series of unique cubic polynomials are fitted between each of the data points, with the stipulation that the curve obtained will be continuous and appear smooth (for a detailed explanation of properties and equations of spline curves see [15,16] and references therein). Simulation results (same conditions of section 5.1) shown similar SNR values as those obtained by linear interpolation (see Fig. 7b), but computational time required for estimating all the parameters of the spline is very high, yielding to a very slow inversion system. This aspect is shown in Fig. 7a, where we can see how cubic spline interpolation takes more time than the exact case, and only using 2% or less of the total available points we obtain a faster system. Cubic spline prove to be useful only if 1% or less of the total available points are considered, where linear method fails, giving a reasonable global SNR result with more than 50% of reduction of the computational time. But even in this case, better results and less computational time are obtained using a linear interpolation over 2% of the points. Consequently, linear interpolation is the best option considering SNR performance and computational time, allowing us to keep a less complex algorithm.

6 Discussion and Conclusions

In this paper we have proposed a simple approximation for fast nonlinear deconvolution and PNL source separation algorithms, based on the interpolation of the

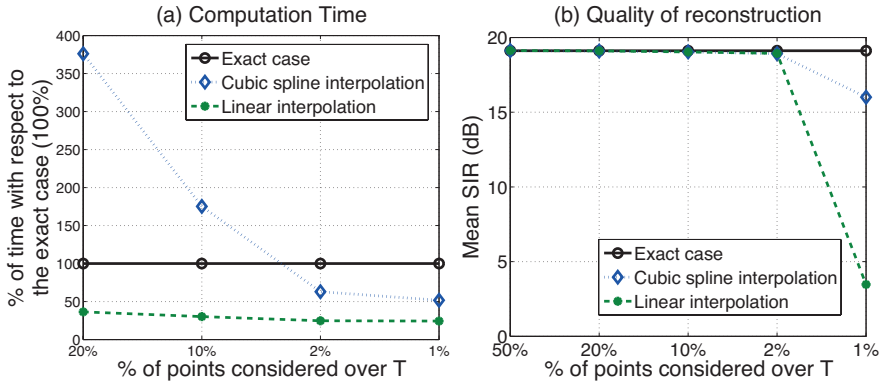

Fig. 7 On the left, computational time for the exact case (100%), cubic spline interpolation and linear interpolation. Only at 2% or less of the available points cubic spline interpolation is faster than the exact case, but in any case slower than the linear interpolation. On the right, mean SNR for the same three cases. At 1% of the available points linear interpolation fails but cubic spline interpolation is aible to obtain a good result.

two more complex equations of the original algorithms: the perturbation signal $\varepsilon(x(t))$ (eq. (1) for the deconvolution case and eq. (9) for the PNL case) and the score function $\widehat{\psi}_y(y)$ (4). This method allows us to reduce the original complexity of critical parts from order $\mathcal{O}(T^2)$ to a linear complexity of order $\mathcal{O}(T)$. Simulation results for the deconvolution case, based on synthetically generated datasets, clearly show that the performance is not affected by this approximation even if the number of points (N) used for calculations is much smaller than the original one (T). For small datasets, the approximation is degraded if we use less than 10% of the original points, while for large datasets we can diminish up to only 1% of the original points, giving more than 90% of reduction in computational time without any degradation in the results.

Real time algorithms for speech processing, spike signals deconvolution, microarray data analysis, etc. can be implemented using this method. In particular, it could be useful for the treatment of speech signals where the filter h models the linear distortion of the signal caused by the recording environment and the nonlinearity f can model typical nonlinear distortion associated to the microphone or the amplifier. However, in real world situations such as the case of speech signals, we usually handle non-i.i.d. signals. In such a situation, our algorithm will recover an i.i.d version of the original signal, i.e. the innovation process of the original speech signal. In fact, supposing that the speech signal can be modeled as an i.i.d. sequence filtered with an autoregressive (AR) filter, we can merge the AR filter and the channel filter h in a single combined filter. In other words, the inversion system will recover the inverse of the cascade, and the output of the inversion system will be the i.i.d. sequence at the input of the AR filter, i.e. the so-called innovation process. This result was verified in [9] by applying the Min-MI NL-BD algorithm to signals generated with an i.i.d. sequence (innovation process) as the input of an AR filter, and to real music signals, which are no longer i.i.d. signals.

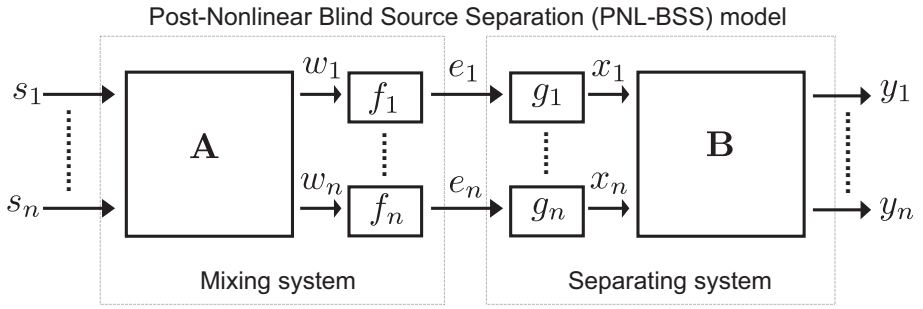

Fig. 8 The PNL-BSS model. The unknown nonlinear mixing system and the separating system are shown in the left and in the right, respectively.

Appendix A: Post-Nonlinear Blind Source Separation (PNL-BSS)

A.1 The PNL-BSS model

In [10,11], Taleb and Jutten have studied a realistic case of nonlinear mixtures, called post nonlinear (PNL) mixtures which are separable. As it is shown in Fig. 8, this two stage system consists of a linear mixing matrix $\mathbf{A}$, followed by component wise nonlinear distortions $f_1(.), ..., f_n(.)$. We assume that the input of the system consists of a set of n independent sources $s_1, ..., s_n$. The square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is assumed unknown and invertible and the nonlinear distortions (memoryless) $f_i : \mathbb{R} \to \mathbb{R}$ $(i = 1, 2, ..., n)$ are also assumed unknown and invertible. Again, we assume that all involved stochastic processes are wide sense stationary and ergodic so expectations are computed using time samples.

As in the blind deconvolution case, we will need to compute the score function of the outputs. Due to now we have several outputs, each one with a corresponding score function ψ_i, we also define a vector of score functions, the so-called *marginal score function* $\Psi = [\psi_1, \psi_2, ..., \psi_n]^T$.

A.2 The Min-MI Post-Nonlinear Blind Source Separation Algorithm (Min-MI PNL-BSS)

An algorithm for solving this problem was introduced in [12] where the separating system coefficients (matrix $\mathbf{B}$) and the compensation non-linearities $g_1, ..., g_n$ are chosen such that the mutual information among estimated sources $y_1, ..., y_n$ is minimized. The mixing observations are:

$$e_i(t) = f_i \left(\sum_{j=1}^{n} a_{ij} s_j(t) \right), \tag{8}$$

where $i = 1, 2, ..., n$, $s_j(t)$ $(j = 1, 2, ..., n)$ are the independent sources, $e_i(t)$ is the i-th observation, a_{ij} denotes the entries of the unknown mixing matrix $\mathbf{A}$ and f_i is the unknown nonlinear mapping on the component i.

We highlight that, since the statistical independence among sources is the main assumption, the separation structure is tuned so that the components of its output

become statistically independent. Similarly to the previous case (NL-BD) where we minimized MI between time simples of the (only) output, in the PNL-BSS we have to minimize the MI between estimated sources.

The gradient of the MI with respect to the parameters of the system can be written as follows (see [10] for a detailed derivation of these equations). Special *perturbation signals* $\varepsilon(x_i(t))$ are needed to compensate the nonlinear distortions. The vector composed by the *perturbation signals* is defined as follows:

$$\varepsilon(x_j(t)) = -\mathbb{E}\left[\left\{\sum_{l=1}^{n}\psi_l(s_l(t))b_{lj}\right\}K(y_j(t)-\tau_j) + K'(y_j(t)-\tau_j)\right], \qquad (9)$$

Additionally, the gradient of the MI with respect to matrix $\mathbf{B}$ becomes

$$\frac{\partial \mathbf{I}(\mathbf{y})}{\partial \mathbf{B}} = -\mathbb{E}\left[\mathbf{\Psi}_{\mathbf{y}}(\mathbf{y})\mathbf{x}^T\right] - \mathbf{B}^{-T}, \qquad (10)$$

where K must satisfy an additional condition of positivity in order to insure convergence of the gradient method.

As proposed in the NL-BD case, replacing expectations by their corresponding time average we can obtain practical equations analogous to equations (5) and (6) for the deconvolution problem. Additionally, we are able to reduce the complexity of the algorithm by using the same strategy as proposed for NL-BD, i.e. by computing eqn. 9 only at N points in a regular grid covering the ranges of the variables. In Algorithm 2, the steps of the Min-MI PNL-BSS Algorithm are shown.

Algorithm 2 `Min-MI PNL-BSS` **[10]**

INPUT: Maximum Number of Iterations *Niter* and output signal $\mathbf{e}(t)$
OUTPUT: Estimated input signal $\widehat{\mathbf{s}}(t)$

$i = 1$; $\mathbf{B} = \mathbf{I}$; Inicialization
$\mathbf{y} = \mathbf{Be}$
while $i \leq Niter$ and NOT convergence is reached **do**
  for $j = 1 : n$ **do**
    Estimate the *score function* ψ_j (eq. (4))
    Compute the *perturbation signal* $\varepsilon(x_j(t))$ (eq. (9))
    $x_j(t) \leftarrow x_j(t) + \mu_1\varepsilon(x_j(t))$; Nonlinear compensation [4]
    $x_j(t) \leftarrow (x_j(t) - m_{x_j})/\sigma_{x_j}$; Normalization
  end for
  Estimate $E = \frac{\partial \mathbf{I}(\mathbf{y})}{\partial \mathbf{B}}$ (eq. (10))
  $\mathbf{B} \leftarrow \mathbf{B} + \mu_2 E$; Separating matrix update [4]
  $\mathbf{y} = \mathbf{Bx}$; Current Estimation of source signals
  for $j = 1 : n$ **do**
    $y_j(t) \leftarrow (y_j(t) - m_{y_j})/\sigma_{y_j}$; Normalization [5]
    $\mathbf{B} \leftarrow \mathbf{\Lambda}^{-1}\mathbf{B}$; Normalization [6]
  end for
  $i = i + 1$;
end while
$\widehat{\mathbf{s}}(t) = \mathbf{y}(t)$;

[4] μ_1 and μ_2 are small scalars, typically $0.1 \sim 0.2$

[5] m_x and σ_x are the mean and the standard deviation of the signal $x(t)$ respectively

[6] $\mathbf{\Lambda}$ is a diagonal matrix with the standard deviations of each source in its main diagonal, i.e. $\mathbf{\Lambda}_{i,i} = \hat{\sigma}_{\hat{s}_i}$

Acknowledgements We would like to thank the anonymous Reviewers for their useful and insightful comments which helped us to considerably improve this work.

References

1. Comon, P., Jutten, C.: Handbook of Blind Source Separation: Independent Component Analysis and Applications. Academic Press (2010)
2. Härdle, W.: Smoothing techniques: with implementation in S. Springer (1991)
3. Hunter, I.W., Korenberg, M.J.: The identification of nonlinear biological systems: Wiener and hammerstein cascade models. Biological Cybernetics **55**(2), 135–144 (1986)
4. Parzen, E.: On estimation of a probability density function and mode. The annals of mathematical statistics pp. 1065–1076 (1962)
5. S Haykin, S.: Blind deconvolution, vol. II. Wiley (2000)
6. S Haykin, S.: Blind source separation, vol. I. Wiley (2000)
7. Silverman, B.W.: Density estimation for statistics and data analysis. Chapman and Hall, London (1986)
8. Solé-Casals, J., Caiafa, C.: A Simple Approximation for Fast Nonlinear Deconvolution, Advances in Nonlinear Speech Processing - 5th International Conference on Nonlinear Speech Processing, NOLISP 2011, Las Palmas de Gran Canaria, Spain, November 7-9, 2011, Proceedings: Lecture Notes in Computer Science 7015, vol. 7015, pp. 55–62. Springer Berlin / Heidelberg (2011)
9. Solé-Casals, J., Jutten, C., Taleb, A.: Parametric approach to blind deconvolution of nonlinear channels. Neurocomputing **48**(1-4), 339 – 355 (2002)
10. Taleb, A., Jutten, C.: Batch algorithm for source separation in postnonlinear mixtures. In: First Int. Workshop on Independent Component Analsysis and Signal Separation (ICA 1999), pp. 155–160 (1999)
11. Taleb, A., Jutten, C.: Source separation in post-nonlinear mixtures. Signal Processing, IEEE Transactions on **47**(10), 2807 –2820 (1999)
12. Taleb, A., Solé-Casals, J., Jutten, C.: Quasi-nonparametric blind inversion of wiener systems. Signal Processing, IEEE Transactions on **49**(5), 917 –924 (2001)
13. Baccarelli, E., Galli, S.: A New Approach Based on Soft Statistics to the Nonlinear Blind-Deconvolution of Unknown Data Channels. Signal Processing, IEEE Transactions on **49**(7), 1481 –1491 (2001)
14. Fukunaga, S., Fujimoto, K.: Nonlinear blind deconvolution based on a state-space model., Proceedings of the 45th IEEE Conference on Decision & Control, San Diego, CA, USA, December 13-15, (2007).
15. Härdle, W.: Smoothing Techniques with implementation. Springer, Belgium (1990)
16. Prenter, P.M.: Splines and Variational Methods. Dover, New York (2008)