A Simple Approximation for Fast Nonlinear Deconvolution

Jordi Solé-Casals[1] and Cesar F. Caiafa[2,3]

[1] Digital Technologies Group
University of Vic, Sagrada Famlia 7, 08500, Vic, Spain
[2] Instituto Argentino de Radioastronomía (CCT La Plata, CONICET)
C.C.5, (1894) Villa Elisa, Buenos Aires, Argentina
[3] Facultad de Ingeniería - UBA
Av. Paseo Colón 850. 4to. piso, Ala sur (1063) Capital Federal, Argentina
jordi.sole@uvic.cat, ccaiafa@iar.unlp.edu.ar

Abstract. When dealing with nonlinear blind deconvolution, complex mathematical estimations must be done giving as a result very slow algorithms. This is the case, for example, in speech processing or in microarray data analysis. In this paper we propose a simple method to reduce computational time for the inversion of Wiener systems by using a linear approximation in a minimum-mutual information algorithm. Experimental 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.

1 Introduction

Nonlinear blind deconvolution deals with a particular class of nonlinear systems. This is composed by a linear subsystem (filter) and a memoryless nonlinear distortion (function) (Figure 1, left). This class of nonlinear systems, also known as Wiener systems, is not only another nice and mathematically attracting model, but also a model found in various areas, such as biology: study of the visual system [1], relation between the muscle length and tension [2]; computer vision [3]; industry: description of a distillation plant [4]; sociology and psychology, see also [5] and the references therein. This model is popular because it has a convenient block representation, a transparent relationship to linear systems, and it is easier to implement compared to heavy-duty nonlinear models (such as neural networks and Volterra models).

One of the areas where Wiener systems are applied is audio and speech processing. For example, it can be used in order to estimate and compensate microphone distortion in speaker recognition/identification scenarios [6]. Or it can be used to determinate the innovation process of a speech signal, supposing that the signal can be modeled as an independent and identically distributed (i.i.d.) sequence filtered with an AR (autoregressive) filter. In this case, we can merge the AR part and the channel filter into a single filter. Then, the inversion system will recover the inverse of the cascade, and the output of the inversion system

C.M. Travieso-González, J.B. Alonso-Hernández (Eds.): NOLISP 2011, LNAI 7015, pp. 55–62, 2011.

will be the i.i.d. sequence at the input of the AR filter, i.e. the so-called innovation process [7]. As this kind of algorithms can be applied in any situation where we have a signal collected by a sensor, which in turns affects the signal through a filtering effect and a possible distortion, it can also be applied to the case of fluorescence signals of microarray experiments, where Wiener deconvolution filtering algorithm can improve either spot segmentation or gene profiling [8,9].

Due to the nonlinear part must be compensated, the gradient equations that come out in the optimization procedure are much more complex than in the pure linear case. When a nonlinear distortion is considered, the gradient of Mutual Information (MI) has an expectation form where the score function appears jointly with other functions that depend on the filter coefficients [10]. These 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. The nonlinear deconvolution model and its equations are described in Section 2; in Section 3, the Minimum-Mutual Information (Min-MI) algorithm is presented; in Section 4, we propose simple techniques to reduce the computation complexity dramatically; in Section 5, a set of experimental results are presented; and, in Section 6, the main conclusions of our work are outlined.

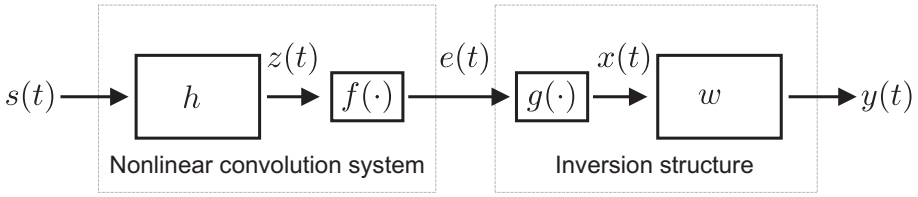

Fig. 1. The unknown nonlinear convolution system (left) and the used inversion structure

2 Model, Assumptions and Notation

Following the same notation used in [10], 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 (Figure 1, right). The nonlinear part g is concerned by 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 [5].

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.

3 The Min-MI Nonlinear Blind Deconvolution Algorithm

It is known that the inverse system, composed by the function g and the linear filter w, can be estimated by minimizing the mutual information (MI) of the output $\mathcal{Y} = \{y(t)\}$.

In [10], the Min-MI NLBD (Minimum-Mutual Information Non-linear 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 non-linear distortion. The *perturbation signal* is defined as follows:

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

where, the functions $\delta(x)$ and $\delta'(x)$ are chosen as the *ramp* function and its derivative respectively, i.e. $\delta(x) = xH(x)$ and $\delta'(x) = H(x)$ 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 equations (1) and (2) the *score function* $\psi_y(y)$ is used which is not available a priori and should be estimated in some manner. Here, we use a nonparametric approach based on Parzen windows [11,12,13]. 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 to 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 [13].

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 $y(t)$ and $x(t)$ to be ergodic processes, we are able to compute expectations by averaging over time samples. Therefore, we see that equations (1) and (2) can be estimated, respectively, by:

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

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

In order to reduce the quadratic complexity to linear complexity we propose to compute equations (4) and (5) only at N points in a regular grid covering the ranges of the variables 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 $\mathcal{O}(T\log_2(T))$.

5 Experiments and Results

In order to compare the exact algorithm with the approximated version we consider an i.i.d. random sequence $s(t)$ as input, filtered by a non-minimum phase FIR filter $h = [-0.082, 0, -0.1793, 0, 0.6579, 0, -0.1793, 0, -0.082]$ and then distorted with $f(u) = 0.1u + \tanh(5u)$.

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 set to 21 with the same length for the causal and anti-causal parts. Finally, in order to evaluate the average performance, we computed the mean values of SNR and the Computation Time over a set of 10 repetitions for each experiment. 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 2, 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.

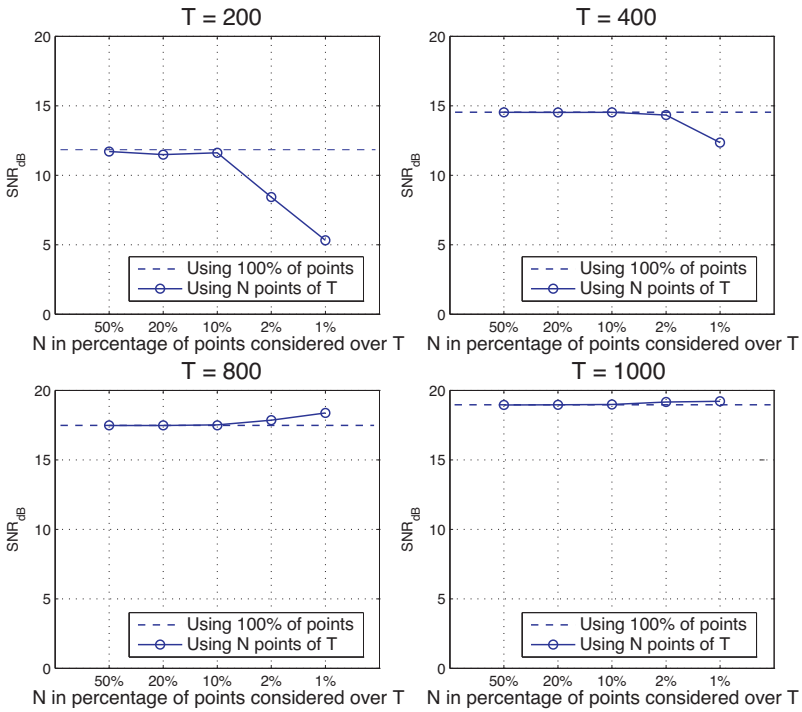

Fig. 2. Signal to Noise Ratio (SNR) versus percentage of points considered over T for the cases $T = 200$, $T = 400$, $T = 800$ and $T = 1000$

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 $T = 200$ and $T = 400$, if the number of points N is decreased to a 2% or less, the performance also decrease and 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 2% or 1%.

Concerning the Computation Time, we show in Figure 3 its evolutions for different values of T and N. Also, in Table 1 the cases of N from 50% to 1% are detailed in percentage of diminishing time need: $100-(T_a/T_e)\times 100$ 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 Computation Time decreases using our approximate method, from 30% to 90% of the time needed for the exact method, approximately. If we take into account previous results in SNR, and we consider as interesting cases those corresponding to N in 50%, 20% and 10% of the points in T, where we maintain SNR values, we observe a reduction in computational time is within 33%- 80% (except for the case $T = 200$), 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%.

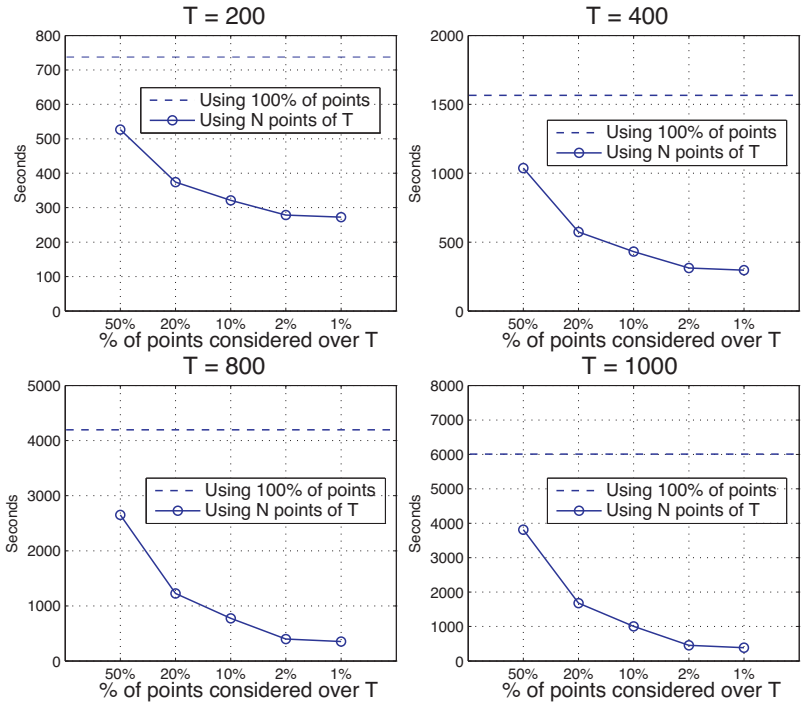

Fig. 3. Computation Times versus percentage of points considered over T for the cases $T = 200, 400, 800, 1000$

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%

6 Conclusions

In this paper we have proposed a simple approximation for fast nonlinear deconvolution algorithm based on linear interpolation of the two more complex equations of the original algorithm: the *perturbation signal* $\varepsilon(x(t))$ (1) and the *score function* ψ_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 $\mathcal{O}(T)$. Experimental results clearly show that 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 or microarray data analysis can be implemented using this method.

Acknowledgments. This work has been in part supported by the MINCYT-MICINN Research Program 2010-2011 (Argentina-Spain): *Desarrollo de Herramientas de Procesado de Señales para el Análisis de Datos Bioinformáticos* (Ref. AR2009-0010) and by the University of Vic under the grant R0904.

References

1. Brinker, A.C.: A comparison of results from parameter estimations of impulse responses of the transient visual system. Biological Cybernetics 61, 139–151 (1989)
2. Hunter, I.W.: Frog muscle fiber dynamic stiffness determined using nonlinear system identification techniques. Biophys. J. 49, 81a (1985)
3. Ma, W., Lim, H., Sznaier, M., Camps, O.: Risk adjusted identification of wiener systems. In: 45th IEEE Conference on Decision and Control 2006, pp. 2512–2517 (2006)
4. Bars, R., Bèzi, I., Pilipar, B., Ojhelyi, B.: Nonlinear and long range control of a distillation pilont plant. In: Identification and Syst. Parameter Estimation, Preprints 9th IFAC/FORS Symp., Budapest, pp. 848–853 (1990)
5. Hunter, I.W., Korenberg, M.J.: The identification of nonlinear biological systems: Wiener and hammerstein cascade models. Biological Cybernetics 55, 135–144 (1986)

6. Solé-Casals, J., Faundez-Zanuy, M.: Application of the mutual information minimization to speaker recognition/identification improvement. Neurocomput. 69, 1467–1474 (2006)
7. Solé-Casals, J., Jutten, C., Taleb, A.: Parametric approach to blind deconvolution of nonlinear channels. Neurocomputing 48, 339–355 (2002)
8. Daskalakis, A., Argyropoulos, C., Glotsos, D., Kostopoulos, S., Athanasiadis, E., Cavouras, D.: Wiener-based deconvolution methods for improving the accuracy of spot segmentation in microarray images. In: 5th European Symposium on Biomedical Engineering. University of Patras Cultural and Conference Center Patras, Patras, Greece (2006)
9. Lu, P., Nakorchevskiy, A., Marcotte, E.M.: Expression deconvolution: A reinterpretation of dna microarray data reveals dynamic changes in cell populations. Proceedings of the National Academy of Sciences 100, 10370–10375 (2003)
10. Taleb, A., Solé-Casals, J., Jutten, C.: Quasi-nonparametric blind inversion of wiener systems. IEEE Transactions on Signal Processing 49, 917–924 (2001)
11. Parzen, E.: On estimation of a probability density function and mode. The Annals of Mathematical Statistics, 1065–1076 (1962)
12. Härdle, W.: Smoothing techniques: with implementation in S. Springer, Heidelberg (1991)
13. Silverman, B.W.: Density estimation for statistics and data analysis. Chapman and Hall, London (1986)