Analysis of source separation algorithms in industrial acoustic environments

Clevis Lozano[*][†], Andrés Gómez[*], Alfonso Chacón-Rodríguez[*], Fernando Merchán[†], and Pedro Julian[‡]

[*] DCILab, Escuela de Ingeniería Electrónica, Tecnológico de Costa Rica
Email: alchacon@tec.ac.cr
[†]Facultad de Ingeniería Eléctrica, Universidad Tecnológica de Panamá
Email: anaclevis.lozano@utp.ac.pa, fernando.merchan@utp.ac.pa
[‡]Instituto de Investigaciones en Ingeniería Eléctrica, IIIE (UNS-CONICET)
Departamento de Ingeniería Eléctrica y de Computadoras
Universidad Nacional del Sur, Bahía Blanca, Argentina
Email: pjulian@uns.edu.ar

***Abstract*—This paper shows the results from the computation cost evaluation of three blind source separation algorithms. The algorithms tested were: FastICA, Adaptive Algorithm Based on Natural Gradient, and Adaptive EASI Based on Relative Gradient. The algorithms were chosen for their relative simplicity, and taking into account their hardware implementation feasibility, either on a FPGA or an ASIC, as part of a system for acoustic localization of mobile agents in industrial environments.**

***Index Terms*—Blind Source Separation (BSS), FastICA, Adaptive Algorithm Based on Natural Gradient, Adaptive EASI Based on Relative Gradient, FPGA, acoustic localization.**

I. Introduction

It has been shown that is possible to estimate the distance at which a sound source is located by either processing the acoustic signals obtained by a set of microphones in a particular environment [1], or by using the time and phase difference between the signals received by a network of sensors [2]. However, before being able to spatially locate an object by the sound it emits, it is necessary to extract that same sound from the acoustic signal in which other sounds (or even echoes and reflections of the target signal) come bundled. This means that Blind Source Separation (BSS) has to be performed on the raw sound data, in order to extract the signal in question, before proceeding to the localization process. However, BSS usually means complex signal processing algorithms, which requires a lot of matricial and statistical arithmetic, representing high computational loads [3].

In industrial environments, where different sounds are emitted at the same time, and where physical obstacles (some of them mobile) unavoidably produce multi-path distortion and reflections, audio signal analysis becomes a complex problem, even more complex if you want to apply the techniques of BSS [4]. These techniques are based on some probabilistic assumptions, especially related to the statistical independence among the signals to be extracted [4], [5]. Considering the feasibility of having BSS in an integrated circuit, a sensible step is to analyze and evaluate the required arithmetic operations by the most common algorithms, in order to rank them in terms of implementation costs (mainly: a reasonable power consumption on a relatively small silicon area), but keeping in mind identification robustness. With this objective, Section II presents the algorithmic description and preliminary computational cost analysis of three BSS algorithms, selected because of their relatively low complexity, implemented using National Instruments' LabView programming framework. Section III discusses the evaluation of the performance of each algorithm, using signals captured from various wood processing machines at an industrial workshop. Section IV presents conclusions and future work.

II. Description and computational cost of the chosen algorithms

Three BSS algorithms were chosen because of their relative simplicity and their high potential of implementation in an integrated circuit of low-cost in acoustic applications. For the extraction of a sound source of interest in a mixture, statistical independence is assumed among the sources for the signals captured by the microphones [5]. Mixtures can be either instantaneous or convolutive, and may be analyzed either in the time domain or the frequency one. In the case reported here, samples are instantaneous and analyzed in the time domain, following examples in the literature of BSS applied in real environments (see [6]). The following conventions are defined in the implementation of the algorithms:

- *P*: Number of microphones.
- *N*: Sample size of data.
- *J*: Number of sources.
- A: Mixing matrix, $P \times J$.
- B: Final separation matrix, $J \times P$.
- $\mathbf{s} = [s_1, s_2, \ldots, s_J]^T$: Sound sources from the environment.
- $\mathbf{v} = [v_1, v_2, \ldots, v_P]^T$: Observed signals in the microphones.
- $\mathbf{x} = [x_1, x_2, \ldots, x_P]^T$: Observed signals after pre-processing.
- $\mathbf{y} = [y_1, y_2, \ldots, y_J]^T$: Estimated original sources.

A. FastICA Algorithm

FastICA is one of the most known algorithms for BSS, and it is used especially because of its robustness and speed of convergence [4], [7]. This algorithm operates with data blocks, which entails having memory enough to store the blocks of N data at each stage of processing. FastICA consists of three phases, as shown in figure 1.

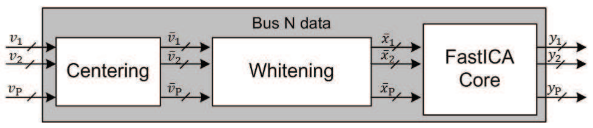

Fig. 1. Phases of the FastICA algorithm. The centering and the whitening processes are indeed pre-processing operations that optimize the subsequent ICA core processing.

The centering is the estimated data average of the input array $\mathbf{v}$ and each of its rows $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_P$, as follows:

$$\bar{\mathbf{v}}_i = \mathbf{v}_i - \frac{1}{N}\sum_{j=1}^{N} v_{ij} \tag{1}$$

The basic arithmetic operations needed to center a row vector are listed in table I. In hardware, this block should be replicated *P* times in order to calculate the average of all in parallel rows; an alternative is to insert sequencing logic to use the same resources in *P* different times (pipelining thus the process).

TABLE I
BASIC ARITHMETIC OPERATIONS NEEDED TO CENTER A VECTOR

Operations	**Estimated number**	**Total**
Sums	N	PN
Multiplications	1	P
Subtractions	N	PN
Total	$2N+1$	$2PN+P$

The whitening process focuses on the estimation of the covariance matrix from the data block. Table II shows the total of arithmetic operations needed.

TABLE II
NUMBER OF ARITHMETIC OPERATIONS FOR THE WHITENING PROCESS, WHERE k IS THE NUMBER OF ITERATIONS, A PARAMETER THAT DETERMINES THE CONVERGENCE OF THE ALGORITHM.

Operations	**Total**
Sums	$\frac{1}{2}(3N-1)P^2-\frac{1}{2}(N+1)P+k((2P-1)\sum_{j=1}^{P-1} j+P^3-P)$
Multiplications	$\frac{3}{2}(N+1)P^2+\frac{1}{2}(N+3)P+k(P^3+3P^2+3P\sum_{j=1}^{P-1} j)$
Subtractions	$kP\sum_{j=1}^{P-1} j$
Divisions	$kP+1$
Square Roots	$kP+P$
Total	$(3N+1)P^2+2P+1+k(2P^3+3P^2+((6P-1)\sum_{j=1}^{P-1} j)+P)$

The FastICA technique has two core versions: one based on kurtosis and one based on negentropy. In this case, an ICA core version based on kurtosis was selected, considering that its speed of convergence is 10 to 100 times faster compared with others algorithms [8]. Table III summarizes the number of arithmetic operations involved in the FastICA algorithm.

TABLE III
NUMBER OF ARITHMETIC OPERATIONS NEEDED FOR THE FASTICA CORE ALGORITHM, BASED ON KURTOSIS. EVALUATED ON A *N*=20,000 SAMPLES INPUT DATA VECTOR.

Operations	**Total**
Sums	$k\left[(J-1)P^2+2NJP-J(N+2)+1\right]+(J-1)^2\frac{P^2+P}{2}$
Multiplications	$k\left[(J-1)P^2+(6J+2NJ-2)P+2NJ\right]+J(J-1)(\frac{P^2+P}{2})$
Subtractions	$kP(2J-1)$
Divisions	$k(2J-1)$
Square Roots	$k(2J-1)$
Total	$k\left[2(J-1)P^2+(4NJ+8J-3)P+NJ+2J-1\right]+(2J-1)(J-1)(\frac{P^2+P}{2})$

B. Adaptive Algorithm Based on Natural Gradient.

This algorithm operates from sample to sample in each of the mixtures, which represents an advantage by not requiring large amounts of memory to store input data blocks. In addition, being adaptive, this algorithm minimizes the statistical dependence between the sources, estimated on the basis of the mutual information parameter; however, it has the disadvantage that it may require a significant time to converge [9]. The algorithm only requires centering of the input data before applying the samples to the central core; a low-pass filter (Eq. 2) is used for this operation, where α is the gain of the filter. This calculation performs two multiplications, an addition and a subtraction.

$$m_i(t) = \alpha x_i(t) + (1-\alpha)m_i(t-1) \tag{2}$$

The central core of the algorithm executes the following steps:

$$y(t) = B(t-1)x(t) \tag{3}$$
$$B(t) = B(t-1) - \mu[f(y)y^T - I]B(t-1) \tag{4}$$

where, $\mathbf{B}$ is the matrix of size $J \times P$ and $f(y)$ is the activation function, defined as

$$f(y) = \frac{3}{4}y^{11} + \frac{25}{4}y^9 - \frac{14}{3}y^7 - \frac{47}{4}y^5 + \frac{29}{4}y^3$$

Table IV shows the total of arithmetic operations from the previous two steps.

TABLE IV
NUMBER OF ARITHMETIC OPERATIONS NEEDED FOR THE CORE OF THE ADAPTIVE ALGORITHM BASED ON NATURAL GRADIENT.

Operations	**Estimated number**
Sums	PJ^2-J
Multiplications	$(1+P)J^2+(2P)J$
Subtractions	$(1+P)J$
Total	$(2P+1)J^2+3PJ$

C. Adaptive Algorithm EASI Based on Relative Gradient.

This adaptive algorithm was proposed by J. Cardoso [11]. This algorithm integrates whitening within the processing, so data are not correlated and have unit variance. Like the former algorithm, it requires only two stages, for a total of two multiplications, a sum and a subtraction (for the centering of the data). The number of operations for the EASI algorithm is shown in table V, where a great variation in the number of operations needed is not noticed with respect to the natural gradient algorithm.

TABLE V
NUMBER OF ARITHMETIC OPERATIONS FOR THE CORE OF THE EASI ALGORITHM.

Operations	**Estimated number**
Sums	$(1+P)J^2 - J$
Multiplications	$(2+P)J^2 + (2P)J$
Subtractions	$J^2 + (1+P)J$
Total	$2(P+2)J^2 + (3P)J$

III. EVALUATION OF THE ALGORITHMS'S EFFECTIVENESS

For the analysis and validation of the three implemented algorithms, several sets of data were prepared. The first set included three sounds coming from an industrial environment: a truck's horn, a tractor engine in steady-state and a chain-saw that accelerates and decelerates (all super Gaussians). These sounds were downloaded from the internet and were used to check the algorithms' performance on LabView. The second data set included real data from several woodworking machines, recorded at an industrial wood workshop. Signals were acquired with a conventional LifeChat LX-6000 microphone, located at 1 m distance from three different machines, with each machine recorded one at a time. The recorded machines were: an edger saw, a wood planer and a straight saw (see figure 2). Data were sampled at 44.1 KHz. This preliminary data allowed for the determination of the higher energy components of each machine's spectrogram, mainly centered for all between 100 and 1000 Hz. A third data set included two reference signals (a sine tone and a square waveform with duty cycle of 10%), that were synthesized at two different frequencies: 600 Hz and 1 kHz. These signals were chosen after evaluating their non-Gaussianity to make sure they are extractable (one of the requirements of the tested BSS algorithms). The same test was performed on the recorded sources and the results are indicated in table VI. A fourth data set was obtained recordings the three machines simultaneously with a network of omnidirectional microphones (this last data set is still under analysis as of the writing of this paper).

To estimate the quality of the separation, a metric based on the signal-to-interference ratio (S/I or SIR) was used (see [10]). For the validation of the three algorithms's effectiveness, four linear and instantaneous mixtures $\mathbf{y} = [y_1, y_2, y_3, y_4]^T$ were created as follows.

$$\mathbf{v} = \mathbf{A}\mathbf{s} \tag{5}$$

where $\mathbf{s} = [s_1, s_2, s_3, s_4]^T$ are three real sources and one reference signal (a square waveform with a duty cycle of 10%), all with a size of 20,000 samples.

$$\mathbf{A} = \begin{bmatrix} 0.173137 & 0.206178 & 0.808512 & 0.008549 \\ 0.477643 & -0.681523 & -0.995208 & 0.325804 \\ -0.099216 & -0.830972 & -0.894772 & 0.130999 \\ -0.549044 & 0.788603 & -0.013563 & 0.298793 \end{bmatrix}$$

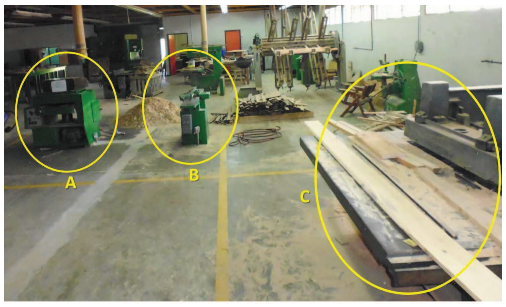

Fig. 2. Picture of the data collection setting: a small wood workshop. Encircled are the three machines recorded: (A) A wood planer, (B) An edger saw, (C) A straight saw.

TABLE VI
ESTIMATED KURTOSIS FOR EACH SOURCE. EACH SOURCE'S DATA BLOCK IS 20000 SAMPLES LONG.

Source	**Kurtosis**	**Classification**
Edger saw	0.0145959	Super Gaussian
Wood planer	−1.09793	Sub Gaussian
Straight saw	0.190877	Super Gaussian
Square waveform	38.989	Sub Gaussian
Sine tone	−1.499	Sub Gaussian

The analysis was based on the number of iterations executed and the quality of the separation for each algorithm, as shown in Fig. 3. These results show that all signals were extracted. The FastICA algorithm converged at the fourth iteration. More iterations did not remarkably increase separation quality. By three iterations, all estimations had exceeded 90% of the expected value. In Fig. 3, we can observe that two signals, the square waveform (in blue) and the wood planer (in green), reached a very high SIR while the other two (edger saw and straight saw) reached around 12 dB. We have noticed that these two signals have a kurtosis near zero.

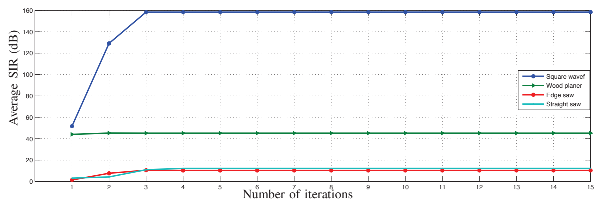

Fig. 3. Number of iterations needed for the FastICA's core algorithm.

To evaluate the two adaptive algorithms, the same blocks of data, the function of non-linearity $f(y) = tanh(y)$ and a learning rate of 0.0001 were used. Table VII shows the results. The first observed effect is that the quality of the separation was very low (3 dB aprox.). Also, neither of the adaptive algorithms was able to separate the four sources, although the number of iterations was much greater compared with FastICA algorithm. The reason for these results could be the presence of two signals with kurtosis near zero. This means that their distributions are not very different from the Gaussian distribution. Results showed no big difference on the behavior of the adaptive algorithms, in terms of the number of iterations taken to converge.

TABLE VII
SEPARATION QUALITY RESULTS FOR THE NATURAL GRADIENT AND EASI ALGORITHMS ($\mu = 0.0001$).

Algorithm	SIR (dB) Natural Gradient	SIR(dB) EASI
Edger saw	3.27346	3.53114
Wood planer	3.04441	0.721997
Straight saw	2.29507	2.96695
Unidentified	-3.03844	-1.41929
Iterations	30	30

Another test was carried out using a different version of FastICA [5]. This algorithm was implemented in Matlab. According to Table VIII, all four signals were extracted. The lowest two SIR correspond to the edger saw and the straight saw as in the first evaluation. This shows that the FastICA's Labview implementation performs as well as Matlab's one.

TABLE VIII
RESULTS OF THE EVALUATION OF THE QUALITY OF THE SEPARATION FOR FASTICA IMPLEMENTED IN MATLAB

Algorithm	SIR (dB) FastICA in Matlab
Edger saw	13.9533
Wood planer	29.6524
Straight saw	12.6654
Square waveform	26.6341

From the total number of basic arithmetic operations of the each algorithm, with *P*= 4 and *J*= 4, table IX was built. Extra complexity of divisions and square roots are taken into account[1]. Adaptive algorithms are simpler but sequential, leading to greater converging times, but with much less memory requirements.

TABLE IX
NUMBER OF BASIC ARITHMETIC OPERATIONS OF EACH ALGORITHM (*$\mu = 0.001$) AND k=8.

Algorithm	FastICA	Grad. Natural*	EASI*
Sums+Subtractions	2,741,476	3,930,000	4,710,000
Multiplications	3,862,530	5,640,000	5,580,000
Divisions	89 (2,536)	0	0
Square Roots	92	0	0
Total	6,606,654	9,570,000	10,290,000

[1]For FastICA, division using a SRT division algorithm is supposed, (number of operations given inside the parenthesis are the number of subtractions and multiplications for this implementation: two multiplications and a subtraction for each iteration). For the square root, a special computing block is assumed.

IV. CONCLUSIONS

Three BSS algorithms have been analyzed because of their relative simplicity and their high potential for implementation in a low cost integrated circuit. Algorithms showed favorable results in initial evaluations with signals downloaded from Internet but, when tested with sounds from a real industrial setting, results were no longer as good for the adaptive algorithms. Also, an approximate computational cost required by each of them has been estimated, in terms of the number of simple arithmetic operations executed. Results point to FastICA as the best candidate for hardware implementation, if an extra stage of deflationary orthogonalization and negentropy is added. This analysis is extendable to a frequency version with convolutional mixtures. An evaluation with the fourth set of data described above and the algorithm chosen is still being worked on, before moving on to a FPGA prototype.

ACKNOWLEDGMENTS

Authors thank Dr. Alexander Berrocal and his collaborators at the wood workshop in Escuela de Ingeniería Forestal at Tecnológico de Costa Rica. Thanks also to Reinaldo Castro, Ronald Soto, Fabian Coto and Asdrúal Rojas of the Escuela de Ingeniería Electrónica at Tecnológico de Costa Rica, for their help during data collection. Thanks to National Instruments (NI) for their advice. Part of this project is sponsored by PICT Bicentenario 2010 code 2657, Agencia Nacional de Promoción Científica y Técnica of Argentina.

REFERENCES

[1] J. Lanslots, F. Deblawe, K. Janssens, *"Selecting Sound Source Localization Techniques for Industrial Applications"*, The Noise and Vibration Control Magazine, pp. 6-9, Junio 2010.
[2] A. Chacon., F. Martin-Pirchio, S. Sanudo, and P. Julian, "A low-power integrated circuit for interaural time delay estimation without delay lines", IEEE Trans. Circuits Systems II, Express Briefs, 56, (7), 2009.
[3] A. Van Schaik, S. Shamma, *"A Neuromorphic Sound Localizer for a Smart MEMS System"*, Analog Integrated Circuits and Signal Processing, 39, pp. IV-864-867, 2004.
[4] P. Common and C. Jutten. Handbook of Blind Source Separation, chapter 1: Introduction, pages 1–22. Elsevier Ltd, 1 edition, 2010.
[5] A. Hyvarinen, J. Karhunen, E. Oja, "Independent component analysis", Wiley & Sons, 2001.
[6] M. Stanacevic and G. Cauwenberghs. Gradient flow adaptive beamforming and signal separation in a miniature microphone array. In Proc. IEEE Int. Conf. Acoustic Speech and Signal Processing (ICASSP'2002), Orlando FL, pages 13–17, 2002.
[7] C. Alvarez, J. Monzón, "Aplicaciones de ICA con conceptos de estabilidad para separar señales", Departamento de Ingeniería, Facultad de Ciencias Exactas, Universidad Nacional del Nordeste, Corrientes – Argentina. Resumen: E-004, Comunicaciones Científicas y tecnológicas. 2006.
[8] A. Hyvärinen. New approximations of differential entropy for independent component analysis and projection pursuit. In Advances in Neural Information Processing Systems, 10:273–279, 1998.
[9] S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. In Advances in Neural Information Processing Systems, pages 757–763. MIT Press, 1996.
[10] D. Schobben, K. Torkkola, and P. Smaragdis. Evaluation of blind signal separation methods. Proceedings Int. Workshop Independent Component Analysis and Blind Signal Separation, pp. 261-266, Aussois, France, January 11-15, 1999.
[11] J. F. Cardoso and B. Laheld. Equivariant adaptive source separation. IEEE Trans. on Signal Processing, 44:3017-3030, 1996.