General Block LMS Algorithm

María Elena Domínguez
Dept. Appl. Math., ETSII
Universidad Politécnica de Madrid
edominguez@etsii.upm.es

Wilmar Hernandez
Department of Circuits and Systems, EUITT
Universidad Politécnica de Madrid
whernan@ics.upm.es

Gabriela Sansigre
Dept. Appl. Math., ETSII
Universidad Politécnica de Madrid
gsansigre@etsii.upm.es

***Abstract*— In this paper, we analyze the conventional Block-Least-Mean-Square (BLMS) algorithm. Usual constraints such as real input data, steady-state analysis and positive adaptive step-size parameter are discarded. Some modifications are introduced in order that the new complex frequency-domain BLMS algorithm equals the former versions in case any of the constraints are imposed. Furthermore, if the steady-state analysis is considered, the proposed algorithm avoids the inversion of the autocorrelation matrix of the transformed input.**

I. Introduction

Due to their many good performance characteristics, adaptive filters have been satisfactorily used for canceling unknown interferences contained in the relevant signal of a wide range of dynamic systems for many years. These filters perform satisfactorily in environments where system designers do not have complete knowledge of the statistical characteristics of the input data.

In accordance with [1], adaptive filtering is gaining favor in numerous applications to help cope with time-variations of system parameters, and to compensate for the lack of a priori knowledge of the statistical properties of the input data. Over the last several years, a wide range of algorithms has been developed.

However, in spite of the fact that there is a wide variety of adaptive filters available to a system designer, it is important to understand the capabilities and limitations of the adaptive filtering algorithms in order to use the most appropriate algorithm for each specific application [2]. Also, the choice of an adaptive filtering algorithm for an application of interest should be made taking into consideration the following important issues: computational cost, performance, and robustness [2]. Nevertheless, not all the adaptive filtering algorithms available in the scientific literature on adaptive filters have a very low computational cost and very good performance and robustness characteristics either [3].

In this paper we reformulate the work of Lee and Un [4] as well as the works presented in [2], [5]–[8] so that the filter input needs neither be a stationary signal nor a real one. Also, in the formulation given in [4] one needs to invert the autocorrelation matrix of the transformed input; however, in our proposal –more general– we follow the idea of [2], [5] and the inversion of the above autocorrelation matrix is avoided by using a cross-correlation vector between the input and the error signal.

Finally, we show a signal-flow graph representation of the proposed algorithm.

In Section II the complex BLMS algorithm is presented, Section III provides an efficient implementation of the complex BLMS algorithm by using the FFT algorithm and Section IV is devoted to the conclusions.

Nomenclature— All vectors are column vectors of complex elements, written with bold lowercase letters; bold uppercase letters stand for matrices; * denotes complex conjugate, ^{T} transposition and ^{H} transpose-conjugate for vectors and matrices. $\mathrm{diag}(\mathbf{x})$ denotes the diagonal matrix whose principal diagonal is the vector $\mathbf{x}$, and $\circledast$ stands for the circular convolution of two finite vectors. The fast Fourier transform (FFT) of a vector $\mathbf{x}$ of length N is given by $FFT[\mathbf{x}] = \mathbf{F}_N^* \mathbf{x}$ and the inverse FFT of such a vector is given by $IFFT[\mathbf{x}] = \frac{1}{N}\mathbf{F}_N \mathbf{x}$, where the Fourier matrix is defined as

$$\mathbf{F}_N = (w^{kj})_{\ 0\le k,j\le N-1}, \quad w = e^{i2\pi/N}$$

The well-known property

$$FFT[\mathbf{x} \circledast \mathbf{y}] = \mathrm{diag}(FFT[\mathbf{x}])\ FFT[\mathbf{y}]$$

will be used as well as

$$\frac{1}{N}\mathbf{F}_N^2 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} x_1 \\ x_N \\ \vdots \\ x_2 \end{bmatrix}. \tag{1}$$

II. General Complex BLMS Algorithm

Let us consider the BLMS adaptive algorithm with complex data, given by

$$\begin{aligned} &e(kL+i) = d(kL+i) - \hat{\mathbf{w}}^H(k)\mathbf{u}(kL+i) \\ &\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \mu \sum_{j=0}^{L-1} e^*(kL+j)\mathbf{u}(kL+j) \\ &\text{for} \quad 0 \le i \le L-1,\ k = 1, 2, \ldots \end{aligned}$$

where L is the block length, M is the number of taps, μ is the step-size parameter, $d(n)$ is the value of the primary signal at time n, $e(n)$ is the error at time n, the finite vector $\mathbf{u}(n)$ is

978-1-4244-4649-0/09/$25.00 ©2009 IEEE

given by the samples from time n to time $n-M+1$ of the reference signal and $\hat{\mathbf{w}}(k)$ is the tap-weight vector:

$$\mathbf{u}(n) = \begin{bmatrix} u(n) \\ \vdots \\ u(n-M+1) \end{bmatrix}$$

and

$$\hat{\mathbf{w}}(k) = \begin{bmatrix} \hat{w}_0(k) \\ \vdots \\ \hat{w}_{M-1}(k) \end{bmatrix}$$

Note that for real signals we obtain the BLMS algorithm presented in [2], [5] for real data. Therefore, the above equations can be seen as the generalization of the BLMS algorithm.

Let us introduce some notations. For each fixed index k, we denote

$$\mathbf{A}^T(k) = \begin{bmatrix} \mathbf{u}(kL) & \dots & \mathbf{u}(kL+L-1) \end{bmatrix}, \quad (2)$$

and

$$\mathbf{e}(k) = \begin{bmatrix} e(kL) \\ \vdots \\ e(kL+L-1) \end{bmatrix}. \quad (3)$$

$\mathbf{A}^T(k)$ is an M-by-L matrix which allows us to write the adjustment equation as

$$\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \mu \mathbf{A}^T(k)\mathbf{e}^*(k) \quad (4)$$

or

$$\hat{\mathbf{w}}^*(k+1) = \hat{\mathbf{w}}^*(k) + \mu^* \mathbf{A}^H(k)\mathbf{e}(k) := \hat{\mathbf{w}}^*(k) + \mu^* \Phi(k),$$

where $\Phi(k)$ is defined as the cross-correlation vector given by

$$\Phi(k) = \mathbf{A}^H(k)\mathbf{e}(k). \quad (5)$$

III. Implementation of the complex BLMS algorithm by using FFT

In order to carry out an efficient implementation of the BLMS algorithm, the filter parameters are computed in the frequency domain by using the FFT algorithm [2], [5]–[8]. For computational complexity purposes, we consider $L = M$ [2].

Let us define

$$\tilde{\mathbf{u}}(k) = \begin{bmatrix} u(kM-M) \\ \vdots \\ u(kM+M-1) \end{bmatrix},$$

$$\mathbf{U}(k) = \mathrm{diag}(FFT\left[\tilde{\mathbf{u}}(k)\right]),$$

$$\widehat{\mathbf{W}}_*(k) = FFT \begin{bmatrix} \hat{\mathbf{w}}^*(k) \\ \mathbf{0} \end{bmatrix},$$

and

$$\mathbf{E}(k) = FFT \begin{bmatrix} \mathbf{0} \\ \mathbf{e}(k) \end{bmatrix},$$

where $\mathbf{0}$ is the M-by-1 null vector.

In this section we provide the following result:

Theorem:

(a) The output $\hat{\mathbf{w}}^H(k)\mathbf{u}(kM+j)$ $(0 \le j \le M-1)$ is equal to the last M elements of $IFFT[\mathbf{U}(k)\widehat{\mathbf{W}}_*(k)]$.

(b) $\Phi(k)$ is the vector consisting of the first M elements of $IFFT[\mathbf{U}^H(k)\mathbf{E}(k)]$.

Proof: In order to prove assertion (a), we use the circular convolution product

$$IFFT\left[\mathbf{U}(k)\widehat{\mathbf{W}}_*(k)\right] = \tilde{\mathbf{u}}(k) \circledast \begin{bmatrix} \hat{\mathbf{w}}^*(k) \\ \mathbf{0} \end{bmatrix}.$$

Now, to obtain the last M elements of the above circular convolution product, we take the circulant matrix that has $[\hat{\mathbf{w}}^*(k)\ \mathbf{0}]^T$ as first column. The $(M+j+1)$-row $(0 \le j \le M-1)$ of that circulant matrix is given by

$$\begin{bmatrix} 0 & \overset{j+1)}{\dots} & 0 & \hat{w}^*_{M-1}(k) & \dots & \hat{w}^*_0(k) & 0 & \overset{M-j-1)}{\dots} & 0 \end{bmatrix}.$$

Hence the j-th element of the vector consisting of the last M elements of the above circular convolution product is given by

$$\sum_{m=0}^{M-1} \hat{w}^*_m(k) u(kM+j-m)$$

$$= \left[\hat{w}^*_0(k), \hat{w}^*_1(k), \dots, \hat{w}^*_{M-1}(k)\right] \begin{bmatrix} u(kM+j) \\ u(kM+j-1) \\ \vdots \\ u(kM+j-M+1) \end{bmatrix}$$

$$= \hat{\mathbf{w}}^H(k)\mathbf{u}(kM+j).$$

So this concludes the proof of the result (a).

Next we will prove assertion (b). To do so, we need to build the vector $\mathbf{v}(k)$ such that

$$\mathbf{U}^H(k) = \mathrm{diag}\left(FFT\left[\mathbf{v}(k)\right]\right).$$

To this end, we use (1) and obtain

$$\mathbf{v}(k) = \frac{1}{2M}\mathbf{F}^2_{2M}\tilde{\mathbf{u}}^*(k) = \begin{bmatrix} u^*(kM-M) \\ u^*(kM+M-1) \\ \vdots \\ u^*(kM+1) \\ u^*(kM) \\ u^*(kM-1) \\ \vdots \\ u^*(kM-M+1) \end{bmatrix}. \quad (6)$$

Therefore

$$IFFT[\mathbf{U}^H(k)\mathbf{E}(k)] = \mathbf{v}(k) \circledast \begin{bmatrix} \mathbf{0} \\ \mathbf{e}(k) \end{bmatrix}.$$

Now, it suffices to compute the first M elements of the above circular convolution product, which are given by the product of the matrix

$$\begin{bmatrix} u^*(kM) & u^*(kM+1) & \dots & u^*(kM+M-1) \\ u^*(kM-1) & u^*(kM) & \dots & u^*(kM+M-2) \\ \vdots & \vdots & & \vdots \\ u^*(kM-M+1) & u^*(kM-M+2) & \dots & u^*(kM) \end{bmatrix}$$

times the vector

$$\begin{bmatrix} e(kM) \\ \vdots \\ e(kM+M-1) \end{bmatrix},$$

but such a product turns out to be

$$\mathbf{A}^H(k)\mathbf{e}(k) = \Phi(k)$$

which concludes the proof. ■

The signal-flow graph representation of the fast complex BLMS algorithm is shown in Fig. 1, where the recursive relation for updating the tap-weight vector of the filter in the frequency domain is given by

$$\widehat{\mathbf{W}}_*(k+1) = \widehat{\mathbf{W}}_*(k) + \mu^* FFT \begin{bmatrix} \Phi(k) \\ \mathbf{0} \end{bmatrix}.$$

At this point, it should be highlighted that the signal-flow graph representation presented in this paper is a generalization of the one shown in [2], [5] because our approach can operate on both real- and complex-valued data; besides, its implementation requires no extra computational complexity. Moreover, for real-valued data the proposed complex algorithm is exactly the same as the conventional BLMS algorithm shown in [2]. In addition, the signal-flow graph representation shown in Fig. 1 is easier to implement than the one shown in [4].

IV. Conclusions

In this paper we revisited the conventional BLMS algorithm as presented in [2] in order to make it suitable for complex data, without extra operations. Moreover, our algorithm can be applied for non steady-state data signals and it avoids inverting the autocorrelation matrix of the transform input, as it is done in [4].

Acknowledgment

This work has been partially supported by the Universidad Politécnica de Madrid through the UPM TACA research group, and the Ministry of Education and Science (MEC) of Spain under the research project TEC2007-63121.

References

[1] A. H. Sayed and T. Kailath, "A state-space approach to adaptive RLS filtering," *IEEE Signal Process. Mag.,* vol. 11, pp. 18-60, Jul. 1994.

[2] S. Haykin, *Adaptive Filter Theory*, 4th ed. Upper Saddle River, NJ: Prentice-Hall, 2002.

[3] B. Hassibi, A. H. Sayed, and T. Kailath, "H^∞ optimality of the LMS algorithm," *IEEE Trans. Signal Process.*, vol. 44, pp. 267-280, Feb. 1996.

[4] J. C. Lee, C. K. Un, "Block Realization of Multirate Adaptive Digital Filters," *IEEE Trans. Acoust. Speech Signal Process.,* vol. 34, pp. 105-117, Feb. 1986.

[5] J. J. Shynk, "Frequency-domain and multirate adaptive filtering," *IEEE Signal Process. Mag.,* vol. 9, pp. 14-37, Jan. 1992.

[6] E. Ferrara, "Fast implementation of LMS adaptive filters," *IEEE Trans. Acoust. Speech Signal Process.,* vol. 28, pp. 474-475, Aug. 1980.

[7] G. A. Clark, S. K. Mitra, S. R. Parker, "Block implementation of adaptive digital filters," *IEEE Trans. Circuits Syst.,* vol. 28, pp. 584-592, Jun. 1981.

[8] G. A. Clark, S. R. Parker, S. K. Mitra, "A unified approach to time- and frequency- domain realization of FIR adaptive digital filters," *IEEE Trans. Acoust. Speech Signal Process.,* vol. 31, pp. 1073-1083, Oct. 1983.

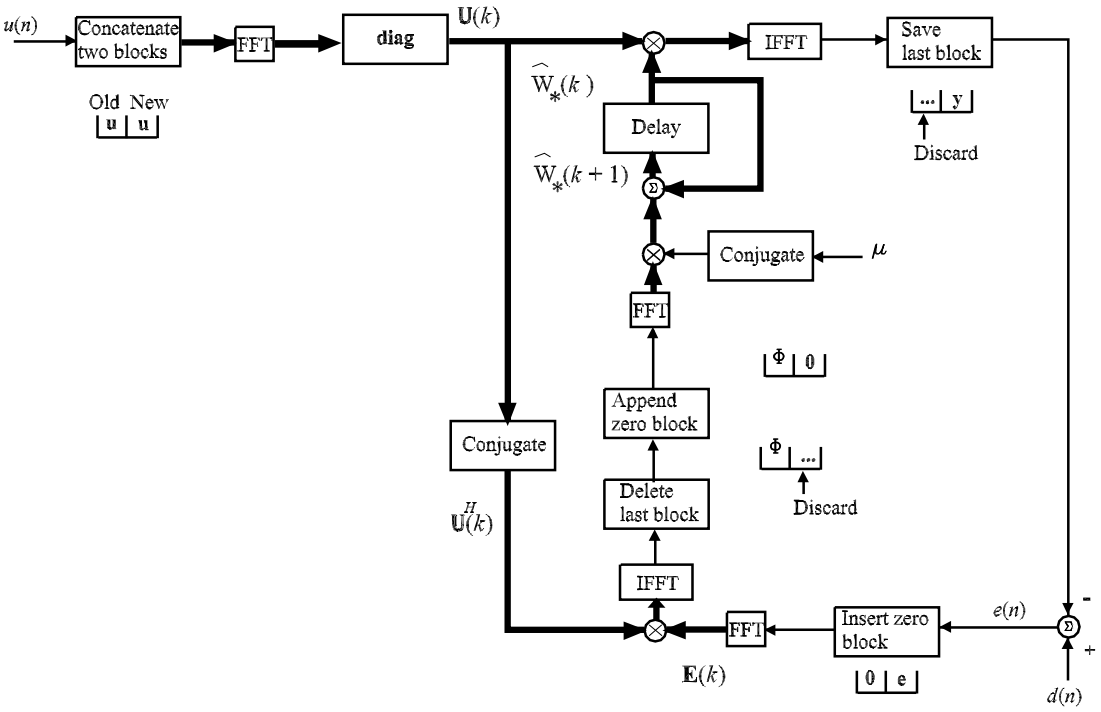

Fig. 1. Signal-flow graph representation of the fast complex BLMS algorithm