Infinitely Reiterated Data-reusing LMS Algorithm

R. F. Vigelis, A. L. F. de Almeida, J. C. M. Mota

***Resumo*— Este trabalho apresenta um novo algoritmo de filtragem baseado na técnica de reuso de dados sobre LMS. O algoritmo proposto reusa os dados recebidos, da amostra inicial até a amostra atual, um número infinito de vezes. A formulação algébrica do algoritmo resultante mostrou-se similar ao algoritmo RLS. O novo algoritmo converge tão rápido quanto o algoritmo RLS, e converge para o mesmo erro médio quadrático, em regime permanente, produzido pelo LMS. Com um valor pequeno do fator de passo, o algoritmo proposto atinge a solução de Wiener, sem qualquer degradação em sua taxa de convergência. Simulações em computador confirmaram tais observações.**

***Palavras-Chave*— Algoritmos adaptativos, LMS, RLS, APA, Reuso de dados.**

***Abstract*— This work presents a new filtering algorithm based on data-reusing LMS techniques. The proposed algorithm reuses the received data from the initial sample up to the current sample an infinity number of times. The resulting algebraic formulation of the algorithm have shown to be similar to the RLS algorithm. The new algorithm converges as fast as the RLS algorithm, and converges to the same LMS misadjustment. With a small step-size value, the proposed algorithm can achieve the Wiener solution without any degradation in its convergence rate. Computer simulations have confirmed these observations.**

***Keywords*— Adaptive algorithms, LMS, RLS, APA, Data-reusing techniques.**

I. INTRODUCTION

In the class of data-reusing algorithms past samples are reprocessed in order to achieve better convergence performance. In principle, data-reusing techniques can be successfully applied to any stochastic gradient based algorithm. But here we focus our attention to the LMS case. There are basically two categories of data-reusing algorithms based on the LMS algorithm. If at each iteration the current sample is reused, we have the data-reusing LMS (DR-LMS) algorithm. For a detailed analysis of the DR-LMS algorithm, see [1]. When a fixed number of past samples is reused, the Schnaufer and Jenkins data-reusing LMS (SJ-DR-LMS) algorithm is obtained. The SJ-DR-LMS algorithm (also named New Data-reusing LMS Algorithm) is proposed in [2]. The terminology adopted here is the same as in [3]. As the number of reuses increases, data-reusing LMS based algorithms present faster convergence, but its computational complexity increases proportionally. There is a trade-off between computational complexity and convergence rate.

The SJ-DR-LMS algorithm is strongly related to the affine projection algorithm (APA) [4], [5], [6]. In the APA, the coefficient vector is updated in a direction that is orthogonal to a hyperplane defined by the last K input data vectors. When the numbers of reiterations R (see details in Section II) in the SJ-DR-LMS tends to infinity, the resulting algorithm is equivalent to the APA [7], [8]. The SJ-DR-LMS algorithm can be seen as an approximation to the APA. This fact permits to analyze the convergence performance of SJ-DR-LMS based algorithms.

R. F. Vigelis, A. L. F. Almeida, J. C. M. Mota, Universidade Federal do Ceará, Campus do Pici, CT–DETI, Fortaleza–CE, Brazil, E-mails: rfvigelis@deti.ufc.br, andre@gtel.ufc.br, mota@deti.ufc.br. This work was supported by FUNCAP – Fundação Cearense de Amparo à Pesquisa.

The proposed algorithm, at each iteration, reuses data from the initial sample up to the current sample an infinity number of times. This assures a fast convergence of the proposed algorithm, whose simulations showed to be as fast as the RLS algorithm. The steady-state mean-square error value of the proposed algorithm, as also confirmed by simulations, for equal step-size value, is the same of the LMS algorithm. The algebraic formulations of the RLS and the proposed algorithm have shown to be similar, so resulting in computational complexity of $O(N^2)$. The proposed algorithm mixes RLS and LMS characteristics: it converges fast and presents a residual mean-square error determined by a step-size parameter.

This paper is organized as follows. Section II summarizes graphical interpretations of the LMS, the affine projection and the SJ-DR-LMS algorithms. Section III contains the algebraic development of the proposed algorithm and a comparison to the RLS algorithm. In Section IV, we present some simulation results. Finally, our conclusions and perspectives are contained in Section V.

II. PRELIMINARY DISCUSSION

Let $\mathbf{x}_n = (x_n, \ldots, x_{n-N+1})^T$ be the vector containing the last N past samples of the input signal sequence (x_i), $i \geq 1$, where the superscript ^{T} denotes transpose of vector or matrix. In the LMS algorithm, the coefficient vector $\mathbf{w}_n$ of dimension $N \times 1$ is updated recursively as

$$e_n = d_n - \mathbf{w}_n^H \mathbf{x}_n \tag{1}$$

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \mathbf{x}_n e_n^*, \tag{2}$$

where the superscript ^{H} stands for hermitian operation. In the LMS algorithm, the system output $\hat{y}_n = \mathbf{w}_n^H \mathbf{x}_n$ approximates as close as possible the desired response d_n. Convergence is ensured if the step-size parameter μ obeys $0 < \mu < 2/\lambda_{\max}$, where $\lambda_{\max}$ is the maximum eigenvalue of the input signal's correlation matrix. A wide discussion concerning the LMS algorithm can be found in [4], [9].

Now we will describe geometrically the LMS algorithm. Let $\mathcal{S}_n$ be the translated subspace of dimension $N-1$ that contains all vectors $\mathbf{w}$ such that $\mathbf{x}_n^H \mathbf{w} = d_n^*$. Except for a noise term, the optimum Wiener solution $\mathbf{w}_o$ is in $\mathcal{S}_n$. The coefficient-updating equations (1)-(2) move $\mathbf{w}_n$ orthogonally to $\mathcal{S}_n$. Depending on the value of μ, if $0 < \mu < \frac{1}{\|x_n\|^2}$, $\mu = \frac{1}{\|x_n\|^2}$ or $\frac{1}{\|x_n\|^2} < \mu < \frac{2}{\|x_n\|^2}$, the vector $\mathbf{w}_{n+1}$, which is in $\mathbf{w}_n + \operatorname{span}(\mathbf{x}_n)$, will lie between $\mathbf{w}_n$ and $\mathcal{S}_n$, in $\mathcal{S}_n$, or beyond $\mathcal{S}_n$, respectively, as illustrated in Fig. 1. In this manner, the updated coefficient vector $\mathbf{w}_{n+1}$ results closer to $\mathbf{w}_o$.

The LMS algorithm can be improved if $\mathbf{w}_n$ is projected onto $\mathcal{S}_n^{(L)} = \mathcal{S}_n \cap \cdots \cap \mathcal{S}_{n-L+1}$, i.e., $\mathcal{S}_n^{(L)}$ is an hyperplane that contains all vectors $\mathbf{w}$ such that $\mathbf{X}_n^H \mathbf{w} = \mathbf{d}_n^*$, where $\mathbf{X}_n = (\mathbf{x}_n, \ldots, \mathbf{x}_{n-L+1})$ and $\mathbf{d}_n = (d_n, \ldots, d_{n-L+1})^T$. As $\mathcal{S}_n^{(L)}$ has dimension smaller than $\mathcal{S}_n$, and contains $\mathbf{w}_o$, the updated vector $\mathbf{w}_{n+1}$ will be closer to $\mathbf{w}_o$, compared to the LMS algorithm. Let $\mathbf{X}_n(\mathbf{X}_n^H \mathbf{X}_n)^{-1}\mathbf{X}_n^H$ be the projection operator onto $\mathcal{S}_n^{(L)}$. Then $\mathbf{X}_n(\mathbf{X}_n^H \mathbf{X}_n)^{-1}(\mathbf{d}_n^* - \mathbf{X}_n^H \mathbf{w}_n)$ is the translation vector added to $\mathbf{w}_n$ in order to obtain the vector projected onto $\mathcal{S}_n^{(L)}$. The coefficient-updating equations becomes:

$$\mathbf{e}_n = \mathbf{d}_n^* - \mathbf{X}_n^H \mathbf{w}_n$$
$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \mathbf{X}_n (\mathbf{X}_n^H \mathbf{X}_n)^{-1} \mathbf{e}_n.$$

The equations above define the affine projection algorithm. In the case $L = 2$, we have the binormalized data-reusing LMS algorithm [10]. The APA is strongly related to the SJ-DR-LMS algorithm, as we will discuss bellow.

The SJ-DR-LMS algorithm is based on a simple and efficient idea. At instant $n-1$ the LMS algorithm generates $\mathbf{a} = \mathbf{w}_n$. Let $m < n$. If the updating-coefficient equations (1)-(2) are initialized by $\mathbf{a}$, and updated by $\mathbf{x}_i$ and d_i in any order for $m \le i \le n$, then the resulting coefficient vector $\mathbf{b}$, compared to $\mathbf{w}_n$, except for a noise term, will be closer to the Wiener solution. The SJ-DR-LMS algorithm, at each instant n, for $m = n - K + 1$, realizes this operation using a decreasing order of indices ($i = n, \ldots, n-K+1$), and so updates $\mathbf{w}_{n+1} = \mathbf{b}$. In the proposed algorithm, instead, we adopt an increasing order of updating. The performance of the technique above is improved if this procedure is repeated R times. Summarizing, a general data-reusing LMS algorithm may be expressed by the following steps, where K_n and R_n are the number of reuses and reiterations at time n, respectively:

```
Initialization: $\mathbf{w}_n^{(0,0)} = \mathbf{w}_n$
  For 1: $i = 0, \ldots, R_n - 1$
    For 2: $j = 0, \ldots, K_n - 1$
      $k = n - K_n + 1 + j$
      $e_n^{(i,j)} = d_k - (\mathbf{w}_n^{(i,j)})^H \mathbf{x}_k$                (3)
      $\mathbf{w}_n^{(i,j+1)} = \mathbf{w}_n^{(i,j)} + \mu_k \mathbf{x}_k (e_n^{(i,j)})^*$     (4)
    End For 2
  $\mathbf{w}_n^{(i+1,0)} = \mathbf{w}_n^{(i,K_n)}$
  End For 1
Update: $\mathbf{w}_{n+1} = \mathbf{w}_n^{(R_n,0)}$
```

The subscript of μ_k indicates that this variable is function of k. At each increment j, as the updating equations (3)-(4) move $\mathbf{w}_n^{(i,j)}$ orthogonally to $\mathcal{S}_{n-K_n+1+j}$, the resulting vector $\mathbf{w}_n^{(i,j+1)}$ approximates to the Wiener solution $\mathbf{w}_o$. The procedure above corresponds to a sequence of repeated projections towards $\mathcal{S}_{n-K_n+1}, \ldots, \mathcal{S}_n$. Increases in the values of K_n and R_n lead to coefficient vectors $\mathbf{w}_{n+1}$ closer to $\mathbf{w}_o$.

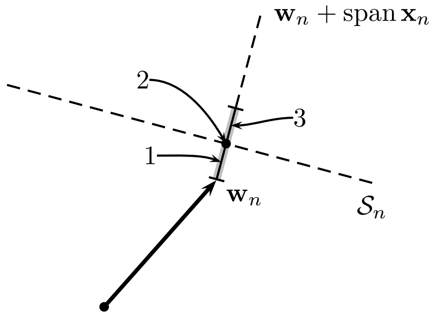


Fig. 1. Geometrical interpretation of the LMS algorithm. The updated vector $\mathbf{w}_{n+1}$ will lie in region 1, point 2 or region 3 if $0 < \mu < \frac{1}{\|\mathbf{x}_n\|^2}$, $\mu = \frac{1}{\|\mathbf{x}_n\|^2}$ or $\frac{1}{\|\mathbf{x}_n\|^2} < \mu < \frac{2}{\|\mathbf{x}_n\|^2}$, respectively.

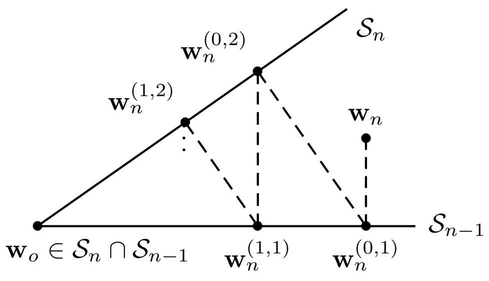


Fig. 2. Geometrical illustration of the reiterated version of the SJ-DR-LMS algorithm for $\mu_n = \frac{1}{\|\mathbf{x}_n\|^2}$ and $K_n = 2$.

Fig. 2 illustrates geometrically the procedure above for the simple case $\mu_n = \frac{1}{\|x_n\|^2}$ and $K_n = 2$. In this illustration, we take the assumption that the algorithm operates on a noise free environment, where we have $\mathbf{w}_o \in \mathcal{S}_n \cap \mathcal{S}_{n-1}$. Initially, the vector $\mathbf{w}_n^{(0,0)} = \mathbf{w}_n$ is projected onto $\mathcal{S}_{n-1}$. The obtained vector $\mathbf{w}_n^{(0,1)}$ is projected onto $\mathcal{S}_n$. These operations continue up to $i = R_n - 1$ and $j = 2$. The final vector $\mathbf{w}_n^{(R_n-1,2)}$ results closer to $\mathcal{S}_n^{(2)}$, compared to $\mathbf{w}_n^{(0,1)}$. If R_n tends to infinity, the final vector $\mathbf{w}_n^{(R_n-1,2)}$ approximates to the orthogonal projection of $\mathbf{w}_n$ onto $\mathcal{S}_n^{(2)}$. Such geometrical argument links the SJ-DR-LMS algorithm to the APA. These results can be extended to any μ_n and K_n. The SJ-DR-LMS algorithm when is infinitely reiterated ($R_n \to \infty$) is equivalent to the affine projection algorithm. For a theoretical treatment linking the APA and the SJ-DR-LMS algorithm, in the Kaczmarz's method context, see [7] and its references.

In the proposed algorithm, the updating equations (3)-(4) are repeated for $\mu_n = \mu$, $K_n = n$ and $R_n \to \infty$. As $\mathcal{S}_n^{(N)}$ has null dimension and equals $\mathbf{w}_o$ except for a noise term, just $K_n = N$ would be necessary in order to obtain fast convergence. But due to computational considerations and noise impairments, we prefer to adopt $K_n = n$.

In the next section we effectuate the algebraic development of the proposed algorithm and a comparison to the RLS algorithm.

III. The Proposed Algorithm

A. Algebraic development

Now we will express $\mathbf{w}_n^{(i,n)}$ as a function of $\mathbf{w}_n^{(i,0)}$, where $K_n = n$ is assumed. Rewriting the updating equations (3)-(4) as $\mathbf{w}_n^{(i,j+1)} = (\mathbf{I} - \mu\mathbf{Q}_{j+1})\mathbf{w}_n^{(i,j)} + \mu\mathbf{x}_{j+1}d_{j+1}^*$, where $\mathbf{Q}_{j+1} = \mathbf{x}_{j+1}\mathbf{x}_{j+1}^H$, and $\mathbf{I}$ is the identity matrix with proper dimension, we have

$$\begin{aligned}
\mathbf{w}_n^{(i,1)} &= (\mathbf{I} - \mu\mathbf{Q}_1)\mathbf{w}_n^{(i,0)} + \mu\mathbf{x}_1 d_1^* \\
\mathbf{w}_n^{(i,2)} &= (\mathbf{I} - \mu\mathbf{Q}_2)\mathbf{w}_n^{(i,1)} + \mu\mathbf{x}_2 d_2^* \\
&\vdots \\
\mathbf{w}_n^{(i,n)} &= (\mathbf{I} - \mu\mathbf{Q}_n)\mathbf{w}_n^{(i,n-1)} + \mu\mathbf{x}_n d_n^*.
\end{aligned}$$

Substituting the right side of the equation that defines $\mathbf{w}_n^{(i,j)}$ for $\mathbf{w}_n^{(i,j)}$ in the equation that defines $\mathbf{w}_n^{(i,j+1)}$, and doing this for $j = 1, \ldots n$, we obtain

$$\begin{aligned}
\mathbf{w}_n^{(i,n)} &= \underbrace{\prod_{k=1}^{n}(\mathbf{I} - \mu\mathbf{Q}_k)}_{\mathbf{A}_n}\mathbf{w}_n^{(i,0)} + \underbrace{\mu\sum_{k=1}^{n}\prod_{l=k+1}^{n}(\mathbf{I} - \mu\mathbf{Q}_l)\mathbf{x}_k d_k^*}_{\tilde{\mathbf{w}}_{n+1}} \\
&= \mathbf{A}_n\mathbf{w}_n^{(i,0)} + \tilde{\mathbf{w}}_{n+1} \qquad (5)
\end{aligned}$$

In the matrix product above pre-multiplications are realized. The terms $\mathbf{A}_n$ and $\tilde{\mathbf{w}}_{n+1}$ are recursively determined by

$$\begin{aligned}
\mathbf{A}_n &= (\mathbf{I} - \mu\mathbf{Q}_n)\mathbf{A}_{n-1} \\
\tilde{\mathbf{w}}_{n+1} &= (\mathbf{I} - \mu\mathbf{Q}_n)\tilde{\mathbf{w}}_n + \mu\mathbf{x}_n d_n^*
\end{aligned}$$

or

$$\begin{aligned}
\tilde{e}_n &= d_n - \tilde{\mathbf{w}}_n^H\mathbf{x}_n \\
\tilde{\mathbf{w}}_{n+1} &= \tilde{\mathbf{w}}_n + \mu\mathbf{x}_n\tilde{e}_n^*.
\end{aligned}$$

Notice that $\tilde{\mathbf{w}}_n$ is recognized as the coefficient vector of the LMS algorithm initialized by the null vector $\tilde{\mathbf{w}}_0 = \mathbf{0}$.

Now let's iterate over the superscript i in $\mathbf{w}_n^{(i,0)}$. From (5), observing that $\mathbf{w}_n^{(i+1,0)} = \mathbf{w}_n^{(i,n)}$, we obtain

$$\begin{aligned}
\mathbf{w}_n^{(0,0)} &= \mathbf{w}_n \\
\mathbf{w}_n^{(1,0)} &= \mathbf{A}_n\mathbf{w}_n^{(0,0)} + \tilde{\mathbf{w}}_{n+1} \\
\mathbf{w}_n^{(2,0)} &= \mathbf{A}_n\mathbf{w}_n^{(1,0)} + \tilde{\mathbf{w}}_{n+1} \\
&\vdots \\
\mathbf{w}_{n+1} = \mathbf{w}_n^{(R_n,0)} &= \mathbf{A}_n\mathbf{w}_n^{(R_n-1,0)} + \tilde{\mathbf{w}}_{n+1}
\end{aligned}$$

Substituting the right side of the equation that defines $\mathbf{w}_n^{(i,0)}$ for $\mathbf{w}_n^{(i,0)}$ in the equation that defines $\mathbf{w}_n^{(i+1,0)}$, and doing this for $i = 1, \ldots, R_n$, we get

$$\begin{aligned}
\mathbf{w}_{n+1} &= \mathbf{A}_n^{R_n}\mathbf{w}_n + (\mathbf{I} + \mathbf{A}_n + \mathbf{A}_n^2 + \cdots + \mathbf{A}_n^{R_n-1})\tilde{\mathbf{w}}_{n+1} \\
&= \mathbf{A}_n^{R_n}\mathbf{w}_n + (\mathbf{I} - \mathbf{A}_n^{R_n})\cdot(\mathbf{I} - \mathbf{A}_n)^{-1}\tilde{\mathbf{w}}_{n+1}. \qquad (6)
\end{aligned}$$

With $0 < \mu < \frac{2}{\|\mathbf{x}_n\|}$, which ensures the LMS algorithm convergence, the matrix $\mathbf{A}_n^{R_n}$ tends to the null matrix as $R_n \to \infty$. Then, with $R_n \to \infty$, equation (6) becomes

$$\mathbf{w}_{n+1} = (\mathbf{I} - \mathbf{A}_n)^{-1}\tilde{\mathbf{w}}_{n+1}. \qquad (7)$$

Now we will determine the matrix $(\mathbf{I} - \mathbf{A}_n)^{-1}$ recursively. To do this, we will use the matrix inversion lemma. Let $\mathbf{D}$, $\mathbf{E}$ and $\mathbf{G}$ be non-singular matrices. It is assumed that $\mathbf{D}$ and $\mathbf{E}$ have the same dimension $N \times N$ and $\mathbf{G}$ has dimension $M \times M$. Let $\mathbf{F}$ be a matrix of dimension $N \times M$. If these matrices follow the relation

$$\mathbf{D} = \mathbf{E}^{-1} + \mathbf{F}\mathbf{G}^{-1}\mathbf{F}^H,$$

then

$$\mathbf{D}^{-1} = \mathbf{E} - \mathbf{E}\mathbf{F}(\mathbf{G} + \mathbf{F}^H\mathbf{E}\mathbf{F})^{-1}\mathbf{F}^H\mathbf{E}.$$

Applying the matrix inversion lemma to the following equation

$$\begin{aligned}
\mu^{-1}(\mathbf{I} - \mathbf{A}_n)\mathbf{A}_{n-1}^{-1} &= \mu^{-1}[\mathbf{I} - (\mathbf{I} - \mu\mathbf{x}_n\mathbf{x}_n^H)\mathbf{A}_{n-1}]\mathbf{A}_{n-1}^{-1} \\
&= \mu^{-1}(\mathbf{I} - \mathbf{A}_{n-1})\mathbf{A}_{n-1}^{-1} + \mathbf{x}_n\mathbf{x}_n^H,
\end{aligned}$$

with

$$\begin{aligned}
\mathbf{D} &= \mu^{-1}(\mathbf{I} - \mathbf{A}_n)\mathbf{A}_{n-1}^{-1} & \mathbf{F} &= \mathbf{x}_n \\
\mathbf{E} &= \mu\mathbf{A}_{n-1}(\mathbf{I} - \mathbf{A}_{n-1})^{-1} & \mathbf{G} &= 1,
\end{aligned}$$

we have

$$\mu\mathbf{A}_{n-1}(\mathbf{I} - \mathbf{A}_n)^{-1} = \mathbf{C}_{n-1} - \frac{\mathbf{C}_{n-1}\mathbf{x}_n\mathbf{x}_n^H\mathbf{C}_{n-1}}{1 + \mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n}, \qquad (8)$$

where $\mathbf{C}_{n-1} = \mu\mathbf{A}_{n-1}(\mathbf{I} - \mathbf{A}_{n-1})^{-1}$. Pre-multiplying both sides of (8) by $(\mathbf{I} - \mu\mathbf{x}_n\mathbf{x}_n^H)$, and after some simplifications, we get

$$\begin{aligned}
\mathbf{C}_n &= \mathbf{C}_{n-1} - \mu\mathbf{x}_n\mathbf{x}_n^H\mathbf{C}_{n-1} + \\
&\quad + \mu\mathbf{x}_n\mathbf{x}_n^H\frac{\mathbf{C}_{n-1}\mathbf{x}_n\mathbf{x}_n^H\mathbf{C}_{n-1}}{1 + \mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n} - \frac{\mathbf{C}_{n-1}\mathbf{x}_n\mathbf{x}_n^H\mathbf{C}_{n-1}}{1 + \mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n} \\
&= \mathbf{C}_{n-1} - \frac{\mu\mathbf{x}_n\mathbf{x}_n^H\mathbf{C}_{n-1}}{1 + \mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n} - \frac{\mathbf{C}_{n-1}\mathbf{x}_n\mathbf{x}_n^H\mathbf{C}_{n-1}}{1 + \mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n} \\
&= \mathbf{C}_{n-1} - \mathbf{g}_n\mathbf{x}_n^H\mathbf{C}_{n-1}, \qquad (9)
\end{aligned}$$

where

$$\mathbf{g}_n = \frac{(\mathbf{C}_{n-1} + \mu\mathbf{I})\mathbf{x}_n}{1 + \mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n}. \qquad (10)$$

Noticing that $(\mathbf{I} - \mathbf{A}_n)^{-1} = \mu^{-1}\mathbf{C}_n + \mathbf{I}$, equation (7) becomes

$$\mathbf{w}_{n+1} = (\mu^{-1}\mathbf{C}_n + \mathbf{I})\tilde{\mathbf{w}}_{n+1}. \qquad (11)$$

Developing (10), we find that $\mathbf{g}_n$ can be rewritten as follows:

$$\begin{aligned}
\mathbf{g}_n + \mathbf{g}_n\mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n &= (\mathbf{C}_{n-1} + \mu\mathbf{I})\mathbf{x}_n \\
\mathbf{g}_n &= (\mathbf{C}_{n-1} - \mathbf{g}_n\mathbf{x}_n^H\mathbf{C}_{n-1} + \mu\mathbf{I})\mathbf{x}_n \\
\mathbf{g}_n &= (\mathbf{C}_n + \mu\mathbf{I})\mathbf{x}_n.
\end{aligned}$$

Using the equation above, we will eliminate $\tilde{\mathbf{w}}_{n+1}$ from (11) and write $\mathbf{w}_{n+1}$ directly as a function of $\mathbf{w}_n$. Doing so, we

TABLE I
THE PROPOSED AND THE RLS ALGORITHM

Initialization:	
$\delta =$ small positive constant	
$\mathbf{C}_0 = \delta^{-1}\mathbf{I}$	$\mathbf{P}_0 = \delta^{-1}\mathbf{I}$
$\mathbf{w}_0 = \mathbf{0}$	$\hat{\mathbf{w}}_0 = \mathbf{0}$
Proposed algorithm:	
$\mathbf{g}_n = \dfrac{(\mathbf{C}_{n-1} + \mu\mathbf{I})\mathbf{x}_n}{1 + \mathbf{x}_n^H\mathbf{C}_{n-1}\mathbf{x}_n}$	$e_n = d_n - \mathbf{w}_n^H\mathbf{x}_n$
$\mathbf{C}_n = \mathbf{C}_{n-1} - \mathbf{g}_n\mathbf{x}_n^H\mathbf{C}_{n-1}$	$\mathbf{w}_{n+1} = \mathbf{w}_n + \mathbf{g}_n e_n^*$
RLS algorithm:	
$\mathbf{k}_n = \dfrac{\mathbf{P}_{n-1}\mathbf{x}_n}{1 + \mathbf{x}_n^H\mathbf{P}_{n-1}\mathbf{x}_n}$	$\alpha_n = d_n - \hat{\mathbf{w}}_{n-1}^H\mathbf{x}_n$
$\mathbf{P}_n = \mathbf{P}_{n-1} - \mathbf{k}_n\mathbf{x}_n^H\mathbf{P}_{n-1}$	$\hat{\mathbf{w}}_n = \hat{\mathbf{w}}_{n-1} + \mathbf{k}_n\alpha_n^*$

finally get

$$\begin{aligned}\mathbf{w}_{n+1} &= (\mu^{-1}\mathbf{C}_n + \mathbf{I})[(\mathbf{I} - \mu\mathbf{x}_n\mathbf{x}_n^H)\tilde{\mathbf{w}}_n + \mu\mathbf{x}_n d_n^*] \\ &= (\mu^{-1}\mathbf{C}_n + \mathbf{I})\tilde{\mathbf{w}}_n - \\ &\quad - (\mathbf{C}_n + \mu\mathbf{I})\mathbf{x}_n\mathbf{x}_n^H\tilde{\mathbf{w}}_n + (\mathbf{C}_n + \mu\mathbf{I})\mathbf{x}_n d_n^* \\ &= (\mu^{-1}\mathbf{C}_{n-1} + \mathbf{I} - \mu^{-1}\mathbf{g}_n\mathbf{x}_n^H\mathbf{C}_{n-1})\tilde{\mathbf{w}}_n - \\ &\quad - \mathbf{g}_n\mathbf{x}_n^H\tilde{\mathbf{w}}_n + \mathbf{g}_n d_n^* \\ &= (\mu^{-1}\mathbf{C}_{n-1} + \mathbf{I})\tilde{\mathbf{w}}_n - \\ &\quad - \mathbf{g}_n\mathbf{x}_n^H(\mu^{-1}\mathbf{C}_{n-1} + \mathbf{I})\tilde{\mathbf{w}}_n + \mathbf{g}_n d_n^* \\ &= \mathbf{w}_n + \mathbf{g}_n(d_n^* - \mathbf{x}_n^H\mathbf{w}_n) \\ &= \mathbf{w}_n + \mathbf{g}_n e_n^*, \end{aligned} \tag{12}$$

where $e_n = d_n - \mathbf{w}_n^H\mathbf{x}_n$.

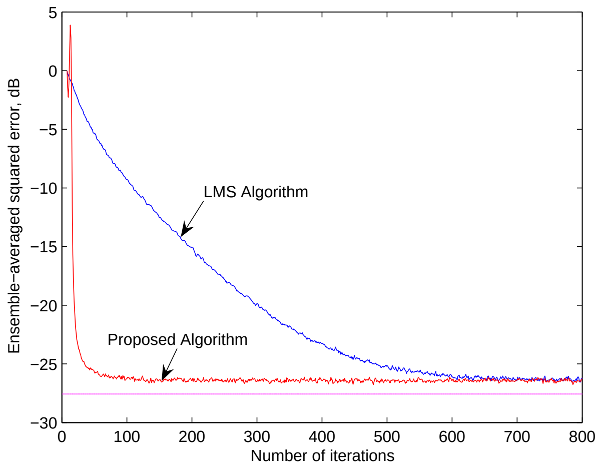
Equations (10), (9) and (12) define the proposed algorithm, which is summarized in the Table I.

B. Comparison to the RLS algorithm

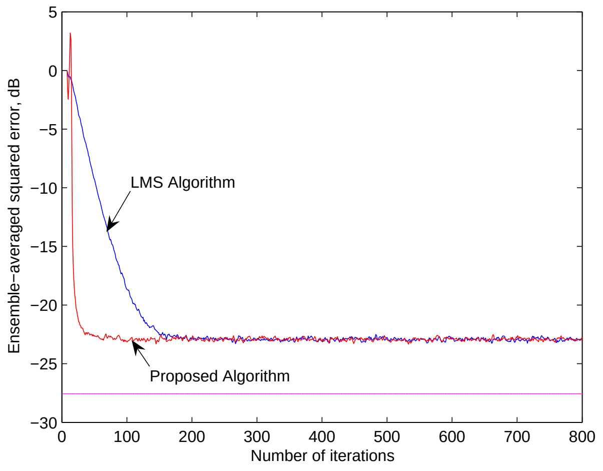
The RLS algorithm is shown in Table I. The matrix $\mathbf{P}_n$ is the inverse of correlation matrix $\mathbf{\Phi}_n = \sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^H$. (Here we do not consider the forgetting factor λ, which corresponds to the "memory" of the algorithm.) Let $\boldsymbol{\theta}_n = \sum_{i=1}^{n}\mathbf{x}_i d_i^*$ be the cross-correlation between the tap inputs of the transversal filter and the desired response. The RLS algorithm estimates in a least-square sense the coefficient vector as $\hat{\mathbf{w}}_n = \mathbf{\Phi}_n^{-1}\boldsymbol{\theta}_n$. See [4], [9] for a detailed RLS discussion.

As can be observed, the proposed and the RLS algorithm are algebraically similar. In both algorithms, the matrices $\mathbf{C}_n$ and $\mathbf{P}_n$ are initialized by $\delta^{-1}\mathbf{I}$, where δ is small. This value avoids the singularity of $\mathbf{C}_1$ and $\mathbf{P}_1$. The matrices $\mathbf{C}_n$ and $\mathbf{P}_n$, as well as the vectors $\mathbf{g}_n$ and $\mathbf{k}_n$, play the same role in an algebraic sense, though they have different interpretations. Notice that $\mathbf{C}_n$ is not hermitian. If μ is taken sufficiently small, the proposed algorithm approaches the RLS algorithm. Each iteration of the proposed algorithm requires $3N^2 + 5N$ multiplications, whereas in the RLS algorithm we need $2N^2 + 4N$ multiplications. Both algorithms have complexity of $O(N^2)$.

In the following section, we present some simulation results that illustrate the performance of the new algorithm.



(a)



(b)

Fig. 3. Comparison of the proposed and the LMS algorithm for (a) $\mu = 0.025$ and (b) $\mu = 0.075$.

IV. SIMULATIONS

We run the proposed algorithm in a simple channel equalization problem. The symbol source consists of a sequence (s_n) that assumes values $+1$ or -1 with equal probability, i.e, we have used a BPSK modulation scheme. This sequence pass through the linear channel $\mathbf{h}$ of length L corrupted by the additive white gaussian noise sequence (ν_n) of variance $\sigma_\nu^2 = 0.001$, resulting in the following received sequence:

$$x_n = \mathbf{h}^H\mathbf{s}_n + \nu_n, \quad \text{for } n = 1, 2, \ldots,$$

where $\mathbf{s}_n = (s_n, \ldots, s_{n-L+1})^T$. The elements of $\mathbf{h}$ are defined by

$$h_i = \begin{cases} 0 & \text{for } i = 0, \\ \dfrac{1}{2}\left[1 + \cos\left(\dfrac{2\pi}{W}(i-2)\right)\right] & \text{for } i = 1, 2, 3, \end{cases}$$

where the factor W controls the distortion produced by the channel, expressed by the eigenvalue spread of the matrix correlation of (x_n). The value $W = 3.1$ was used in the simulation, which corresponds to eigenvalue spread equal to 11.1238. The number of elements of the weight coefficient vector was $N = 11$. This value yields an optimum delay of 7 samples for the LMS algorithm, demanding the desired response $d_n = s_{n-7}$.

For each experiment, the instantaneous squared error between the desired response and the filter equalizer output was averaged over 5.000 independent trials. The line at the bottom of the graphics indicates the optimum Wiener mean-square error. The value $\delta = 10^{-5}$ was used.

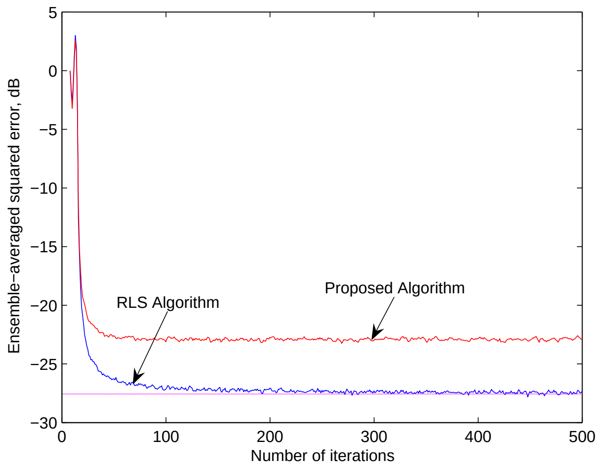
Fig. 3 shows the learning curves of the LMS and the proposed algorithm for $\mu = 0.025$ and 0.075. The convergence rate of the proposed algorithm has shown insensitive to the step-size μ value. The mean-square error of the proposed and the LMS algorithm converged to the same steady-state value. The curves of further experiments for different μ values have obeyed the same behavior as observed above.

Fig. 4-(a) exhibits a comparison of the proposed algorithm for $\mu = 0.075$ and the RLS algorithm. The rate of convergence is similar. The curves differ from the mean-square steady-state value. In Fig. 4-(b), for the proposed algorithm we chose a sufficiently small μ value in order to approximate the minimum mean-square error to the Wiener solution. For $\mu = 0.001$, the proposed algorithm behaves similarly to the RLS algorithm, as it was expected.

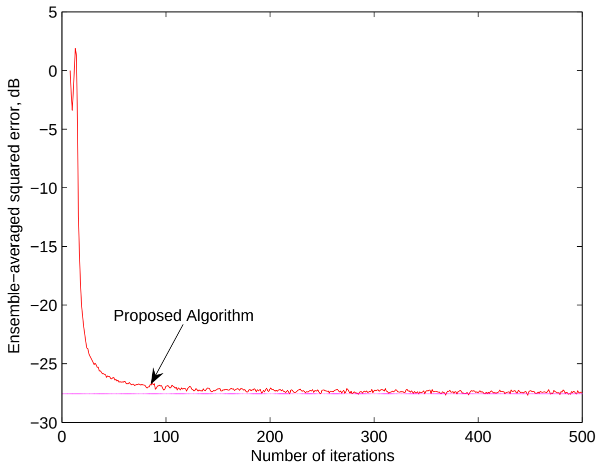
V. Conclusions and Perspectives

In this paper, based on data-reusing techniques we developed an algorithm that mixes some LMS and RLS characteristics. The proposed algorithm showed to converge to the same misadjustment produced by the LMS algorithm, and to converge as fast as the RLS algorithm. As in the LMS algorithm, the residual mean-square error can be controlled by a step-size parameter. The proposed algorithm presented complexity of $O(N^2)$, and was algebraically similar to the RLS algorithm, although it has different interpretation.

A perspective of this work includes a fast version implementation, an analysis of the relationship between the proposed algorithm and Kalman filters, a theoretical understanding of the step-size influence over the misadjustment, as well as the implementation of a forgetting factor that might extend the proposed algorithm to non-stationary environments.



(a)



(b)

Fig. 4. (a) Comparison of the proposed algorithm for $\mu = 0.075$ and the RLS algorithm for $\lambda = 1$. (b) The proposed algorithm for $\mu = 0.001$.

References

[1] S. Roy and J. J. Shynk, "Analysis of the data-reusing LMS algorithm," in *Proceedings of the 32nd Midwest Symposium on Circuits and Systems*, 1990, pp. 1127–1130.

[2] B. A. Schnaufer and W. K. Jenkins, "New data-reusing LMS algorithms for improved convergence," in *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, Nov. 1993, pp. 1584–1588.

[3] J. Benesty and T. Gänsler, "On data-reuse adaptive algorithms," in *Proc. 8th International Workshop on Acoustic Echo and Noise Control (IWAENC 2003)*, Kyoto, Japan, Sept. 2003, pp. 31–34.

[4] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*, Kluwer Academic Press, Norwell, MA, 2nd edition, 2002.

[5] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communications in Japan*, vol. 67-A, no. 5, pp. 19–27, 1984.

[6] R. A. Soni, W. K. Jenkins and K. Gallivan, "Projection methods for improved performance in FIR adaptive filters," in *Proceedings of the the 1997 IEEE Midwest Symposium on Circuits and Systems*, 1997, pp. 746–749.

[7] Y. Censor, P. P. B. Eggermont and D. Gordon, "Strong underrelaxation in Kaczmarz's method for inconsistent systems," *Numerische Mathematik*, vol. 41, pp. 83–92, 1983.

[8] R. A. Soni, *Projection Methods for Improved Performance in Adaptive Systems*, Ph.D. thesis, University of Illinois at Urbana-Champaign, IL, USA, 1998.

[9] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, Englewood Cliffs, NJ, 4nd edition, 2001.

[10] J. A. Apolinário Jr., M. L. R. de Campos and P. S. R. Diniz, "The binormalized data-reusing LMS algorithm," in *Proc. XV Simpósio Brasileiro de Telecomunicações*, Sept. 1997, pp. 77–80.