Article

An Optimized Differential Step-Size LMS Algorithm

Alexandru-George Rusu ^{1,2}, Silviu Ciochină ¹, Constantin Paleologu ^{1,*} and Jacob Benesty ³

¹ Department of Telecommunications, University Politehnica of Bucharest, 1-3, Iuliu Maniu Blvd., 061071 Bucharest, Romania

² Department of Research and Development, Rohde & Schwarz Topex, 014142 Bucharest, Romania

³ INRS-EMT, University of Quebec, Montreal, QC H5A 1K6, Canada

* Correspondence: pale@comm.pub.ro; Tel.: +40-21-402-4634

Received: 28 June 2019; Accepted: 21 July 2019; Published: 24 July 2019

Abstract: Adaptive algorithms with differential step-sizes (related to the filter coefficients) are well known in the literature, most frequently as “proportionate” algorithms. Usually, they are derived on a heuristic basis. In this paper, we introduce an algorithm resulting from an optimization criterion. Thereby, we obtain a benchmark algorithm and also another version with lower computational complexity, which is rigorously valid for less correlated input signals. Simulation results confirm the theory and outline the performance of the algorithms. Unfortunately, the good performance is obtained by an important increase in computational complexity. Nevertheless, the proposed algorithms could represent useful benchmarks in the field.

Keywords: adaptive filter; least-mean-square (LMS) algorithm; optimized differential step-size LMS algorithm; system identification; echo cancellation

1. Introduction

System identification is a widespread application, where an adaptive filter is used to identify the impulse response of an unknown system [1]. Using the same input signal, the main idea is to determine the coefficients of this filter so that its output generates a signal which is as close as possible to the output signal of the unknown system that must be identified. In many applications, the impulse response of the unknown system is characterized by a small percentage of coefficients with high magnitude, while the rest of them are almost zero. Consequently, the adaptive filter can exploit this sparseness feature in order to improve its overall performance (especially in terms of convergence rate and tracking). The basic idea is to “proportionate” the algorithm behavior, i.e., to update each coefficient of the filter independently of the others, by adjusting the adaptation step-size in proportion to the magnitude of the estimated filter coefficient. In this manner, the adaptation gain is “proportionately” redistributed among all the coefficients, emphasizing the large ones (in magnitude) in order to speed up their convergence. In other words, larger coefficients receive larger step-sizes, thus increasing the convergence rate of those coefficients. This has the effect that active coefficients are adjusted faster than the non-active ones (i.e., small or zero coefficients). On the other hand, the proportionate algorithms could be sensitive to the sparseness degree of the system, i.e., the convergence rate could be reduced when the impulse responses are not very sparse.

A typical application of these algorithms is the echo cancellation problem, with emphasis on the identification of sparse echo paths. Among the most popular such algorithms are proportionate normalized least-mean-square (PNLMS) [2], PNLMS++ [3], and improved PNLMS (IPNLMS) [4] algorithms. Furthermore, many algorithms have been designed in order to exploit the sparseness character of the unknown system [5–7], while their main focus is to improve both the convergence rate and tracking ability [8–10]. However, most of these algorithms have been designed in a heuristic

manner. For example, in the case of the celebrated PNLMS algorithm [2], the equations used to calculate the step-size control factors are not based on any optimization criteria but are designed in an ad hoc way. For this reason, after an initial fast convergence phase, the convergence rate of the PNLMS algorithm significantly slows down. Nevertheless, following the PNLMS idea, other more sophisticated algorithms working on the same principle have been proposed with the ultimate purpose to fully exploit sparsity of impulse responses. Among them, the IPNLMS algorithm [4] represents a benchmark solution, which is more robust to the sparseness degree of the impulse response.

A connection could be established between proportionate algorithms and variable step-size adaptive filters. In terms of their purpose, both categories adjust the step-size in order to enhance the overall performance. While proportionate filters aim to improve the convergence rate by increasing the adaptation step-size of the larger coefficients, variable step-size algorithms target the well-known compromise between the convergence rate and misadjustment [1]. In both cases, the adaptation control rules should rely on some specific optimization criteria.

Based on the initial work reported in [11,12], in this paper, we propose a least-mean-square (LMS) algorithm controlled by a diagonal matrix, $\mathbf{Y}$, instead of a usual scalar step-size. The elements of the matrix are adjusted at each iteration by considering the mean-squared deviation (MSD) minimization. Hence, following a stochastic model [13], we obtain an optimized differential step-size LMS (ODSS-LMS) algorithm, tailored by a state variable model. The theory has been driven on the model of an unknown system characterized by a time-varying behavior. The algorithm developed in [12], namely the optimized LMS algorithm with individual control factors (OLMS-ICF), represents only a particular case of the ODSS-LMS algorithm.

The motivation behind our approach can be summarized as follows. In many related works concerning optimized step-size algorithms, e.g., [14–18], the reference signal is considered the output of a time-invariant system corrupted by additive noise. However, in realistic scenarios, the system to be identified could be variable in time. For example, in acoustic echo cancellation, it can be assumed that the impulse response of the echo path is modeled by a time-varying system following a first-order Markov model [13]. In addition, it makes more sense to minimize the system misalignment (in terms of the MSD of the coefficients), instead of the classical error-based cost function. The reason is that, in real-world system identification problems, the goal of the adaptive filter is not to make the error signal goes to zero. The objective, instead, is to recover the additive noise (which corrupts the system's output) from the error signal of the adaptive filter, after this one converges to the true solution. For example, in echo cancellation, this “corrupting noise” comes from the near-end and could also contain the voice of the near-end talker. Consequently, it should be delivered (with little or no distortion) to the far-end, through the error signal.

This paper is structured as follows. In Section 2, we describe the system model. Furthermore, the autocorrelation matrix of the coefficients error is introduced in Section 3. We develop the ODSS-LMS algorithm in Section 4, as well as a simplified version for a white Gaussian input. Simulations of the proposed algorithms are shown in Section 5. Finally, conclusions are discussed in Section 6.

2. System Model

We begin with the traditional steepest descent update equation [11]:

$$\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mu E[\mathbf{x}(n)\alpha(n)]. \quad (1)$$

In Equation (1), we denote $\hat{\mathbf{w}}(n) = [\hat{w}_0(n) \ \hat{w}_1(n) \ \cdots \ \hat{w}_{L-1}(n)]^T$ as the adaptive filter (of length L), where the superscript T is the transpose operator, μ is the fixed step-size, $\mathbf{x}(n) = [x(n) \ x(n-1) \ \cdots \ x(n-L+1)]^T$ is a vector that contains the L most recent real-valued samples of the (zero-mean) input signal at the discrete time index n , $\alpha(n)$ is the a priori error signal, and $E(\cdot)$ is the statistical expectation. The goal of the adaptive filter is to estimate the impulse response

of the unknown system denoted as $\mathbf{w}(n) = [w_0(n) \ w_1(n) \ \cdots \ w_{L-1}(n)]^T$. Using the previous notation, the a priori error is expressed as $\alpha(n) = d(n) - \mathbf{x}^T(n)\hat{\mathbf{w}}(n-1)$, where the desired signal, $d(n) = \mathbf{x}^T(n)\mathbf{w}(n) + v(n)$, consists of the output of the unknown system corrupted by the measurement noise, $v(n)$, which is assumed to be a zero-mean Gaussian noise signal.

Then, by applying the stochastic gradient principle and substituting the fixed step-size with a diagonal matrix, $\mathbf{Y}(n)$, the coefficients of the estimated impulse response can be updated as follows:

$$\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{Y}(n)\mathbf{x}(n)\alpha(n). \quad (2)$$

Considering a stochastic system model described by $\mathbf{w}(n) = \mathbf{w}(n-1) + \delta_{\mathbf{w}}(n)$, where $\delta_{\mathbf{w}}(n)$ is a zero-mean Gaussian random vector, the goal consists of finding the diagonal matrix, $\mathbf{Y}(n)$, that minimizes at each iteration the MSD, $m(n) = E[\mathbf{c}^T(n)\mathbf{c}(n)]$, where $\mathbf{c}(n) = \hat{\mathbf{w}}(n) - \mathbf{w}(n)$ denotes the a posteriori misalignment. Thereby, the a posteriori error of the adaptive filter is

$$e(n) = d(n) - \mathbf{x}^T(n)\hat{\mathbf{w}}(n). \quad (3)$$

Substituting the desired signal and the a posteriori misalignment in Equation (3), the a posteriori error becomes $e(n) = v(n) - \mathbf{x}^T(n)\mathbf{c}(n)$. Proceeding in the same manner with the a priori error, we obtain $\alpha(n) = \mathbf{x}^T(n)[\delta_{\mathbf{w}}(n) - \mathbf{c}(n-1)] + v(n)$. Therefore, we can introduce (2) in the a posteriori misalignment to obtain the recursive relation:

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mathbf{w}(n-1) - \mathbf{w}(n) + \mathbf{Y}(n)\mathbf{x}(n)\alpha(n). \quad (4)$$

Furthermore, we can process Equation (4) as follows:

$$\begin{aligned} \mathbf{c}(n) &= \mathbf{c}(n-1) - \delta_{\mathbf{w}}(n) + \mathbf{Y}(n)\mathbf{x}(n)\alpha(n) \\ &= \mathbf{c}(n-1) - \delta_{\mathbf{w}}(n) + \mathbf{Y}(n)\mathbf{x}(n)\{\mathbf{x}^T(n)[\delta_{\mathbf{w}}(n) - \mathbf{c}(n-1)] + v(n)\} \\ &= [\mathbf{I}_L - \mathbf{Y}(n)\mathbf{x}(n)\mathbf{x}^T(n)][\mathbf{c}(n-1) - \delta_{\mathbf{w}}(n)] + \mathbf{Y}(n)\mathbf{x}(n)v(n), \end{aligned} \quad (5)$$

where $\mathbf{I}_L$ is the identity matrix of size $L \times L$.

3. Autocorrelation Matrix of the Coefficients Error

In this framework, we assume that the input signal is wide-sense stationary. Therefore, its autocorrelation matrix is given by $\mathbf{R} = E[\mathbf{x}(n)\mathbf{x}^T(n)]$. The autocorrelation matrix of the coefficients error is

$$\mathbf{R}_{\mathbf{c}}(n) \triangleq E[\mathbf{c}(n)\mathbf{c}^T(n)]. \quad (6)$$

In addition, we denote $\mathbf{\Gamma}(n) \triangleq \mathbf{R}_{\mathbf{c}}(n) + \sigma_w^2 \mathbf{I}_L$ and $\gamma(n) \triangleq \text{diag}[\mathbf{\Gamma}(n)]$, where σ_w^2 is the variance of the system noise and $\text{diag}(\cdot)$ defines a vector retaining only the main diagonal elements of $\mathbf{\Gamma}(n)$. Going forward, Equation (5) is developed by applying the ℓ_2 norm and the statistical expectation, so that we can introduce the notation:

$$m(n) = E[\|\mathbf{c}(n)\|_2^2] = \text{tr}[\mathbf{R}_{\mathbf{c}}(n)], \quad (7)$$

where $\text{tr}(\cdot)$ denotes the trace of a square matrix. Substituting (5) into (6), the autocorrelation matrix of the coefficients error results in

$$\begin{aligned} \mathbf{R}_c(n) = & E\{[\mathbf{I}_L - \mathbf{Y}(n)\mathbf{x}(n)\mathbf{x}^T(n)][\mathbf{c}(n-1) - \delta_w(n)] \\ & \times [\mathbf{c}^T(n-1) - \delta_w^T(n)][\mathbf{I}_L - \mathbf{x}(n)\mathbf{x}^T(n)\mathbf{Y}(n)] \\ & + \mathbf{Y}(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{Y}(n)v^2(n)\}. \end{aligned} \quad (8)$$

At this point, we consider that $\delta_w(n)$ is uncorrelated with the a posteriori misalignment at the discrete time index $n-1$ and with the input signal at the discrete time index n . In addition, we assume that $\mathbf{c}(n-1)$ and $\mathbf{x}(n)$ are uncorrelated. The zero-mean measurement noise, $v(n)$, is independent of the input signal, $\mathbf{x}(n)$, and of the system noise, $\delta_w(n)$. The variance of the additive noise is $\sigma_v^2 = E[v^2(n)]$. With these assumptions, the statistical expectation of the product term in (8) can be written as

$$\begin{aligned} & E\{[\mathbf{I}_L - \mathbf{Y}(n)\mathbf{x}(n)\mathbf{x}^T(n)][\mathbf{c}(n-1) - \delta_w(n)][\mathbf{c}^T(n-1) - \delta_w^T(n)][\mathbf{I}_L - \mathbf{x}(n)\mathbf{x}^T(n)\mathbf{Y}(n)]\} \\ & = \mathbf{R}_c(n-1) + \mathbf{I}_L\sigma_w^2 - [\mathbf{R}_c(n-1) + \mathbf{I}_L\sigma_w^2]\mathbf{R}\mathbf{Y}(n) - \mathbf{Y}(n)\mathbf{R}[\mathbf{R}_c(n-1) + \mathbf{I}_L\sigma_w^2] + \mathbf{Y}(n)\mathbf{Q}(n)\mathbf{Y}(n), \end{aligned} \quad (9)$$

where

$$\begin{aligned} \mathbf{Q}(n) & \triangleq E\{[\mathbf{x}(n)\mathbf{x}^T(n)][\mathbf{c}(n-1) - \delta_w(n)][\mathbf{c}^T(n-1) - \delta_w^T(n)][\mathbf{x}(n)\mathbf{x}^T(n)]\} \\ & = E[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{p}(n)\mathbf{p}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)] \\ & = E\{[\mathbf{p}^T(n)\mathbf{x}(n)]^2\mathbf{x}(n)\mathbf{x}^T(n)\} \\ & = E\left[\sum_{i,j=0}^{L-1} p_i(n)p_j(n)x(n-i)x(n-j)\hat{\mathbf{R}}(n)\right] \end{aligned} \quad (10)$$

and

$$\hat{\mathbf{R}}(n) = \mathbf{x}(n)\mathbf{x}^T(n), \quad (11)$$

$$\mathbf{p}(n) = \delta_w(n) - \mathbf{c}(n-1) = [p_0(n) \ p_1(n) \ \cdots \ p_{L-1}(n)]^T. \quad (12)$$

Including Equation (11) in (10), we can express the elements of the matrix $\mathbf{Q}(n)$ as

$$[\mathbf{Q}(n)]_{r,l} = \sum_{i,j=0}^{L-1} E[p_i(n)p_j(n)]E[x(n-i)x(n-j)x(n-r)x(n-l)], \quad (13)$$

for $r, l = 0, 1, \dots, L-1$. By applying the Gaussian moment factoring theorem [19], three terms can be defined as

$$\begin{aligned} [\mathbf{Q}_1(n)]_{r,l} & = \sum_{i,j=0}^{L-1} E[p_i(n)p_j(n)]x(i-j)x(r-l) \\ & = x(r-l) \sum_{i,j=0}^{L-1} [\mathbf{\Gamma}(n-1)]_{i,j}x(i-j) \\ & = \text{tr}[\mathbf{\Gamma}(n-1)\mathbf{R}][\mathbf{R}]_{r,l}, \end{aligned} \quad (14)$$

$$\begin{aligned}
[\mathbf{Q}_2(n)]_{r,l} &= \sum_{i,j=0}^{L-1} \mathbb{E}[p_i(n)p_j(n)]x(i-r)x(j-l) \\
&= \sum_{i=0}^{L-1} \mathbf{R}_{r,i} \sum_{j=0}^{L-1} [\Gamma(n-1)]_{i,j} \mathbf{R}_{j,l} \\
&= [\mathbf{R}\Gamma(n-1)\mathbf{R}]_{r,l},
\end{aligned} \tag{15}$$

and

$$\begin{aligned}
[\mathbf{Q}_3(n)]_{r,l} &= \sum_{i,j=0}^{L-1} \mathbb{E}[p_i(n)p_j(n)]x(i-l)x(j-r) \\
&= [\mathbf{Q}_2(n)]_{r,l},
\end{aligned} \tag{16}$$

with

$$\begin{aligned}
\mathbf{Q}(n) &= \mathbf{Q}_1(n) + \mathbf{Q}_2(n) + \mathbf{Q}_3(n) \\
&= 2\mathbf{R}\Gamma(n-1)\mathbf{R} + \mathbf{R}\text{tr}[\Gamma(n-1)\mathbf{R}].
\end{aligned} \tag{17}$$

Finally, the statistical expectation of the last term in Equation (8) is

$$E[\mathbf{Y}(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{Y}(n)v^2(n)] = \mathbf{Y}(n)\mathbf{R}\mathbf{Y}(n)\sigma_v^2. \tag{18}$$

Thus, we can summarize the autocorrelation matrix of the coefficients error as

$$\begin{aligned}
\mathbf{R}_c(n) &= \Gamma(n-1) - \Gamma(n-1)\mathbf{R}\mathbf{Y}(n) - \mathbf{Y}(n)\mathbf{R}\Gamma(n-1) \\
&\quad + \mathbf{Y}(n)\{2\mathbf{R}\Gamma(n-1)\mathbf{R} + \mathbf{R}\text{tr}[\Gamma(n-1)\mathbf{R}]\}\mathbf{Y}(n) \\
&\quad + \mathbf{Y}(n)\mathbf{R}\mathbf{Y}(n)\sigma_v^2.
\end{aligned} \tag{19}$$

4. ODSS-LMS Algorithm

Let us define $\mathbf{Y}(n) = \text{mdia}[\mathbf{y}(n)]$, where $\text{mdia}(\cdot)$ denotes a diagonal matrix, having on its main diagonal the elements of the vector $\mathbf{y}(n)$. If the input argument is a matrix $\mathbf{A}$, then $\mathbf{B} = \text{mdia}(\mathbf{A})$ returns a diagonal matrix $\mathbf{B}$, retaining only the elements of the main diagonal of $\mathbf{A}$.

4.1. Minimum MSD Value

Replacing $\Gamma(n-1)$ and using (19), the MSD can be rewritten as

$$\begin{aligned}
m(n) &= \text{tr} \left\{ [\mathbf{R}_c(n-1) + \sigma_w^2 \mathbf{I}_L] \{ \mathbf{I}_L - \mathbf{R}\mathbf{Y}(n) - \mathbf{Y}(n)\mathbf{R} + \sigma_x^2 \mathbf{R}\text{tr}[\mathbf{Y}^2(n)] + 2\mathbf{R}\mathbf{Y}^2(n)\mathbf{R} \} \right. \\
&\quad \left. + \text{tr} [\sigma_v^2 \mathbf{R}\mathbf{Y}^T(n)\mathbf{Y}(n)] \right\},
\end{aligned} \tag{20}$$

where σ_x^2 is the variance of the input signal, $x(n)$. The last term in (20) can be easily computed as

$$\text{tr} [\sigma_v^2 \mathbf{R}\mathbf{Y}^T(n)\mathbf{Y}(n)] = \sigma_x^2 \sigma_v^2 \mathbf{y}^T(n)\mathbf{y}(n). \tag{21}$$

Therefore, the product term of (20) results in

$$\begin{aligned}
&\text{tr} \left\{ \mathbf{R}_c(n-1) \{ \mathbf{I}_L - \mathbf{R}\mathbf{Y}(n) - \mathbf{Y}(n)\mathbf{R} + \sigma_x^2 \mathbf{R}\text{tr}[\mathbf{Y}^2(n)] + 2\mathbf{R}\mathbf{Y}^2(n)\mathbf{R} \} \right\} \\
&= m(n-1) - 2\mathbf{y}^T(n)\text{diag}[\mathbf{R}\mathbf{R}_c(n-1)] + \sigma_x^2 \text{tr}[\mathbf{R}_c(n-1)\mathbf{R}]\mathbf{y}^T(n)\mathbf{y}(n) \\
&\quad + 2\mathbf{y}^T(n)\text{mdia}[\mathbf{R}\mathbf{R}_c(n-1)\mathbf{R}]\mathbf{y}(n)
\end{aligned} \tag{22}$$

and

$$\begin{aligned} & \text{tr} \left\{ \sigma_w^2 \{ \mathbf{I}_L - \mathbf{R}\mathbf{Y}(n) - \mathbf{Y}(n)\mathbf{R} + \sigma_x^2 \mathbf{R} \text{tr}[\mathbf{Y}^2(n)] + 2\mathbf{R}\mathbf{Y}^2(n)\mathbf{R} \} \right\} \\ & = \sigma_w^2 \{ L - 2\mathbf{y}^T(n)\mathbf{r} + L\sigma_x^4 \mathbf{y}^T(n)\mathbf{y}(n) + 2\mathbf{y}^T(n) \text{mdia}[\mathbf{R}^2]\mathbf{y}(n) \}, \end{aligned} \quad (23)$$

with $\mathbf{r} \triangleq \text{diag}(\mathbf{R})$. Based on (20), (21), and (23), the MSD becomes

$$\begin{aligned} m(n) &= m(n-1) - 2\mathbf{y}^T(n) \text{diag}[\mathbf{R}\mathbf{R}_c(n-1)] + \sigma_x^2 \text{tr}[\mathbf{R}_c(n-1)\mathbf{R}]\mathbf{y}^T(n)\mathbf{y}(n) \\ & \quad + 2\mathbf{y}^T(n) \text{mdia}[\mathbf{R}\mathbf{R}_c(n-1)\mathbf{R}]\mathbf{y}(n) \\ & \quad + \sigma_w^2 \{ L - 2\mathbf{y}^T(n)\mathbf{r} + L\sigma_x^4 \mathbf{y}^T(n)\mathbf{y}(n) + 2\mathbf{y}^T(n) \text{mdia}[\mathbf{R}^2]\mathbf{y}(n) \} \\ & \quad + \sigma_x^2 \sigma_v^2 \mathbf{y}^T(n)\mathbf{y}(n). \end{aligned} \quad (24)$$

4.2. Optimum Step-Size Derivation

Taking the gradient of MSD with respect to $\mathbf{y}(n)$, it results in

$$\begin{aligned} \nabla_{\mathbf{y}(n)}[m(n)] &= -2\text{diag}[\mathbf{R}\mathbf{R}_c(n-1)] + 2\sigma_x^2 \text{tr}[\mathbf{R}_c(n-1)\mathbf{R}]\mathbf{y}(n) + 4\text{mdia}[\mathbf{R}\mathbf{R}_c(n-1)\mathbf{R}]\mathbf{y}(n) \\ & \quad + \sigma_w^2 [-2\mathbf{r} + 2L\sigma_x^4 \mathbf{y}(n) + 4\text{mdia}(\mathbf{R}^2)\mathbf{y}] + 2\sigma_x^2 \sigma_v^2 \mathbf{y}(n). \end{aligned} \quad (25)$$

Solving the quadratic form of (25), we obtain the general optimal solution:

$$\mathbf{y}_o(n) = \left\{ \sigma_x^2 \mathbf{I}_L \{ \text{tr}[\mathbf{\Gamma}(n-1)\mathbf{R}] + \sigma_v^2 \} + 2\text{mdia}[\mathbf{R}\mathbf{\Gamma}(n-1)\mathbf{R}] \right\}^{-1} \text{diag}[\mathbf{R}\mathbf{\Gamma}(n-1)]. \quad (26)$$

In brief, (19), (24), and (26) represent the main equations of the optimized differential least-mean-square “general” (ODSS-LMS-G) algorithm.

4.3. Simplified Version

At this stage, we can define the main equations for the optimized differential least-mean-square “white” (ODSS-LMS-W) algorithm, in the scenario of a white Gaussian noise signal as input. This algorithm is similar to the OLMS-ICF algorithm presented in [12], which was developed based on the same simplified assumption.

Considering that $\mathbf{R} = \sigma_x^2 \mathbf{I}_L$, we can rewrite (19), (24), and (26) as

$$\begin{aligned} \mathbf{R}_c(n) &= \mathbf{\Gamma}(n-1) - \sigma_x^2 \mathbf{\Gamma}(n-1)\mathbf{Y}(n) - \sigma_x^2 \mathbf{Y}(n)\mathbf{\Gamma}(n-1) \\ & \quad + \sigma_x^4 \mathbf{Y}(n) \{ 2\mathbf{\Gamma}(n-1) + \mathbf{I}_L \text{tr}[\mathbf{\Gamma}(n-1)] \} \mathbf{Y}(n) + \sigma_x^2 \sigma_v^2 \mathbf{Y}^2(n), \end{aligned} \quad (27)$$

$$\begin{aligned} m(n) &= [m(n-1) + L\sigma_w^2] [1 + \sigma_x^4 \mathbf{y}^T(n)\mathbf{y}(n)] - 2\sigma_x^2 \gamma^T(n-1)\mathbf{y}(n) \\ & \quad + 2\sigma_x^4 \mathbf{y}^T(n) \text{mdia}[\gamma(n-1)]\mathbf{y}(n) + \sigma_v^2 \sigma_x^2 \mathbf{y}^T(n)\mathbf{y}(n), \end{aligned} \quad (28)$$

and

$$\mathbf{y}_o(n) = \left\{ \sigma_x^2 \mathbf{I}_L [m(n-1) + \sigma_w^2 (L+2)] + 2\sigma_x^2 \text{mdia}[\mathbf{R}_c(n-1)] + \sigma_v^2 \mathbf{I}_L \right\}^{-1} \gamma(n-1). \quad (29)$$

Based on (27), we can define the recursive equation of $\gamma(n)$ as

$$\begin{aligned} \gamma(n) &= \gamma(n-1) + \sigma_w^2 \mathbf{1}_{L \times 1} + 2\sigma_x^2 [\sigma_x^2 \mathbf{y}_o(n) - \mathbf{1}_{L \times 1}] \odot \gamma(n-1) \odot \mathbf{y}_o(n) \\ & \quad + \sigma_x^2 \{ \sigma_x^2 \text{tr}[\mathbf{\Gamma}(n-1)] + \sigma_v^2 \} \mathbf{y}_o(n) \odot \mathbf{y}_o(n), \end{aligned} \quad (30)$$

where $\mathbf{1}_{L \times 1}$ denotes a vector with all its elements equal to one and $\odot$ represents the Hadamard product. In order to verify the stability condition, let us define

$$\mathbf{M}(n) \triangleq \sigma_x^2 \mathbf{I}_L [m(n-1) + \sigma_w^2 (L+2)] + 2\sigma_x^2 \text{mdia}[\mathbf{R}_c(n-1)] + \sigma_v^2 \mathbf{I}_L, \quad (31)$$

which leads to $\mathbf{y}_o(n) = \mathbf{M}^{-1}(n) \gamma(n-1)$.

If the system is assumed to be time-invariant, i.e., $\sigma_w^2 = 0$, then

$$\begin{aligned} m(n) &= m(n-1) + \mathbf{y}^T(n) [\sigma_x^4 m(n-1) \mathbf{I}_L + 2\sigma_x^4 \mathbf{R}_c(n-1) + \mathbf{I}_L \sigma_x^2 \sigma_v^2] \mathbf{y}(n) - 2\sigma_x^2 \gamma^T(n-1) \mathbf{y}(n) \\ &= m(n-1) + \sigma_x^2 \mathbf{y}^T(n) \mathbf{M}(n) \mathbf{y}(n) - 2\sigma_x^2 \gamma^T(n-1) \mathbf{y}(n) \\ &= m(n-1) - \sigma_x^2 \gamma^T(n-1) \mathbf{M}^{-1}(n) \gamma(n-1). \end{aligned} \quad (32)$$

Since $\mathbf{M}(n)$ is a diagonal positive definite matrix, $m(n) < m(n-1)$, therefore,

$$m(n) = \sum_{l=0}^{L-1} \gamma_l(n-1) [1 - \sigma_x^2 \mu_l(n-1) \gamma_l(n-1)], \quad (33)$$

where $\gamma_l(n)$ denotes the l th element of the vector $\gamma(n)$ and

$$\mu_l(n-1) \triangleq \frac{1}{m(n-1) \sigma_x^2 + 2\sigma_x^2 \gamma_l(n-1) + \sigma_v^2}. \quad (34)$$

Based on (33) and (34), we can outline that

$$\frac{\sigma_x^2 \gamma_l(n-1)}{2\sigma_x^2 \gamma_l(n-1) + \sigma_v^2 + \sigma_x^2 \sum_{i=0}^{L-1} \gamma_i(n-1)} < 1, \quad (35)$$

which guarantees the convergence. At the same time, the stability condition is verified, i.e., $0 < m(n) < m(n-1)$.

4.4. Practical Considerations

The ODSS-LMS-G and ODSS-LMS-W algorithms are depicted in Algorithms 1 and 2, respectively. From a practical point of view, the variance σ_x^2 of the input signal could be estimated with a 1-pole exponential filter, or using $\sigma_x^2 = (1/L) \mathbf{x}^T(n) \mathbf{x}(n)$. In practice, the autocorrelation matrix of the input signal is not known and it can be estimated with a weighting window as $\mathbf{R}_x(n) = \lambda \mathbf{R}_x(n-1) + (1-\lambda) \mathbf{x}(n) \mathbf{x}^T(n)$, where $\lambda = 1 - 1/(PL)$, with $P > 1$. The noise power σ_v^2 could be estimated in different ways as in [20,21]. The influence of the system noise variance σ_w^2 is analyzed in Section 5. Moreover, in order to obtain accurate estimates for some parameters, we suggest using an initialization algorithm such as the normalized LMS (NLMS) [11]. With the purpose to estimate the matrix $\mathbf{R}_c(n)$ in the initial stage of the algorithm, we suppose that $\mathbf{c}(n) \approx \hat{\mathbf{w}}(n) - \hat{\mathbf{w}}(n-1)$, so that we can approximate it with the update term $\mathbf{u}(n)$ defined in Algorithms 1 and 2.

In terms of computational complexity, the matrix operations lead to a high computational complexity. Therewith, the matrix inversion operation is involved and contributes to the complexity increase. Taking into account once the terms that appear several times in multiple equations (i.e., $\Gamma(n-1) \mathbf{R}_x(n)$, $\mathbf{R}_x(n) \Gamma(n-1)$), the computational complexity is summarized as follows. For the ODSS-LMS-G algorithm, the computation of the $\mathbf{M}(n)$ matrix requires $2L^3 + 2L + 1$ multiplications and $2L^3 - 2L^2 + 2L$ additions, while its inversion consists of L divisions of the form $1/x$. Then, computation of the differential step-size $\mathbf{y}_o(n)$ consists of $L^3 + L$ multiplications and $L^3 - L^2$ additions. The update term $\mathbf{u}(n)$ contributes with $2L$ multiplications. Thereto, because the $\mathbf{Y}(n)$ matrix is diagonal, the computation of the $\mathbf{R}_c(n)$ matrix requires $8L$ multiplications and $5L$ additions. In addition, the computation of $\Gamma(n)$ matrix requires L additions and the estimation

of the $\mathbf{R}_x(n)$ matrix imposes $2L^2 + 2L - 1$ multiplications and L^2 additions. In case of the ODSS-LMS-W algorithm, several matrix operations disappear and the computational complexity decreases. Thus, the computation of the $\mathbf{M}(n)$ matrix requires only $2L + 1$ multiplications and $2L + 1$ additions. While $\mathbf{M}(n)$ matrix is diagonal, the computational complexity of its inversion involves L divisions. Therefore, the differential step-size $\mathbf{y}_o(n)$ requires only L multiplications. In addition, the computation of the vector $\gamma(n)$ contributes with $5L + 2$ multiplications and $5L$ additions. Furthermore, the MSD is obtained for L additions, while the update term $\mathbf{u}(n)$ adds $2L$ multiplications. Some equations are required in both algorithms. The coefficients update equation requires L additions and the a priori error computations consist of L multiplications and L additions. Last but not least, the initialization process of both algorithms involves $3L^2 + 2L + 2$ multiplications, $L^2 + L + 3$ additions, and one division. In brief, without the complexity of the initialization process, the ODSS-LMS-G algorithm involves $3L^3 + 2L^2 + 16L$ multiplications, $3L^3 - 2L^2 + 9L$ additions, and L divisions, while ODSS-LMS-W consists of $10L + 3$ multiplications, $10L + 1$ additions, and L divisions. Note that the computational complexity analysis is based on the standard way of inverting and multiplying matrices, so that the computational complexity of the ODSS-LMS-G algorithm is $O(L^3)$. One of the first approaches regarding the reduction of matrix multiplication computational complexity is the Strassen algorithm [22], which is slightly better than the standard matrix multiplication algorithm, with a computational complexity of $O(L^{2.807})$. Going forward, the Coppersmith–Winograd algorithm [23] could be used to reduce the matrix multiplication computational complexity to $O(L^{2.375})$.

Algorithm 1: ODSS-LMS-G algorithm.

Initialization:

- $\hat{\mathbf{w}}(0) = \mathbf{0}_{L \times 1}$
 - $\mathbf{R}_c(0) = c \mathbf{I}_L$, where c is a small positive constant
 - $\mathbf{R}_x(0) = \mathbf{0}_{L \times L}$
 - $\Gamma(0) = \mathbf{R}_c(0) + \sigma_w^2 \mathbf{I}_L$
- Parameters $\sigma_x^2, \sigma_v^2, \sigma_w^2$, known or estimated
 $\lambda = 1 - \frac{1}{PL}$, with $P \geq 1$

For time index $n = 1, 2, \dots$:

- $\alpha(n) = d(n) - \mathbf{x}^T(n) \hat{\mathbf{w}}(n-1)$
- $\mathbf{R}_x(n) = \lambda \mathbf{R}_x(n-1) + (1 - \lambda) \mathbf{x}(n) \mathbf{x}^T(n)$

If time index $n < KL$, with $K \geq 1$:

- $\mu(n) = \frac{\mu}{\mathbf{x}^T(n) \mathbf{x}(n) + \delta}$ (i.e., step-size of the NLMS algorithm)
- $\mathbf{u}(n) = \mu(n) \mathbf{x}(n) \alpha(n)$
- $\mathbf{R}_c(n) = \lambda \mathbf{R}_c(n-1) + (1 - \lambda) \mathbf{u}(n) \mathbf{u}^T(n)$

else:

- $\mathbf{M}(n) = \sigma_x^2 \mathbf{I}_L \{ \text{tr}[\Gamma(n-1) \mathbf{R}_x(n)] + \sigma_v^2 \}$
 $+ 2 \text{mdia}[\mathbf{R}_x(n) \Gamma(n-1) \mathbf{R}_x(n)]$
 - $\mathbf{y}_o(n) = \mathbf{M}^{-1}(n) \text{diag}[\mathbf{R}_x(n) \Gamma(n-1)]$
 - $\mathbf{u}(n) = \mathbf{y}_o(n) \odot \mathbf{x}(n) \alpha(n)$
 - $\mathbf{Y}(n) = \text{mdia}[\mathbf{y}_o(n)]$
 - $\mathbf{R}_c(n) = \Gamma(n-1) - \Gamma(n-1) \mathbf{R}_x(n) \mathbf{Y}(n)$
 $- \mathbf{Y}(n) \mathbf{R}_x(n) \Gamma(n-1) + \mathbf{Y}(n) \{ 2 \mathbf{R}_x(n) \Gamma(n-1) \mathbf{R}_x(n)$
 $+ \mathbf{R}_x(n) \text{tr}[\Gamma(n-1) \mathbf{R}_x(n)] \} \mathbf{Y}(n) + \mathbf{Y}(n) \mathbf{R}_x(n) \mathbf{Y}(n) \sigma_v^2$
 - $\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{u}(n)$
 - $\Gamma(n) = \mathbf{R}_c(n) + \sigma_w^2 \mathbf{I}_L$
-

Finally, the overall computational complexity of the NLMS [1], IPNLMS [4], and JO-NLMS [24] algorithms is summarized for comparison. The NLMS algorithm consists of $2L + 2$ multiplications, $2L + 3$ additions, and 1 division, while the IPNLMS algorithm involves $5L + 2$ multiplications, $5L$ additions, and two divisions. The JO-NLMS algorithm has a computational complexity similar to that of the NLMS algorithm, which is based on $2L + 3$ multiplications, $2L + 4$ additions, and one division. Notice that the computational complexity of these algorithms has been reported by considering σ_x^2 , σ_v^2 , and σ_w^2 parameters known, while their estimation depends on the implementation (e.g., in literature, different estimation methods are available). Of course, these algorithms do not involve matrix operations and the computational complexity is $O(L)$. At this point, it can be highlighted that the computational complexity of the ODSS-LMS-W algorithm is comparable with the complexity of the NLMS, IPNLMS, or JO-NLMS algorithm.

Even if the proposed algorithms present a high computational complexity, they are more affined to “benchmark” type algorithms and their performance is outlined in Section 5.

Algorithm 2: ODSS-LMS-W algorithm.

Initialization:

- $\hat{\mathbf{w}}(0) = \mathbf{0}_{L \times 1}$
- $m(0) = \epsilon > 0$
- $\gamma(0) = c \mathbf{1}_{L \times 1}$, where c is a small positive constant
- $\mathbf{R}_c(0) = c \mathbf{I}_L$
- $\mathbf{\Gamma}(0) = \mathbf{R}_c(0) + \sigma_w^2 \mathbf{I}_L$

Parameters $\sigma_x^2, \sigma_v^2, \sigma_w^2$, known or estimated

$$\lambda = 1 - \frac{1}{PL}, \text{ with } P \geq 1$$

For time index $n = 1, 2, \dots$:

- $\alpha(n) = d(n) - \mathbf{x}^T(n) \hat{\mathbf{w}}(n-1)$

If time index $n < KL$, with $K \geq 1$:

- $\mu(n) = \frac{\mu}{\mathbf{x}^T(n) \mathbf{x}(n) + \delta}$ (i.e., step-size of the NLMS algorithm)
- $\mathbf{u}(n) = \mu(n) \mathbf{x}(n) \alpha(n)$
- $\mathbf{R}_c(n) = \lambda \mathbf{R}_c(n-1) + (1 - \lambda) \mathbf{u}(n) \mathbf{u}^T(n)$
- $\mathbf{\Gamma}(n) = \mathbf{R}_c(n) + \sigma_w^2 \mathbf{I}_L$

else:

- $\mathbf{M}(n) = \sigma_x^2 \mathbf{I}_L [m(n-1) + \sigma_w^2 (L+2)]$
 $+ 2\sigma_x^2 \text{mdia}[\gamma(n-1)] + \sigma_v^2 \mathbf{I}_L$
- $\mathbf{y}_o(n) = \mathbf{M}^{-1}(n) \gamma(n-1)$
- $\gamma(n) = \gamma(n-1) + \sigma_w^2 \mathbf{1}_{L \times 1}$
 $+ 2\sigma_x^2 [\sigma_x^2 \mathbf{y}_o(n) - \mathbf{1}_{L \times 1}] \odot \gamma(n-1) \odot \mathbf{y}_o(n)$
 $+ \sigma_x^2 \{\sigma_x^2 \text{tr}[\mathbf{\Gamma}(n-1)] + \sigma_v^2\} \mathbf{y}_o(n) \odot \mathbf{y}_o(n)$
- $\mathbf{\Gamma}(n) = \text{mdia}[\gamma(n)]$
- $m(n) = \text{tr}[\mathbf{\Gamma}(n)] - L\sigma_w^2$
- $\mathbf{u}(n) = \mathbf{y}_o(n) \odot \mathbf{x}(n) \alpha(n)$

- $\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{u}(n)$
-

5. Simulation Results

The experiments are performed in the context of echo cancellation configuration using an acoustic impulse response and a sparse echo path from ITU-T G168 Recommendation [25] (padded with zeros). The echo paths have 512 coefficients and the same number is used for the adaptive filter, i.e., $L = 512$. The sparseness measure based on the ℓ_1 and ℓ_2 norms [26] of the acoustic

echo path is $\xi_{12} = 0.6130$, while for the network echo path it is $\xi_{12} = 0.8969$. The impulse responses used in our simulations are shown in Figure 1. In these scenarios, the sampling rate is 8 kHz. In this context, the unknown system is the echo path and the adaptive filter should estimate it. The output signal is corrupted by an independent white Gaussian noise with the signal-to-noise ratio (SNR) set to $\text{SNR} = \sigma_y^2 / \sigma_v^2 = 20$ dB, where $\sigma_y^2 = E[y^2(n)]$. We assume that the power of this noise, σ_v^2 , is known. The input signal, $x(n)$, is either a white Gaussian noise, an AR(1) process generated by filtering a white Gaussian noise through a first-order system $1/(1 - \beta z^{-1})$, with $\beta = 0.8, 0.9$, or a speech sequence. The performance measure is the normalized misalignment in dB, defined as $20\log_{10}[\|\mathbf{w}(n) - \hat{\mathbf{w}}(n)\|_2 / \|\mathbf{w}(n)\|_2]$. The proposed algorithms are compared with the IPNLMS algorithm [4] (since, in general, it outperforms the NLMS algorithm) and also with the JO-NLMS algorithm [24] (using constant values of the parameter σ_w^2).

Figure 1. Echo paths used in simulations. (a) acoustic echo path and (b) network echo path (which is adapted from G168 Recommendation [25].)

In the first experiment, the acoustic echo path is involved. The system is considered time-invariant with $\sigma_w^2 = 0$. The ODSS-LMS-W and ODSS-LMS-G algorithms are compared with the IPNLMS algorithm in Figure 2, where the input signal is a white Gaussian noise. Thus, we confirm that relations (19), (24), and (26) are equivalent to (27), (28), and (29), respectively, in case of white Gaussian noise as an input signal. For the same convergence rate as the IPNLMS algorithm, the normalized misalignment of the proposed algorithms is definitely much lower.

Figure 2. Normalized misalignment of the IPNLMS, ODSS-LMS-G, and ODSS-LMS-W algorithms, where $\sigma_w^2 = 0$, the input signal is a white Gaussian noise, $L = 512$, and SNR = 20 dB. The unknown system is the acoustic echo path.

In the context of acoustic echo cancellation, the tracking capability of the algorithm is an important aspect. Unfortunately, the acoustic path can rapidly change during the communication. Therefore, in Figure 3, the ODSS-LMS-G algorithm is analyzed for different values of σ_w^2 in a scenario where the input signal is a white Gaussian noise. The echo path change has been simulated by shifting the impulse response to the right (by 10 samples) in the middle of the experiment. In this scenario, the importance of the variance σ_w^2 is highlighted. The misalignment is much lower for low values of σ_w^2 and is dimmed for high values of this parameter (e.g., $\sigma_w^2 = 10^{-10}$). Moreover, the variance of system noise indicates the need of a compromise between low misalignment and tracking capability (e.g., for $\sigma_w^2 = 0$, the ODSS-LMS-G algorithm tracks the optimal solution in a slow manner).

Figure 3. Normalized misalignment of the IPNLMS and ODSS-LMS-G algorithms for different values of σ_w^2 . Echo path changes at time 10 s. The input signal is a white Gaussian noise, $L = 512$, and SNR = 20 dB. The unknown system is the acoustic echo path.

Furthermore, we analyze the behavior of the ODSS-LMS-G algorithm for the identification of the sparse network echo path. Figure 4 outlines that the convergence rate is much faster than in the acoustic echo cancellation scenario, and the normalized misalignment decreases with at least 12 dB for all values of σ_w^2 as well.

Beyond this, the simulations are performed considering a stationary correlated input signal. In the context of acoustic echo cancellation configuration, Figure 5 shows the performance of the IPNLMS, JO-NLMS, ODSS-LMS-G, and ODSS-LMS-W algorithms in a scenario where the input signal is an AR(1) process, with $\beta = 0.9$. It can be observed that, for high values of σ_w^2 , it takes control in Equations (19), (24), (26) and (27)–(29) through $\Gamma(n)$, and both ODSS-LMS algorithms behave similarly.

Figure 4. Normalized misalignment of the IPNLMS and ODSS-LMS-G algorithms for different values of σ_w^2 . Echo path changes at time 10 s. The input signal is a white Gaussian noise, $L = 512$, and SNR = 20 dB. The unknown system is the network echo path.

Figure 5. Normalized misalignment of the IPNLMS, JO-NLMS, ODSS-LMS-G, and ODSS-LMS-W algorithms for different values of σ_w^2 . The input signal is an AR(1) process, with $\beta = 0.9$, $L = 512$, and SNR = 20 dB. The unknown system is the acoustic echo path.

Next, the ODSS-LMS-G and ODSS-LMS-W algorithms are analyzed for different values of σ_w^2 in a tracking scenario, where the input signal is an AR(1) process, with $\beta = 0.8$. In Figure 6,

the unknown system is represented by the sparse network path. The echo path changes in the same manner as in Figure 3. Up to a certain point, the ODSS-LMS-G algorithm deals better with correlated signal as input. We can remark on the lower normalized misalignment in case of ODSS-LMS-G algorithm for low values of σ_w^2 , while the tracking capability is affected due to the lack of the initial estimation process. Even so, the tracking could be improved if the variance of the system noise is increased.

In Figure 7, we compare the IPNLMS, ODSS-LMS-G, and ODSS-LMS-W algorithms for different values of σ_w^2 in an acoustic echo cancellation scenario, where the input signal is an AR(1) process with $\beta = 0.9$. As in the scenario from Figure 6, the effect of σ_w^2 is attenuated with its increase. Therefore, the need of a variable σ_w^2 parameter arises. Low values of σ_w^2 will lead to a good compromise between fast initial convergence and low misalignment, while high values will keep alive the tracking capability. This issue will be addressed in future works, by targeting a variable parameter to this purpose.

Figure 6. Normalized misalignment of the IPNLMS, ODSS-LMS-G, and ODSS-LMS-W algorithms for different values of σ_w^2 . Echo path changes at time 10 s. The input signal is an AR(1) process, with $\beta = 0.8$, $L = 512$, and SNR = 20 dB. The unknown system is the network echo path.

Figure 7. Normalized misalignment of the IPNLMS, ODSS-LMS-G, and ODSS-LMS-W algorithms for different values of σ_w^2 . Echo path changes at time 10 s. The input signal is an AR(1) process, with $\beta = 0.9$, $L = 512$, and SNR = 20 dB. The unknown system is the acoustic echo path.

Finally, in Figure 8, we compare the ODSS-LMS algorithms in a network echo cancellation scenario, where the input signal is a speech sequence. Again, we remark on the good performance of the ODSS-LMS-G algorithm with a highly correlated (and nonstationary) input signal.

Figure 8. Normalized misalignment of the IPNLMS, JO-NLMS, ODSS-LMS-G, and ODSS-LMS-W algorithms. The input signal is a speech sequence, $L = 512$, and $\text{SNR} = 20$ dB. The unknown system is the network echo path.

6. Conclusions

Starting from the LMS algorithm, we developed an optimized differential step-size algorithm in order to update the adaptive filter coefficients. Our approach is developed in the context of a state-variable model and follows an optimization criterion based on the minimization of MSD. We derived a general algorithm valid for any input signal, ODSS-LMS-G, together with a simplified version for white Gaussian input signal, ODSS-LMS-W. The ODSS-LMS-G algorithm presents a high complexity (e.g., $O(L^3)$) due to matrix manipulations, while the complexity of the ODSS-LMS-W algorithm is $O(L)$. Nevertheless, they can be considered as benchmark versions for performance analysis. Their performance is highlighted in Section 5, emphasizing the features of the optimized step-size. An important parameter of the algorithm is σ_w^2 , which should be tuned to address the compromise between low misalignment and fast tracking. In this context, it would be useful to find a variable parameter. Experiments demonstrate that the ODSS-LMS-G algorithm can be a benchmark in system identification, while the ODSS-LMS-W algorithm could be a possible candidate for field-programmable gate array (FPGA) applications, if the input signal is less correlated.

Author Contributions: Conceptualization, A.-G.R.; Formal analysis, S.C.; Software, C.P.; Methodology, J.B.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Widrow, B. *Least-Mean-Square Adaptive Filters*; Haykin, S.S., Widrow, B., Eds.; Wiley: Hoboken, NJ, USA, 2003.
2. Duttweiler, D.L. Proportionate normalized least-mean-squares adaptation in echo cancelers. *IEEE Trans. Speech Audio Process.* **2000**, *8*, 508–518. [[CrossRef](#)]
3. Gay, S.L. An efficient, fast converging adaptive filter for network echo cancellation. In Proceedings of the Conference Record of Thirty-Second Asilomar Conference on Signals, Systems and Computers (Cat. No.98CH36284), Pacific Grove, CA, USA, 1–4 November 1998; pp. 394–398.

4. Benesty, J.; Gay, S.L. An improved PNLMS algorithm. In Proceedings of the 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing, Orlando, FL, USA, 13–17 May 2002.
5. Deng, H.; Doroslovački, M. Proportionate adaptive algorithms for network echo cancellation. *IEEE Trans. Signal Process.* **2006**, *54*, 1794–1803. [[CrossRef](#)]
6. Das Chagas de Souza, F.; Tobias, O.J.; Seara, R.; Morgan, D.R. A PNLMS algorithm with individual activation factors. *IEEE Trans. Signal Process.* **2010**, *58*, 2036–2047. [[CrossRef](#)]
7. Liu, J.; Grant, S.L. Proportionate adaptive filtering for block-sparse system identification. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2016**, *24*, 623–630. [[CrossRef](#)]
8. Gu, Y.; Jin, J.; Mei, S. ℓ_0 norm constraint LMS algorithm for sparse system identification. *IEEE Signal Process. Lett.* **2009**, *16*, 774–777.
9. Loganathan, P.; Khong, A.W.H.; Naylor, P.A. A class of sparseness-controlled algorithms for echo cancellation. *IEEE Trans. Audio Speech Lang. Process.* **2009**, *17*, 1591–1601. [[CrossRef](#)]
10. Li, Y.; Wang, Y.; Sun, L. A proportionate normalized maximum correntropy criterion algorithm with correntropy induced metric constraint for identifying sparse systems. *Symmetry* **2018**, *10*, 683. [[CrossRef](#)]
11. Rusu, A.G.; Ciochină, S.; Paleologu, C. On the step-size optimization of the LMS algorithm. In Proceedings of the 42nd International Conference on Telecommunications and Signal Processing, Budapest, Hungary, 1–3 July 2019; pp. 168–173.
12. Ciochină, S.; Paleologu, C.; Benesty, J.; Grant, S.L.; Anghel, A. A family of optimized LMS-based algorithms for system identification. In Proceedings of the 2016 24th European Signal Processing Conference (EUSIPCO), Budapest, Hungary, 29 August–2 September 2016; pp. 1803–1807.
13. Enzner, G.; Buchner, H.; Favrot, A.; Kuech, F. Acoustic echo control. In *Academic Press Library in Signal Processing*; Academic Press: Cambridge, MA, USA, 2014; Volume 4, pp. 807–877.
14. Shin, H.-C.; Sayed, A.H.; Song, W.-J. Variable step-size NLMS and affine projection algorithms *IEEE Signal Process. Lett.* **2004**, *11*, 132–135.
15. Benesty, J.; Rey, H.; Rey Vega, L.; Tressens, S. A nonparametric VSS-NLMS algorithm. *IEEE Signal Process. Lett.* **2006**, *13*, 581–584. [[CrossRef](#)]
16. Park, P.; Chang, M.; Kong, N. Scheduled-stepsize NLMS algorithm. *IEEE Signal Process. Lett.* **2009**, *16*, 1055–1058. [[CrossRef](#)]
17. Huang, H.-C.; Lee, J. A new variable step-size NLMS algorithm and its performance analysis. *IEEE Trans. Signal Process.* **2012**, *60*, 2055–2060. [[CrossRef](#)]
18. Song, I.; Park, P. A normalized least-mean-square algorithm based on variable-step-size recursion with innovative input data. *IEEE Signal Process. Lett.* **2012**, *19*, 817–820. [[CrossRef](#)]
19. Isserlis, L. On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika* **1918**, *12*, 134–139. [[CrossRef](#)]
20. Iqbal, M.A.; Grant, S.L. Novel variable step size NLMS algorithms for echo cancellation. In Proceedings of the 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, Las Vegas, NV, USA, 31 March–4 April 2008; pp. 241–244.
21. Paleologu, C.; Ciochină, S.; Benesty, J. Variable step-size NLMS algorithm for under-modeling acoustic echo cancellation. *IEEE Signal Process. Lett.* **2008**, *15*, 5–8. [[CrossRef](#)]
22. Strassen, V. Gaussian Elimination is not Optimal. *Numer. Math.* **1969**, *13*, 354–356. [[CrossRef](#)]
23. Coppersmith, D.; Winograd, S. Matrix multiplication via arithmetic progressions. *J. Symb. Comput.* **1990**, *9*, 251–280. [[CrossRef](#)]
24. Ciochină, S.; Paleologu, C.; Benesty, J. An optimized NLMS algorithm for system identification. *Signal Process.* **2016**, *118*, 115–121. [[CrossRef](#)]
25. ITU. *Digital Network Echo Cancellers, ITU-T Recommendation G 168*; ITU: Geneva, Switzerland, 2002.
26. Hoyer, P.O. Non-negative matrix factorization with sparseness constraints. *J. Mach. Learn. Res.* **2001**, *49*, 1208–1215.