SEPARATE-VARIABLE ADAPTIVE COMBINATION OF LMS ADAPTIVE FILTERS FOR PLANT IDENTIFICATION

J. Arenas-García, V. Gómez-Verdejo, M. Martínez-Ramón and
A. R. Figueiras-Vidal
Department of Signal Theory and Communications,
Universidad Carlos III de Madrid, 28911 Leganés-Madrid, Spain
E-mail: jarenas@tsc.uc3m.es

Abstract. The Least Mean Square (LMS) algorithm has become a very popular algorithm for adaptive filtering due to its robustness and simplicity. An adaptive convex combination of one fast a one slow LMS filters has been previously proposed for plant identification, as a way to break the speed vs precision compromise inherent to LMS filters. In this paper, an improved version of this combination method is presented. Instead of using a global mixing parameter, the new algorithm uses a different combination parameter for each weight of the adaptive filter, what gives some advantage when identifying varying plants where some of the coefficients remain unaltered, or when the input process is colored. Some simulation examples show the validity of this approach when compared with the one-parameter combination scheme and with a different multi-step approach.

INTRODUCTION

Widrow and Hoff's Least Mean Square (LMS) [12] algorithm has become the most popular algorithm for adaptive filtering due to its robustness, good tracking capabilities, and simplicity in terms of computational load and easiness of implementation. It has therefore been applied in a wide variety of applications, including plant identification among many others [4].

One of the most important difficulties concerning LMS is the speed of convergence vs precision tradeoff that is imposed by the selection of a certain value for the adaption step. Thus, a lot of effort has been paid in the literature to the issue of how to improve this balance.

Some proposals modify the cost function to accelerate the convergence of the algorithm. This is the case of the Least Mean Fourth (LMF) algorithm [11], which offers a faster convergence than LMS for values of the absolute

error higher than one. The combination of LMS and LMF cost functions has also been explored in [2, 8, 9].

Following a different way, many researchers have proposed to manage the values of the adaption step. The authors of [7] propose a stochastic gradient adaption scheme in which the learning rate is updated after each iteration to minimize the quadratic error of the filter. Other proposals consist on increasing or decreasing the value of the adaption step according to some rules based on error signs.

The algorithm presented in [3] uses a different step for each coefficient of the adaptive filter, multiplying or dividing it by a constant factor, depending on the signs of the corresponding component of the gradient vector along some of the last iterations. Using a different step for each weight of the filter, offers advantages in any of the following situations:

- when there is a high eigenvalue spread in the autocorrelation matrix of the input process;
- in varying plants where some of the coefficients remain unaltered.

In [6] we proposed an alternative approach that has its roots in the idea of model mixtures [10, 5]. In these papers, a number of filters of different lengths are combined to make the design robust to the selection of a fixed order for the adaptive filter. Alternatively, we proposed to combine one fast (high adaption step) and one slow (low adaption step) LMS filters with the objective of getting the advantages of both of them: speed and reduced steady-state error, respectively. A more refined version of the algorithm is presented in [1], including a series of "speeding up" modifications to make the scheme fully effective. This combination of LMS filters (CLMS from now on), has the same order of complexity than LMS (roughly speaking, twice), and keeps its same basic philosophy.

In [1], CLMS was favorably compared against Variable Step LMS (VS-LMS) from [3] in plant identification tasks. The use of two independent LMS filters with different adaption steps showed to be crucial for the good performance of CLMS in comparison with schemes that are conducted by the error of just one filter. However, some advantages of VS-LMS are derived from the fact that it uses different adaption steps for every weight of the adaptive filter, while CLMS uses a global mixing coefficient.

In this paper, we propose to modify the combination scheme by using a different mixing parameter for each weight of the filter. Then, we introduce the "speeding up" modifications that should be used in this case. We compare the resulting algorithm against both basic CLMS and VS-LMS schemes.

The rest of the paper is organized as follows: in the next section we review the CLMS algorithm. Section 3 (which is the main contribution of the paper), extends CLMS ideas to the case were several mixing parameters are used, and introduces the required "speeding up" procedures to improve the performance of the new algorithm. Section 4 is devoted to some experiments that show the advantages of the new scheme. Finally, we present some conclusions and enlighten some lines for future research.

THE CLMS ALGORITHM

In [6, 1] we proposed to adaptively combine a fast LMS filter (i.e. with a high adaption step μ_1) and a slow LMS filter (low adaption step μ_2). Both filters update their weights using standard LMS adaption rule:

$$\mathbf{w}_i[k+1] = \mathbf{w}_i[k] + \mu_i e_i[k]\mathbf{x}[k], \quad i = 1, 2 \tag{1}$$

where $\mathbf{x}[k]$ is the filter input at instant k, and $e_i[k]$ is the error incurred by each filter, $e_i[k] = d[k] - \mathbf{w}_i^T[k]\mathbf{x}[k]$, $d[k]$ being the desired output.

We use a convex combination to obtain the weights of the CLMS filter

$$\mathbf{w}_{eq}[k] = \lambda[k]\mathbf{w}_1[k] + (1 - \lambda[k])\mathbf{w}_2[k] \tag{2}$$

where parameter $\lambda[k]$ is kept in the interval $[0, 1]$ by defining it as $\lambda[k] = sgm\,(a[k]) = 1/(1 + e^{-a[k]})$. The combination parameter is adapted to minimize the error of the overall adaptive filter, also using LMS adaption rule:

$$\begin{aligned} a[k+1] &= a[k] - \frac{\mu_a}{2}\frac{\partial e_{eq}^2[k]}{\partial a[k]} \\ &= a[k] + \mu_a e_{eq}[k]\left(\lambda[k]\left(1 - \lambda[k]\right)\right)\left(\mathbf{w}_1[k] - \mathbf{w}_2[k]\right)^T \mathbf{x}[k] \end{aligned} \tag{3}$$

The functioning of the combined filter is as follows: in situations where a high speed would be desirable, the fast LMS filter obtains a lower quadratic error, and application of (3) makes $\lambda[k]$ evolve towards 1, and $\mathbf{w}_{eq}[k] \approx \mathbf{w}_1[k]$. However, in stationary intervals, it is the slow filter which operates best, making $\lambda[k]$ get close to 0, and $\mathbf{w}_{eq}[k] \approx \mathbf{w}_2[k]$. So, at each time, the combined filter operates as the best filter in the mixture. Besides, factor $\lambda[k]\,(1 - \lambda[k])$ locks the mixture on one of the LMS filters when it is systematically outperforming the other, while allowing a soft switching between them.

In (3), μ_a must be fixed to a value much higher than μ_1, so the combination can progress even faster than the fastest component filter. In our implementation, we also limit $a[k]$ to lay in the interval [-4,4], to prevent stopping of the algorithm because either $\lambda[k]$ or $1 - \lambda[k]$ is too close to 0.

Using the above algorithm, the overall convergence of the CLMS filter, after an abrupt change of the plant, can never be faster than that of the slow LMS filter (in fact, it will unavoidably be slightly slower, as long as we use an stochastic gradient scheme to update the combination). We can improve the performance of the mixture by using the information provided by the fast filter to accelerate the convergence of the slow filter. To do this, we step-by-step transfer a part of weight vector $\mathbf{w}_1[k]$ to $\mathbf{w}_2[k]$:

$$\mathbf{w}_2[k+1] = \alpha\left(\mathbf{w}_2[k] + \mu_2 e_2[k]\mathbf{x}[k]\right) + (1 - \alpha)\mathbf{w}_1[k+1] \tag{4}$$

This procedure must only be applied if the fast LMS filter is performing much better than the slow one ($\lambda[k] > \beta$), and the weight vectors of the two LMS filters significantly differ ($\|\mathbf{w}_{2\perp}\|_2 / \|\mathbf{w}_{2\|}\|_2 > \gamma$, where $\mathbf{w}_{2\perp}$ and $\mathbf{w}_{2\|}$ are the components of $\mathbf{w}_2$ that are perpendicular and parallel to $\mathbf{w}_1$, respectively).

A second modification reduces the pernicious effect of factor $\mathbf{w}_1[k]-\mathbf{w}_2[k]$ in equation (3) when both weights are similar (and thus, to speed up the switching between LMS filters). To alleviate this problem we previously proposed to use a variable adaption step $\mu_a^{'}$. Here, we propose to add a momentum for the learning of $a[k]$ instead. Some simulations have demonstrated that both methods offer similar performances in practical situations.

$$a[k+1] = a[k] - \frac{\mu_a}{2}\frac{\partial e^2[k]}{\partial a[k]} + \rho(a[k]-a[k-1]) \tag{5}$$

Although CLMS algorithm requires some extra parameters, we showed in [1] that their selection is not critical, and the optimal values are not very dependent on the concrete scenario in which the filter is being applied.

USING A DIFFERENT MIXING PARAMETER FOR EACH WEIGHT: THE D-CLMS ALGORITHM

Sometimes, it would be desirable to have a high adaption step for certain weights of the adaptive filter and a low step for the rest. This is not possible with CLMS filter, as it uses a common combination parameter for all the weights. In this section, we propose an extension of CLMS that uses a different mixing parameter for each coefficient. To do this, we first group the M combination parameters (one for each coefficient in the filter) into diagonal matrix

$$\mathbf{L}[k] = \begin{pmatrix} \lambda_1[k] & 0 & \cdots & 0 \\ 0 & \lambda_2[k] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_M[k] \end{pmatrix} \tag{6}$$

$\mathbf{L}[k]$ allows us to rewrite (2), decoupling the combination of the different weights (so, we call this new filter "decoupled CLMS", D-CLMS)

$$\mathbf{w}_{eq}[k] = \mathbf{L}[k]\mathbf{w}_1[k] + (\mathbf{I}-\mathbf{L}[k])\mathbf{w}_2[k] \tag{7}$$

Again, it is convenient to use a 'sigmoidal' form for the mixing parameters: $\lambda_i[k] = sgm(a_i[k])$. Parameters $a_i[k]$ are updated using

$$\begin{aligned} a_i[k+1] &= a_i[k] - \frac{\mu_a}{2}\frac{\partial e_{eq}^2[k]}{\partial a_i[k]} \\ &= a_i[k] + \mu_a e_{eq}[k]\left(\lambda_i[k]\left(1-\lambda_i[k]\right)\right)\left(w_{1i}[k]-w_{2i}[k]\right)x_i[k] \end{aligned} \tag{8}$$

for $i = 1,\ldots,M$, where $w_{1i}[k]$, $w_{2i}[k]$ and $x_i[k]$ are the i-th components of vectors $\mathbf{w}_1[k]$, $\mathbf{w}_2[k]$ and $\mathbf{x}[k]$, respectively. The interpretation of this equation is similar to that of (3).

In this case, we typically fix μ_a to a value M times higher than that used by CLMS, to compensate for the smaller average value of the derivatives. Parameters $a_i[k]$ are also kept into [-4,4] to avoid the stopping of the algorithm.

As for the CLMS algorithm, to make D-CLMS fully effective, we should include procedures of weight transfer and a momentum for the adaption of parameters $a_i[k]$. Regarding weight transfer, the problem arises because the procedure devised for CLMS transfers the whole weight vector from the fast filter. However, to get the maximum advantage in D-CLMS, we would also like to have the possibility to transfer a limited number of weights (those whose associated $\lambda_i[k]$ is close to 1). A very simple procedure to get this is to transfer weights from the combined filter instead:

$$\mathbf{w}_2[k+1] = \alpha\left(\mathbf{w}_2[k] + \mu_2 e_2[k]\mathbf{x}[k]\right) + (1-\alpha)\mathbf{w}_{eq}[k+1] \tag{9}$$

Weight transfer must only be applied when the fast filter is getting a significantly better approximation of any coefficient (i.e., when $\lambda_i[k] > \beta$ for any $i \in \{1, \ldots, M\}$), and provided that the weight vectors of the slow and the combined filters are different enough. This second condition is satisfied if $\|\mathbf{w}_{2\perp}^{'}\|_2 / \|\mathbf{w}_{2\|}^{'}\|_2 > \gamma$ (where $\mathbf{w}_{2\perp}^{'}$ and $\mathbf{w}_{2\|}^{'}$ are the components of $\mathbf{w}_2$ that are perpendicular and parallel to $\mathbf{w}_{eq}$, respectively).

Finally, the inclusion of a momentum in (8) is straightforward, and serves to speed up adaption of mixing coefficients $a_i[k]$ when $w_{1i}[k] - w_{2i}[k] \approx 0$.

SIMULATION RESULTS

We offer simulation results for a plant identification task including slow, moderate, fast and abrupt changes, as well as stationary intervals. We will first explain the functioning of the proposed filter. Then, we will compare its performance against basic CLMS and VS-LMS filters.

The plant we use is a 3-tap transversal filter with input $x[k]$ being a white, Gaussian, zero mean, unit variance signal. The noise added at the output is the same kind of signal, but with variance 10^{-2}. The coefficients of the plant are initially selected to be

$$\mathbf{w} = [0.7252, -0.3448, 0.5093]^T$$

These coefficients are modified during the example to introduce different type of changes (see Fig. 1). The first coefficient is kept constant during all the example. The second coefficient is used to introduce moderate (it linearly increases from -0.34488 to 0.0862 during $2000 < k \leq 4500$) and slow changes (it increases from 0.0862 to 0.4310 during $10000 < k \leq 13500$). Finally, the third coefficient introduces fast and abrupt changes (it suffers an instantaneous 100% decrement at $k = 6500$, then it linearly increases to 0.7639 during $8500 < k \leq 10000$, and finally it instantaneously decreases 66% at $k = 12500$). We have also left the plant unchanged during long periods.

The performance of the algorithms will be measured as the Normalized Square Deviation (NSD) between the real and the estimated plant

$$\mathrm{NSD}[k] = \frac{\|\mathbf{w}[k] - \hat{\mathbf{w}}[k]\|_2^2}{\|\mathbf{w}[k]\|_2^2} \tag{10}$$

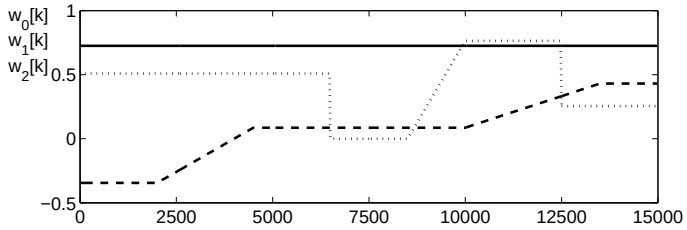


Figure 1: Evolution of the coefficients of the 3-tap plant. $w_0[k]$ (solid line) remains constant during the example, while $w_1[k]$ (dashed) and $w_2[k]$ (dotted) introduce slow and fast changes, respectively.

All the results have been averaged over 1000 runs.

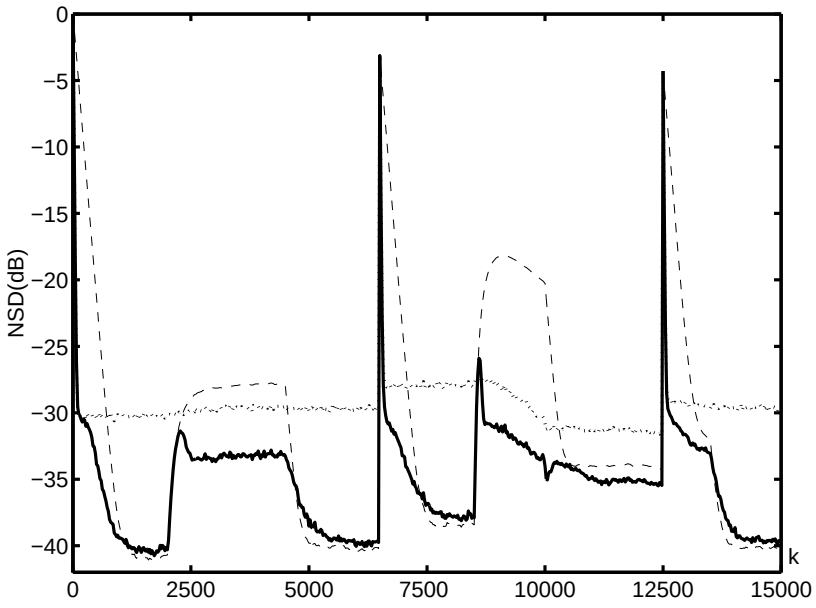
The learning rates that will be used for the D-CLMS scheme are $\mu_1 = 0.05$ for the fast LMS filter, $\mu_2 = 0.005$ for the slow LMS filter, and $\mu_a = 600$ for the combination ($\mu_a \gg \mu_1 > \mu_2$). The settings for the weight transfer procedure are $\alpha = 0.8$, $\beta = 0.95$ and $\gamma = 0.03$. These should be considered conservative values, in the sense that weight transfer is only used when the fast LMS filter is much better than the slow filter for at least one coefficient and, in this case, only a small portion of $\mathbf{w}_{eq}[k]$ is transferred to $\mathbf{w}_2[k]$. Finally, the momentum parameter has been set to $\rho = 0.5$.

We have initialized the weights of both LMS filters to the zero vector, and all the $a_i[k]$ parameters have been initially set to 4.

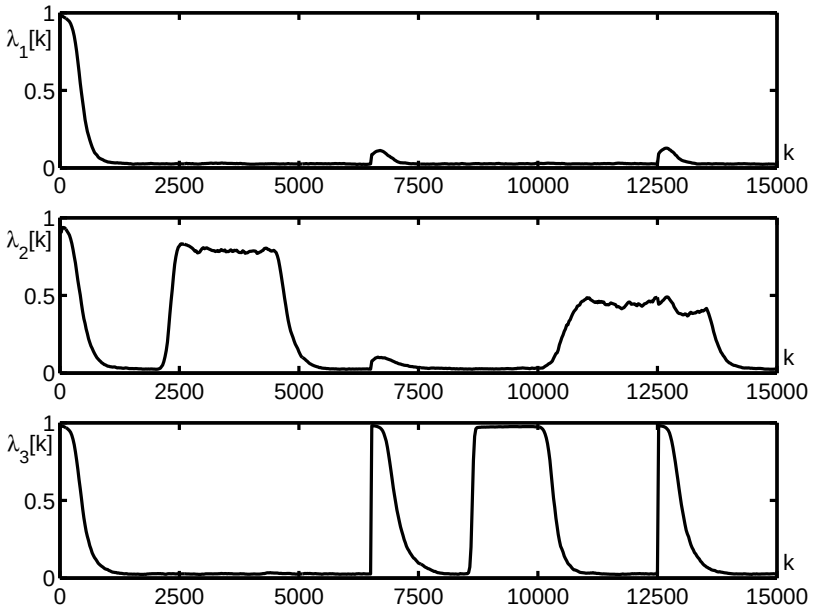
Figure 2(a) presents NSD evolution for both the LMS and D-CLMS filters. As expected, the μ_1 LMS filter has a quick convergence when the abrupt changes take place. Besides, it is able to track the plant when slow, moderate and fast changes occur. Filter $\mathbf{w}_2$ takes longer to converge after abrupt transitions, but it achieves a residual NSD of -40 dB during stationary intervals. During moderate and fast change periods ($2000 < k \leq 4500$ and $8500 < k \leq 10000$, respectively) its NSD increases above that of the fast filter. However, during slow changes, it achieves a better performance.

The D-CLMS algorithm extracts the best properties of each filter: it performs like the μ_2 filter during stationary periods, and gets the fast convergence of the fast filter after abrupt changes. In addition to this, it has a better behavior than any of the component filters for non abrupt changes. The reason is that it applies the fast filter to track the varying coefficient, and keeps the slow filter to model those which remain unaltered. Figure 2(b) shows evolution of mixing coefficients $\lambda_1[k]$, $\lambda_2[k]$ and $\lambda_3[k]$. We can check that D-CLMS correctly identifies, during all the example, which adaption step is more appropriate for each tap. As an example, during $2000 < k \leq 4500$, it is only the second coefficient which requires fast adaption capabilities. As a consequence, $\lambda_2[k]$ gets close to 1, while $\lambda_1[k]$ and $\lambda_3[k]$ remain close to 0.

Next, we compare results obtained by D-CLMS to those achieved by the basic CLMS algorithm. We have used the same settings for both filters, except that $\mu_a = 200$ is selected for CLMS to get a fair comparison. Figure 3 shows average NSD for both filters. We observe that D-CLMS clearly



(a)



(b)

Figure 2: Performance of the D-CLMS filter. (a) NSDs achieved by a fast LMS filter ($\mu_1 = 0.05$) (dotted), a slow LMS filter ($\mu_2 = 0.005$) (dashed), and by their adaptive combination using D-CLMS algorithm with $\mu_a = 600$, $\alpha = 0.8$, $\beta = 0.95$, $\gamma = 0.03$ and $\rho = 0.5$ (solid line); (b) evolution of the mixing coefficients $\lambda_1[k]$, $\lambda_2[k]$ and $\lambda_3[k]$.

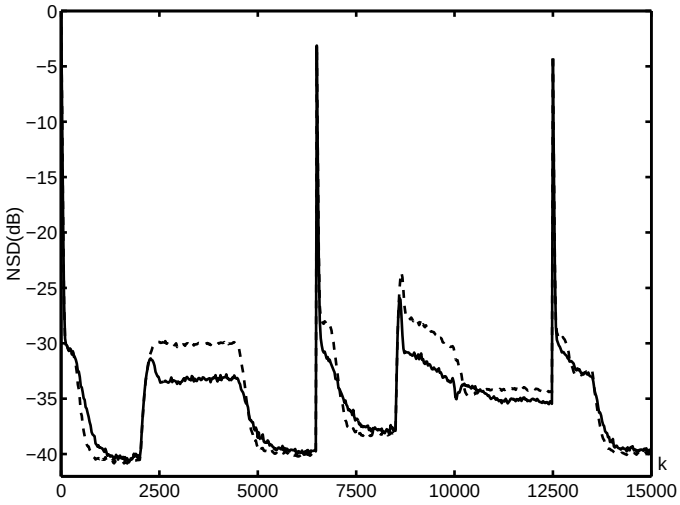


Figure 3: NSDs achieved by CLMS and D-CLMS filters (dashed and solid lines, respectively) when used with common parameters $\mu_1 = 0.05$, $\mu_2 = 0.005$, $\alpha = 0.8$, $\beta = 0.95$, $\gamma = 0.03$ and $\rho = 0.5$. $\mu_a = 200$ is selected for CLMS, while D-CLMS uses $\mu_a = 600$.

outperforms CLMS during periods with slow, moderate and fast changes. The reason for this is that, as CLMS has just one combination parameter, its behavior is limited to that of the best LMS filter. However, as we have already explained, D-CLMS is able to construct a more elaborated combination of the LMS filters, and gets a lower NSD in these situations. Steady-state misadjustments of CLMS and D-CLMS are similar, but CLMS offers a slightly faster convergence. Surely, the reason for this is that learning of CLMS parameter $a[k]$ is easier than learning of the 3 mixing parameters of D-CLMS, because it somehow averages information from all the weights of the filter.

Finally, we offer a comparison with a different multi-step scheme: Variable Step LMS (VS-LMS) [3]. VS-LMS uses a different adaption step, $\mu_i[k]$, for each coefficient. Adaption of $\mu_i[k]$ depends on the signs of the i-th coordinate of the error gradient ($sign\,(e[k]x_i[k])$): if m_0 consecutive alternate signs occur, it is divided by a factor c. $\mu_i[k]$ is multiplied by c after m_1 identical signs. In addition to this, $\mu_i[k]$ is kept into a in interval $[\mu_{min}, \mu_{max}]$.

In our experiments we have fixed $\mu_{max} = 2/9$, $\mu_{min} = 0.005$, $c = 1.1$ and $\mu_i[0] = \mu_{max}$, for $i = 1, \ldots, M$. μ_{max} has been fixed to a value such that stability is preserved. Note that this value is higher than our μ_1 what could give VS-LMS some advantage under fast changes. We have explored VS-LMS behavior for different values of m_0 and m_1.

In Figure 4(a), we have depicted average NSD evolution for $m_0 = m_1 = 1$ (these settings are recommended by the authors of [3] for non-stationary situations). In this case, VS-LMS seems to perform similarly to a fast LMS filter, and D-CLMS clearly outperforms it during all the example.

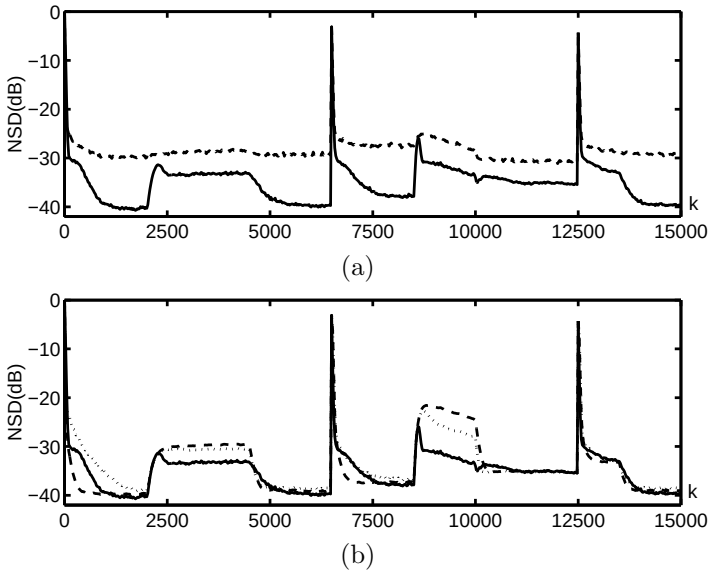


Figure 4: NSDs achieved by D-CLMS and VS-LMS. In all figures D-CLMS (with the usual settings) is depicted using a solid line, while the rest stand for VS-LMS with $\mu_{max} = 2/9$, $\mu_{min} = 0.005$, $c = 1.1$ and variable m_0 and m_1: (a) $m_0 = m_1 = 1$ (dashed); (b) $m_1 = m_0 + 1$, for $m_0 = 1$ (dashed) and for $m_0 = 4$ (dotted).

Selection of $m_1 > m_0$ strongly favors the decrease of the adaption steps towards μ_{min}, what makes VS-LMS more competitive during stationary and slow changes intervals, as can be seen in Fig. 4(b) for VS-LMS with $m_1 = m_0 + 1$ for $m_0 = 1$ and for $m_0 = 4$. However, the performance in situations requiring fast adaption is now clearly worst than that of D-CLMS (see interval $8500 < k \leq 10000$). Analyzing the responses of VS-LMS for the considered settings, we can conclude that a higher ratio m_1/m_0 serves to improve its behavior in situations where low adaption steps are appropriate, at the cost of a higher NSD in the opposite case. We have shown that D-CLMS filter does not suffer from this performance compromise.

CONCLUSIONS

The use of a fast convex combination of one fast and one slow LMS filters is a powerful approach to improve the speed vs precision tradeoff inherent to the LMS algorithm in plant identification tasks. In this paper, we have modified a previous combination method (CLMS), adding a different combination parameter for each step of the filter (D-CLMS), what gives some advantage when identifying time varying plants, or when the input process is colored. "Speeding up" procedures to further improve the performance of the method have also been introduced. Simulation examples show a clear advantage of the proposed scheme.

Among the wide variety of signal processing applications that can benefit from the new combination method, D-CLMS could be of particular interest for those that use colored processes (for example, AR prediction). Besides, the proposed scheme can be further refined by using the mixing parameters to manage the values of the adaption steps of the LMS filters.

Acknowledgments

This work has been partly supported by CICYT grant TIC2002-03713.

REFERENCES

[1] J. Arenas-García, M. Martínez-Ramón, A. Navia-Vázquez and A. R. Figueiras-Vidal, "Adaptive Combination of Transversal LMS Filters for Plant Identification," submitted to **IEEE Trans. on Signal Processing**, 2002.

[2] J. Chambers, O. Tanrikulu and A. Constantinides, "Least Mean Mixed-Norm Adaptive Filter," **Electronic Letters**, vol. 30, no. 19, pp. 1574–1575, 1994.

[3] R. W. Harris, D. M. Chabries and F. A. Bishop, "Variable Step (VS) Adaptive Filter Algorithm," **IEEE Trans. Acoustics, Speech and Signal Processing**, vol. ASSP-34, pp. 305–316, 1986.

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

[5] S. S. Kozat and A. C. Singer, "Multi-stage Adaptive Signal Processing Algorithms," in **Proc. of the 2000 IEEE Sensor Array and Multichannel Signal Processing Workshop**, Cambridge, MA, March 2000, pp. 380–384.

[6] M. Martínez-Ramón, J. Arenas-García, A. Navia-Vázquez and A. R. Figueiras-Vidal, "An Adaptive Combination of Adaptive Filters for Plant Identification," in **Proc. of the 14th Intl. Conf. on Digital Signal Proc.**, Santorini, Greece, July 2002, pp. 1195–1198.

[7] V. J. Mathews and Z. Xie, "A Stochastic Gradient Adaptive Filter with Gradient Adaptive Step Size," **IEEE Trans. on Signal Processing**, vol. 41, no. 6, pp. 2075–2087, 1993.

[8] D. I. Pazaitis and A. G. Constantinides, "LMS - F Algorithm," **Electronic Letters**, vol. 31, no. 17, pp. 1423–1424, 1995.

[9] C. Rusu, M. Helsingius and P. Kuosmanen, "Joined LMS and LMF Threshold Technique for Data Echo Cancellation," in J. Tasic (ed.), **Proc. COST#254 Intl. Workshop on Intelligent Comms. and Multimedia Terminals**, Ljubliana, Slovenia, Nov. 1998, pp. 127–130.

[10] A. C. Singer and M. Feder, "Universal Linear Prediction by Model Order Weighting," **IEEE Trans. on Signal Proc.**, vol. 47, no. 10, pp. 2685–2700, 1999.

[11] E. Walach and B. Widrow, "The Least Mean Fourth (LMF) Algorithm and Its Family," **IEEE Trans. Information Theory**, vol. IT-30, no. 2, pp. 275–283, 1984.

[12] B. Widrow and M. E. Hoff, "Adaptive Swithching Circuits," **Wescon Conv. Record**, pp. 96–140, 1960.