Plant Identification via Adaptive Combination of Transversal Filters ⋆

Jerónimo Arenas-García *, Manel Martínez-Ramón,

Ángel Navia-Vázquez, Aníbal R. Figueiras-Vidal

Department of Signal Theory and Communications,

Universidad Carlos III de Madrid,

28911 Leganés-Madrid, Spain

Abstract

For least mean-square (LMS) algorithm applications, it is important to improve the speed of convergence vs the residual error trade-off imposed by the selection of a certain value for the step size. In this paper, we propose to use a mixture approach, adaptively combining two independent LMS filters with large and small step sizes to obtain fast convergence with low misadjustment during stationary periods. Some plant identification simulation examples show the effectiveness of our method when compared to previous variable step size approaches. This combination approach can be straightforwardly extended to other kinds of filters, as it is illustrated with a convex combination of recursive least-squares (RLS) filters.

Key words: Least Mean Square (LMS), Adaptive Algorithms, Convex Combination, Plant Identification

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

* Corresponding author.
Email address: jarenas@tsc.uc3m.es (Jerónimo Arenas-García).

Plant Identification via Adaptive Combination of LMS Filters

Jerónimo Arenas-García, Manel Martínez-Ramón,

Ángel Navia-Vázquez, Aníbal R. Figueiras-Vidal

Number of Pages: 30

Number of Figures: 5

Number of Tables: 1

Key words: Least Mean Square (LMS), Adaptive Algorithms, Convex Combination, Plant Identification

1 Introduction

Widrow and Hoff's least mean-square (LMS) algorithm [1] was first presented in 1960. Ever since, it has been used in a variety of applications due to its robustness, good tracking capabilities, and simplicity [2]. An important limitation of LMS is the rate of convergence vs misadjustment trade-off imposed by the selection of a certain value for the step size. Consequently, much effort has been spent to improve this balance. Some modifications of the LMS algorithm apply non-quadratic error functions, while other authors have proposed to use an adaptive learning rate that takes a high value when a high speed of convergence is required, and a low value to decrease the final misadjustment in stationary situations.

Among those that modify the cost function, the least mean-fourth (LMF) algorithm [3], which consists of minimizing the fourth power of the error, has played a very important role, since, for values of the absolute error higher than 1, the convergence of the LMF algorithm is faster than that of LMS. However, stability of the LMF filter imposes a more restrictive upper bound on the adaption step size. The idea of combining LMS and LMF cost functions was first proposed in [4], with the objective of keeping the speed of convergence of LMF and the stability of LMS. In [5] the authors use the same basic idea, but let the mixing parameter vary according to the instantaneous value of the error. The idea of switching between both cost functions has been explored in [6].

Following a different direction, many researchers have proposed schemes to manage the step size. Some of such algorithms can be found in [7]. Several

designs in this group have the disadvantage of being batch algorithms, which are not appropriate when on-line learning is required. There are also several schemes that manage the step size in a stochastic manner (see, for instance, [8–13]). All these procedures obtain good results and are computationally efficient, but they add some hyperparameters which must be fixed to *a priori* values. This selection itself implies some kind of compromise between convergence capabilities and precision, and optimal values are highly dependent on the characteristics of the particular application environment.

Following this second way above, we propose a new approach that is an analogy of a well-known neurological fact: in human brains, the thalamus sends the signals that it receives from the ears or the eyes to the amygdala via a fast connection, and to the neocortex following a slower way. After these first transmissions, the neocortex and the amygdala interact. This allows us to combine a fast, coarse reaction against abrupt and potentially dangerous environmental changes with a finer, more elaborated, but necessarily not-so-fast conscious response. The analogous scheme that we propose consists of an adaptive convex combination of one fast and one slow LMS filter, which keeps the advantages of both speed of convergence and reduced steady-state error. In tracking situations, our combined filter behaves as the best LMS filter in the mixture. Thus, instead of using just one LMS filter and adaptively selecting an appropriate value for its step size, our combined filter just has to decide how to optimally combine both filters at each time. It will be shown that this nature-principled approach offers very satisfactory results.

Our combination scheme is similar to that in [14,15], which uses the idea of mixture of different models as Jacobs et al. [16] did for neural networks to increase representation capability. However, the filters in our scheme just

differ in their step sizes, and they are combined with quite a different aim: to simultaneously exploit the good convergence and precision capabilities of the fast and slow LMS filters, respectively.

The same basic ideas of combining adaptive filters in an adaptive manner can also be applied to other adaptive schemes without major modifications, as well as to situations different from plant identification problems, such as adaptive equalization, arrays, blind source separation, etc.

In the next section we describe the basic algorithm to combine LMS filters in an adaptive manner (CLMS algorithm), in the context of plant identification, as well as some modifications to improve its practical performance. Then, we will present a more sophisticated algorithm (CLMS2) that exploits the convex combination ideas in a three-filter configuration. Section 4 analyzes the computational complexity of the combination methods. In Section 5 we show the advantages of our proposal through some simulation results. Finally, we present our conclusions and suggest some lines of future research.

2 Adaptive Combination of LMS Filters: the CLMS Algorithm

We propose to use, for plant identification, an adaptive convex combination of two LMS adaptive filters. The first is a fast filter (i.e., with a large step size μ_1), which achieves fast convergence and good tracking properties in situations where rapid changes take place. The second is a slow filter (small μ_2) that will provide a very good approximation of the plant in stationary (or slowly changing) situations. This idea was presented as a preliminary version in [17]. Here, we add several algorithmic changes to circumvent the serious limitations

of that version.

The two component filters operate, in principle, completely decoupled, and their coefficients are adapted to minimize their own quadratic errors using the standard LMS rule

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) + \mu_i e_i(n)\mathbf{u}(n), \quad i = 1, 2 \tag{1}$$

where $\mathbf{w}_i(n)$ are the filter weight vectors at time n,

$$e_i(n) = d(n) - \mathbf{w}_i^T(n)\mathbf{u}(n), \tag{2}$$

$d(n)$ is the desired output, and $\mathbf{u}(n) = [u(n), u(n-1), \ldots, u(n-M+1)]^T$ are the inputs to the filter, M being the length of the adaptive filter.

The weights of the combined filter are obtained using the following convex combination scheme

$$\mathbf{w}(n) = \eta(n)\mathbf{w}_1(n) + [1 - \eta(n)]\mathbf{w}_2(n) \tag{3}$$

where $\eta(n)$ is a mixing parameter that lies between 0 and 1. One can immediately express the output and the error of the CLMS filter [$y(n)$ and $e(n)$, respectively] as convex combinations of the two component filters:

$$\begin{aligned} y(n) &= \mathbf{w}^T(n)\mathbf{u}(n) = \eta(n)y_1(n) + [1 - \eta(n)]\, y_2(n) \\ e(n) &= d(n) - y(n) = \eta(n)e_1(n) + [1 - \eta(n)]\, e_2(n). \end{aligned} \tag{4}$$

where

$$y_i(n) = \mathbf{w}_i^T(n)\mathbf{u}(n) \tag{5}$$

is the output of the ith adaptive filter.

Regarding parameter $\eta(n)$, we will define it via a sigmoid activation function

[18]

$$\eta(n) = \mathrm{sgm}\left[a(n)\right] = \frac{1}{1 + e^{-a(n)}} \tag{6}$$

where we update $a(n)$ at each iteration to minimize the quadratic error of the combined filter, also by means of a minimum square error stochastic gradient algorithm [2] with step size μ_a:

$$\begin{aligned} a(n+1) &= a(n) - \frac{\mu_a}{2}\frac{\partial e^2(n)}{\partial a(n)} \\ &= a(n) - \mu_a e(n)\left[e_1(n) - e_2(n)\right]\eta(n)\left[1 - \eta(n)\right]. \end{aligned} \tag{7}$$

The benefits of using a sigmoid activation are twofold: first, it serves to constrain $\eta(n)$ values between 0 and 1; second, its derivative, i.e., factor $\eta(n)\left[1 - \eta(n)\right]$ in (7), reduces the adaptation speed and the gradient noise near the endpoints 0 and 1, when the combined scheme is expected to perform, respectively, like the fast and slow filters without degradation.

In our implementation, we have limited $a(n)$ values to the interval $[-4, 4]$, so the algorithm never stops because either $\eta(n)$ or $1 - \eta(n)$ are then never too close to 0. Furthermore, the parameter μ_a must be fixed to a very high value so that the combination is adapted even faster than the fastest LMS filter. Note that, since $\eta(n) \in (0, 1)$, the stability of the combined filter is guaranteed as long as the individual stability conditions of both LMS filters are satisfied.

This scheme has a very intuitive interpretation. When abrupt or fast changes appear, the fast LMS filter achieves a quadratic error lower than that of the slow filter, application of the learning rule (7) will increase $a(n)$, and $\eta(n)$ will approximate 1. So, in this situation, the CLMS algorithm acts as a μ_1 LMS filter. However, in stationary (or nearly-stationary) intervals, it is the slow LMS filter that will perform best, thus decreasing $a(n)$ so that $\eta(n)$ will approach 0, and the combined filter will behave as the slow, more precise LMS

filter [19,20].

It should be noted that (7) only uses the errors of the component filters. So, this convex combination scheme could be used, with no modifications, to combine any other two adaptive filters, as we show in Section 5 with a combination of two recursive least-squares (RLS) filters.

Figure 1 summarizes the proposed scheme up to this point, indicating which error is used to control each LMS filter and their combination.

2.1 Speeding up the convergence of the slow filter

The performance of the basic CLMS algorithm can be improved if we allow interaction between the component filters in certain situations. To see this, note that, if both filters are completely decoupled, abrupt changes will induce independent convergences, their combination being equivalent to the fast filter until the slow filter presents a smaller quadratic error. At that moment, the CLMS filter will switch to the second filter. So, the final overall convergence will be as slow as that of the μ_2 LMS filter.

We can accelerate the convergence of the slow filter when an abrupt change appears by step-by-step transferring a part of weight vector $\mathbf{w}_1$ to $\mathbf{w}_2$. Thus, the (modified) adaption rule for $\mathbf{w}_2$ becomes

$$\mathbf{w}_2(n+1) = \alpha\left[\mathbf{w}_2(n) + \mu_2 e_2(n)\mathbf{u}(n)\right] + (1-\alpha)\mathbf{w}_1(n+1) \tag{8}$$

where α is a parameter close to 1. This way, at each step, the adaption of $\mathbf{w}_2$ is similar to that that of a standard LMS filter, but the application of (8) over many consecutive iterations will speed up the convergence of the μ_2 LMS

filter and, as a consequence, the convergence of the CLMS filter.

An inconvenience of the new learning rule is that it increases the final misadjustment of the slow filter. To avoid this, the weight transfer must only be applied when the fast filter is much better than the slow one in tracking the plant changes. Our combination method provides us with a straightforward way to verify this condition: $\eta(n)$ must be near 1. So, we will only use this speeding up procedure provided that $\eta(n) > \beta$, β being a threshold close to the maximum value that can be reached by $\eta(n)$. It would be possible to use more robust criteria to activate/deactivate this weight transfer procedure. However, we prefer the above mode to keep the simplicity inherent to LMS-type algorithms.

Although this "speeding up" mechanism requires two extra parameters, we have checked that the CLMS algorithm is not very sensitive to the selection of α and β. In any case, this mechanism should be seen as an optional procedure that can be used to improve the performance of the basic combination algorithm in very particular situations.

3 Iterating the Adaptive Combination Scheme: the CLMS2 Algorithm

Obviously, the performance of the CLMS scheme could be further improved by managing μ_1 and μ_2. Parameter $\eta(n)$ can be used as a criterion to manage these step sizes. We will only manage μ_2, since the fast filter is used just as a fast tracker for rapid changes and, consequently, managing μ_1 would not give much advantage.

Instead of directly changing μ_2, we propose to iterate the convex combination scheme and replace filter $\mathbf{w}_2$ with a CLMS filter with step sizes μ_{21} and μ_{22} ($\mu_1 > \mu_{21} > \mu_{22}$)

$$\mathbf{w}_2(n) = \eta_2(n)\mathbf{w}_{21}(n) + [1 - \eta_2(n)]\mathbf{w}_{22}(n) \tag{9}$$

with $\eta_2(n) = \mathrm{sgm}[a_2(n)]$. Since this new configuration consists of a double nesting of CLMS, we call the resulting scheme CLMS2.

The mixing parameter of the inner CLMS filter is adapted like $a(n)$, but to minimize $e_2^2(n) = [d(n) - \mathbf{w}_2^T(n)\mathbf{u}(n)]^2$:

$$a_2(n+1) = a_2(n) - \mu_{a2} e_2(n) \left[e_{21}(n) - e_{22}(n)\right] \eta_2(n) \left[1 - \eta_2(n)\right]. \tag{10}$$

Since we usually have that $|e_{21}(n) - e_{22}(n)| < |e_1(n) - e_2(n)|$, μ_{a2} must be higher than μ_a (we have typically used $\mu_{a2} = 10\mu_a$), and applying weight transfer from $\mathbf{w}_{21}$ to $\mathbf{w}_{22}$ makes little sense.

We will only use this three-filter scheme provided that $\eta(n)$ is below some threshold that, in our implementation, has been set to the same β used as a threshold for weight transfer [1]. Switching between two- and three-filter configurations is made in the following manner:

- We initially use a two-filter scheme with parameters μ_1 and μ_2, and $a(n)$ is set to its highest value [$a(n) = 4$].
- If $\eta(n)$ falls below β, we switch to a three-filter scheme: $\mathbf{w}_{21}(n)$ and $\mathbf{w}_{22}(n)$ are set to $\mathbf{w}_2(n)$, $a_2(n)$ is set to 0, and we choose new learning rates $\mu_{21} = \mu_2$ and $\mu_{22} = \mu_2/r$ ($r > 1$).

[1] It would be possible to use a different threshold. However, using β gives good performance while saving one parameter.

- Finally, if $\eta(n)$ exceeds β, we return to the two-filter configuration according to the present value of $\mathbf{w}_2(n)$.

When using a three-filter configuration, $\eta_2(n)$ gives information on the goodness of filters $\mathbf{w}_{21}$ and $\mathbf{w}_{22}$. This information could be used to iterate the process and replace either $\mathbf{w}_{21}$ or $\mathbf{w}_{22}$ with a convex combination of two filters (i.e., we could use a further nesting of CLMS filters). Alternatively, we propose to manage μ_{21} and μ_{22} following these rules:

- If $\eta_2(n)$ is near 1 [$\eta_2(n) > \beta$, for example], we increase the learning rates to $\mu_{21}^{'} = r\mu_{21}$ and $\mu_{22}^{'} = \mu_{21}$, and set $\mathbf{w}_{22}(n) = \mathbf{w}_{21}(n)$ and $a_2(n) = 0$. Note that we keep a filter with step size μ_{21} in order to guarantee that the performance of filter $\mathbf{w}_2$ does not worsen.
- The opposite is done when $\eta_2(n)$ is close to 0 [$\eta_2(n) < 1 - \beta$].

4 Computational Complexity of the Proposed Algorithms

In this section, we study the computational complexity of the proposed combination methods. Since the number of additions is comparable to the number of multiplications for the proposed combinations, we will consider just the number of real multiplications.

It is a well-known fact that LMS operation requires $2M+1$ multiplications, M being the length of the filter. Since CLMS combines two LMS filters, it needs $4M+2$ multiplications for the adaptation of the component filters, and 6 more products are needed to compute the output of the filter and to update $a(n)$ [see (4) and (7), respectively]. Evaluation of the sigmoid function is not well suited for real time operation, but it could be efficiently implemented using a

look-up table.

Depending on the application, it may be necessary to explicitly compute the CLMS weight vector, $\mathbf{w}(n)$. According to (3), explicit weight calculation requires $2M$ additional multiplications. Finally, if the weight transfer procedure is being applied, it will become necessary to compute $2M$ extra products at some iterations.

Regarding CLMS2, when the two-filter operation is used $[\eta(n) > \beta]$, its computational complexity is exactly the same as for CLMS. However, if the three-filter configuration is active, it requires $6M + 3$ multiplications for the component filter adaptations, and 12 more products to compute the output of the scheme and to adapt $a(n)$ and $a_2(n)$. Further, $3M + 2$ multiplications are necessary in this case for the explicit calculation of the weights of the overall filter. Note that weight transfer is not applied when using the three-filter configuration since $\eta(n) < \beta$.

Table 1 summarizes the computational complexity of CLMS and CLMS2. As we can see, their complexity grows linearly with the number of taps $[O(M)]$, and it is roughly twice and three times higher than that of LMS for the basic combination schemes. The application of the weight transfer procedure increases the computational burden of the filters. Note, however, that this method has been designed to accelerate the convergence of the slow LMS filter in a very particular situation, and it is not necessary to get the main goal of the combination schemes: to put together the best properties of each component filter.

The above analysis and conclusions would still be valid if combining other kinds of adaptive filters different from LMS, modifying accordingly the com-

putational complexity for the adaptation of the component filters.

5 Experimental Results

5.1 Discussion of CLMS Performance

In this section, we use the CLMS filter to model a 24-tap transversal varying plant (see Fig. 1) with input $u(n)$ being a white, Gaussian, zero-mean signal. The variance of $u(n)$ is selected to set $E\{\|\mathbf{u}(n)\|_2^2\} = 1$. The output additive noise, $e_0(n)$, is the same kind of signal, but has variance 10^{-2}.

The weights of the plant are initially set to random values between -1 and 1, and are modified during some intervals according to the random-walk model:

$$\mathbf{w}_0(n+1) = \mathbf{w}_0(n) + \mathbf{q}(n) \tag{11}$$

where $\mathbf{q}(n)$ is an i.i.d. random, zero-mean vector, with diagonal covariance matrix $E\{\mathbf{q}(n)\mathbf{q}^T(n)\} = \sigma_q^2\mathbf{I}$. The variance σ_q^2 is related to the speed of changes in the plant. We will consider two different speeds, $M\sigma_q^2 = 5\cdot 10^{-7}$ and $M\sigma_q^2 = 10^{-4}$, for intervals $50000 < n \leq 75000$ and $100000 < n \leq 125000$, respectively. At $n = 75000$, the plant changes abruptly and new random values are selected within $[-1, 1]$.

To model the plant, we have used a CLMS filter with $M = 24$ and step sizes $\mu_1 = 0.05$ (which satisfies the stability condition), $\mu_2 = 0.005$, and $\mu_a = 400$ for the mixing parameter. For the "speeding up" parameters we have used $\alpha = 0.9$ and $\beta = 0.98$. These are reasonable values for a wide range of practical situations, as we will check in this example. Finally, the mixing parameter $a(n)$

is initially set to its intermediate value $[a(0) = 0]$, and the weights of both LMS filters are initialized with zeros.

Filter performance will be measured with an estimate of the mean square deviation (MSD) between the real and the estimated plant, calculated as the average of the plant identification error

$$\mathrm{MSD}(n) = \|\mathbf{w}_0(n) - \mathbf{w}(n)\|_2^2$$

over 1000 independent runs.

Figure 2(a) presents the performance of the LMS and CLMS filters, and Fig. 2(b) depicts the evolution of the mixing parameter η. The fast LMS filter has a quick convergence after the abrupt changes at $n = 0$ and $n = 75000$, achieving a residual error of approximately -22 dB over the entire interval, with a small increase during the rapid changes at $100000 < n \leq 125000$. On the contrary, the slow filter obtains a lower MSD not only at the stationary intervals (MSD ≈ -32 dB), but also when the plant changes slowly ($50000 < n \leq 75000$). During the fast change period, however, the slow filter is unable to track the plant, and its MSD increases towards -5 dB.

The CLMS algorithm performs like the slow LMS filter during the stationary and slow change intervals [with $\eta(n)$ near 0], and it takes the (lower) MSD of the μ_1 filter when fast and abrupt changes occur [with $\eta(n)$ close to 1]. Note that, after the transitions, the use of the "speeding up" procedures allows the CLMS filter to reach the final misadjustment of a μ_2 filter very soon in comparison to the performance of this LMS filter, thus accelerating the convergence of the combined filter.

5.2 Comparison to an LMS Filter with Variable Step Size

We have carried out a large number of experiments comparing CLMS with other previous variable step size schemes. We will consider here the LMS Algorithm with adaptive gain (AG-LMS) [11,21], which is a good representative of these versions.

When using an adaptive step size, the LMS adaption rule becomes

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n)e(n)\mathbf{u}(n), \tag{12}$$

The AG-LMS algorithm proposes to use a gradient algorithm to adapt the step size

$$\begin{aligned} \mu(n+1) &= \mu(n) - \frac{\epsilon}{2}\frac{\partial e^2(n)}{\partial \mu} \\ &= \mu(n) + \epsilon e(n)\mathbf{\Psi}^T(n)\mathbf{u}(n) \end{aligned} \tag{13}$$

where

$$\mathbf{\Psi}(n) = \frac{\partial \mathbf{w}(n)}{\partial \mu(n)}. \tag{14}$$

The recursion for adapting $\mathbf{\Psi}(n)$ is obtained by differentiating both sides of (12) with respect to μ:

$$\mathbf{\Psi}(n+1) = [\mathbf{I} - \mu(n)\mathbf{u}(n)\mathbf{u}^T(n)]\mathbf{\Psi}(n) + e(n)\mathbf{u}(n). \tag{15}$$

The AG-LMS algorithm iteratively applies (12), (13), and (15) to update $\mathbf{w}(n)$ and $\mu(n)$. In addition, $\mu(n)$ must be truncated to remain within the interval $[\mu_{\min}, \mu_{\max}]$. According to [11], $\mu_{\min}$ is nearly irrelevant to get good behavior, and it can be fixed either to 0 or to a very small constant. On the contrary, $\mu_{\max}$ has a large influence in the performance of the algorithm.

We have carried out experiments with the same plant used in the previous subsection. For these experiments we have fixed $\mu_{\min} = 0$, and have explored different combinations of $\mu_{\max}$ and ϵ. Next, we will describe the characteristics of the AG-LMS performance, and then we will compare it to our CLMS algorithm.

- For fixed ϵ and decreasing $\mu_{\max}$, AG-LMS obtains a decreasing residual misadjustment, but also shows slower convergence. In Fig. 3(a), we show the MSD for AG-LMS with $\epsilon = 0.01$ and two different values of $\mu_{\max}$. As stated before, using a low value $\mu_{\max} = 0.01$ favors the good behavior of AG-LMS during stationary intervals. However, for this value of $\mu_{\max}$, the performance of AG-LMS is very poor during interval $100000 < n \leq 125000$, when fast changes appear in the plant, and following abrupt changes (see convergence after $n = 75000$). It is possible to improve the convergence and tracking capabilities of AG-LMS by using a higher $\mu_{\max}$. This indeed occurs for $\mu_{\max} = 0.05$, but it can be seen that, in this case, the residual MSD in stationary situations is higher.

  We have checked that this performance compromise between slow and fast change situations also appears for other values of ϵ.
- A similar trade-off occurs when $\mu_{\max}$ is kept constant and the value of ϵ is modified. This situation is shown in Fig. 3(b) for $\mu_{\max} = 0.05$ and two values of ϵ. We see that using a very small ϵ reduces the MSD achieved by AG-LMS in very long stationary intervals at the cost of a degraded convergence.

When comparing CLMS to AG-LMS with $\epsilon = 0.01$ [Fig. 3(a)], we see that CLMS obtains a slightly lower misadjustment during stationary and slowly changing intervals. Moreover, CLMS is as good as AG-LMS with $\mu_{\max} = 0.05$ in tracking the fast changes during $100000 < n \leq 125000$ and after the abrupt

change in the plant at $n = 75000$.

The performance of AG-LMS with $\mu_{\max} = 0.05$ and $\epsilon = 0.0003$ [Fig. 3(b)], is generally worse than that of CLMS, although it is able to get a smaller residual MSD in very long stationary intervals, since CLMS can not achieve a lower MSD than that of a μ_2 LMS filter.

From the above, we conclude that AG-LMS performance is very sensitive to the selection of parameters $\mu_{\max}$ and ϵ, which is a difficult task. In addition to this, these parameters introduce a performance compromise for stationary, slow, and fast change situations. We have checked that this trade-off is present in most of the variable step size algorithms. On the contrary, CLMS is very effective at exploiting the best properties of its two component filters in each situation and, although it also introduces a number of parameters, these do not suffer the same kind of compromise.

In Section 3 we introduced the CLMS2 algorithm as a way to effectively manage the learning rate μ_2. Now, we will show how CLMS2 can be used to further improve the performance of the CLMS filter. In Figure 4, we have depicted the MSD achieved by the CLMS2 filter with the same settings of CLMS and additional parameters $\mu_{a2} = 10\mu_a$ and $r = 2$. We see that CLMS2 keeps the good performance of CLMS during the whole example, improving its performance during stationary situations, where it obtains a similar or even lower MSD than AG-LMS (see MSDs of both algorithms after $n = 150000$). Again, note that CLMS2 does not suffer any performance compromise and, in non-stationary situations, its MSD is the same of the CLMS filter.

Obviously, our scheme for combining adaptive filters can be used with no modifications to combine filters different from LMS. In this subsection, we will illustrate this fact with an adaptive combination of recursive least-squares (RLS) filters [2] with different forgetting factors.

The example involves the identification of an 8-tap plant whose weights are selected within $[-1, 1]$, and are modified during $15000 < n \leq 35000$ and $60000 < n \leq 75000$ according to the random-walk model (11) with $M\sigma_q^2 = 10^{-6}$ and $M\sigma_q^2 = 10^{-3}$, respectively. An abrupt change is induced in the plant at $n = 35000$, assigning new random values to all the weights. In this case, the input signal, $u(n)$, is obtained from a first-order autoregressive model with transfer function $\sqrt{1-a^2}/(1-az^{-1})$, $a = 0.7$, fed with i.i.d. Gaussian noise, whose variance was tuned to set $E\{\|\mathbf{u}(n)\|_2^2\} = 1$.

To model the plant we combine two RLS filters minimizing the exponentially weighted costs

$$C_i(n) = \sum_{m=1}^{n} \lambda_i^{n-m} e^2(m), \quad i = 1, 2 \tag{16}$$

with $\lambda_1 = 0.995$ and $\lambda_2 = 0.9995$, so that the first filter has a shorter memory and, consequently, adapts faster to fast or abrupt changes in the plant. For the combination of both filters (CRLS scheme) we have used the same settings considered in the CLMS case, i.e., $\mu_a = 400$, $\alpha = 0.9$, $\beta = 0.98$, and $a(0) = 0$.

In Figure 5 we present the performance of the RLS and CRLS filters. Again, the combined scheme extracts the best properties of each component at each time, performing like the λ_1 RLS filter during $60000 < n \leq 75000$ and after $n = 35000$, and achieving the lower MSD of the second filter in stationary and

nearly-stationary intervals.

Finally, it is worth mentioning that colored inputs do not affect the performance of combined schemes, as long as the component filters are robust to this situation, which is a well-known property of RLS filters.

6 Conclusions

An adaptive convex combination of one fast and one slow LMS filters constitutes a reasonable approach to get both speed and precision in plant identification, as intuitively expected. Practical implementations must include a simple weight transfer mechanism to be fully effective. Additionally, further splitting of the slow filter provides even better capabilities. In this case, the resulting procedure is equivalent to step size management, without adopting any particular algorithm for it, but searching the optimal value at each iteration. All the parameters in the combined schemes have easy design rules, and the whole algorithm is robust enough with respect to their selection. Simulation examples show a clear advantage of the proposed scheme with respect to previous variable step size proposals.

Obviously, the proposed design can be further refined: for example, using different step sizes for each weight will be effective in cancellation applications. Finally, it is also evident that the convex combination scheme could also be used to combine other adaptive filters, as we have illustrated with a combination of RLS filters, and other adaptation rules can be considered for the mixing parameter. This opens the way to exploit the basic ideas of our approach in many other applications of adaptive systems.

References

[1] B. Widrow, M. E. Hoff, Adaptive switching circuits, in: Wescon Conv. Record, (1960) pp. 96–104.

[2] S. Haykin, Adaptive Filter Theory, Prentice-Hall, Upper Saddle River, NJ, 2002.

[3] E. Walach, B. Widrow, The least mean fourth (LMF) algorithm and its family, IEEE Trans. Information Theory 30 (Mar. 1984) 275–283.

[4] J. Chambers, O. Tanrikulu, A. Constantinides, Least mean mixed-norm adaptive filtering, Electronics Letters 30 (Sep. 1994) 1574–1575.

[5] D. I. Pazaitis, A. G. Constantinides, LMS+F algorithm, Electronics Letters 31 (Aug. 1995) 1423–1424.

[6] C. Rusu, M. Helsingius, 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, 1998, pp. 127–130.

[7] A. Cichoki, R. Unbehauen, Neural Networks for Optimization and Signal Processing, Wiley, West Sussex, UK, 1993.

[8] R. W. Harris, D. M. Chabries, F. A. Bishop, Variable step (VS) adaptive filter algorithm, IEEE Trans. Acoustics, Speech, and Signal Processing 34 (1986) 309–316.

[9] R. H. Kwong, E. W. Johnston, A variable step size LMS algorithm, IEEE Trans. Signal Processing 40 (Jul. 1992) 1633–1642.

[10] V. J. Mathews, Z. Xie, A stochastic gradient adaptive filter with gradient adaptive step size, IEEE Trans. Signal Processing 41 (Jul. 1993) 2075–2087.

[11] H. J. Kushner, J. Yang, Analysis of adaptive step-size SA algorithms for parameter tracking, IEEE Trans. Automatic Control 40 (Aug. 1995) 1403–1410.

[12] T. Aboulnasr, K. Mayyas, A robust variable step-size LMS-type algorithm: analysis and simulations, IEEE Trans. Signal Processing 45 (Mar. 1997) 631–639.

[13] S. D. Peters and A. Antoniou, A self-tuning NLMS adaptive filter using parallel adaptation, IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing 44 (Jan. 1997) 11–21.

[14] A. C. Singer, M. Feder, Universal linear prediction by model order weighting, IEEE Trans. Signal Processing 47 (Oct. 1999) 2685–2700.

[15] S. S. Kozat, A. C. Singer, Multi-stage adaptive signal processing algorithms, in: Proc. 2000 IEEE Sensor Array and Multichannel Signal Processing Workshop, 2000, pp. 380–384.

[16] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, G. E. Hinton, Adaptive mixtures of local experts, Neural Computation 3 (Jan. 1991) 79–87.

[17] M. Martínez-Ramón, J. Arenas-García, A. Navia-Vázquez, A. R. Figueiras-Vidal, An adaptive combination of adaptive filters for plant identification, in: A. N. Skodras, A. G. Constantinides (Eds.), Proc. 14th Intl. Conf on DSP, 2002, pp. 1195–1198.

[18] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, New York, NY, 1995.

[19] J. Arenas-García, A. R. Figueiras-Vidal, A. H. Sayed, Steady-state performance of convex combinations of adaptive filters, in: Proc. ICASSP'05, vol. IV, 2005, pp. 33–36.

[20] J. Arenas-García, A. R. Figueiras-Vidal, A. H. Sayed, Mean-square performance of a convex combination of two adaptive filters, IEEE Trans. on Signal Processing (to appear).

[21] A. Benveniste, M. Metivier, P. Priouret, Adaptive Algorithms and Stochastic Approximations, Springer-Verlag, New York, Berlin, 1990.

List of Tables

List of Figures

	LMS	CLMS	CLMS2 $[\eta(n) > \beta]$	CLMS2 $[\eta(n) < \beta]$
Component Filter Adaptation	$2M + 1$	$4M + 2$	$4M + 2$	$6M + 3$
Basic Combination	—	6	6	12
Explicit Weight Calculation	—	$2M$	$2M$	$3M + 2$
Weight Transfer	—	$2M$	$2M$	—

Table 1

Summary of the computational complexity of the combination algorithms measured as the number of real multiplications per iteration.

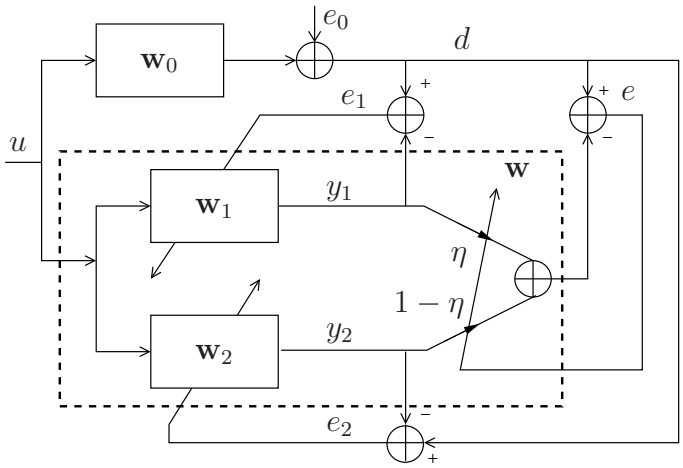
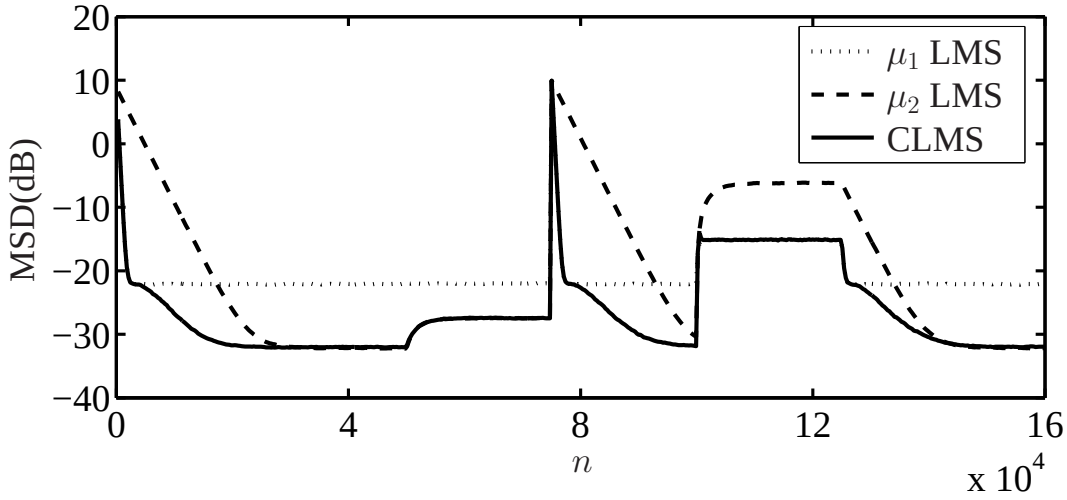
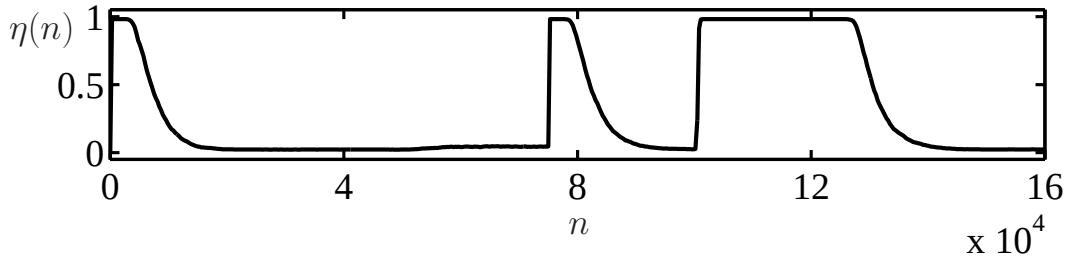


Fig. 1. Plant identification with the proposed adaptive (convex) combination scheme. Two LMS adaptive filters $\mathbf{w}_1$ and $\mathbf{w}_2$, one of them "fast" and the other "slow" (large and small step sizes, respectively) adaptively combine their outputs by means of a very rapidly adapted parameter η in order to identify the plant $\mathbf{w}_0$.

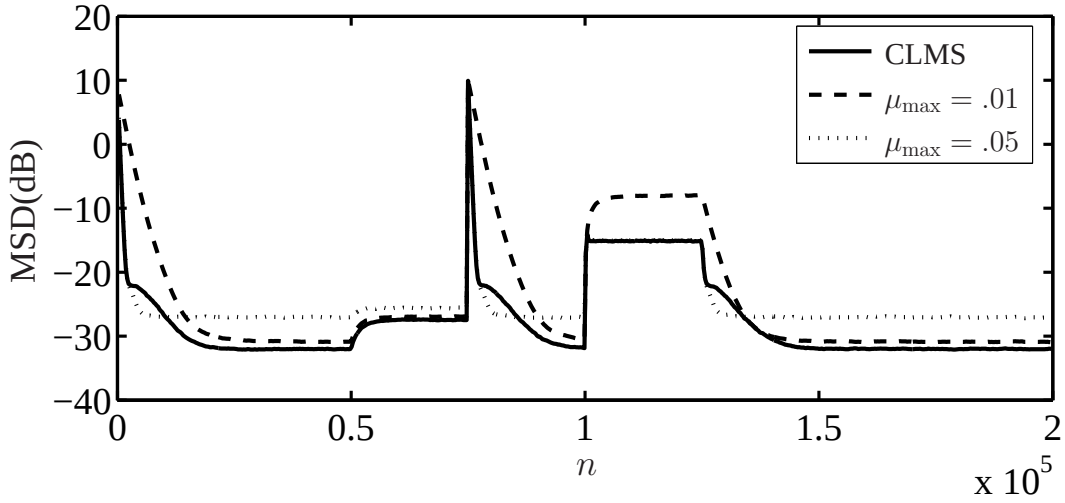


(a)

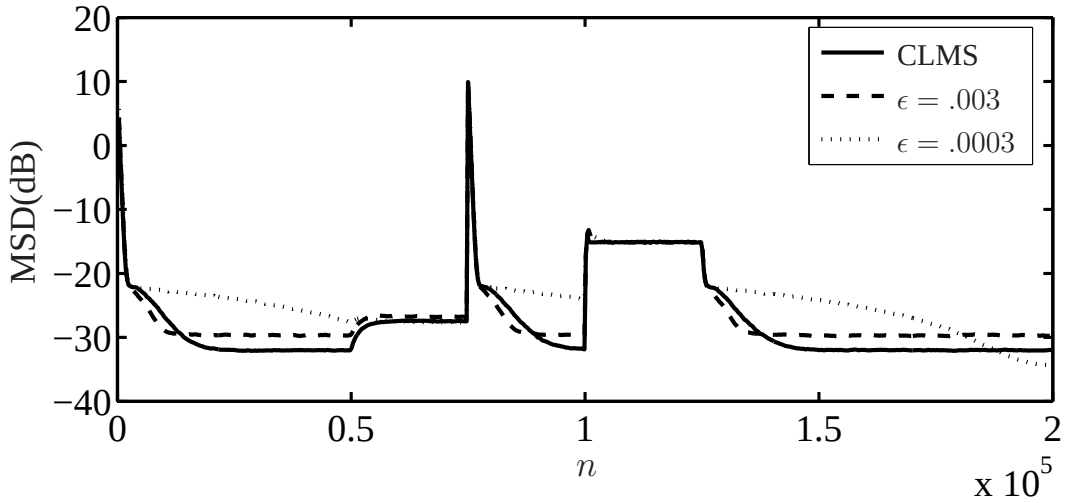


(b)

Fig. 2. Performance of CLMS in a time varying plant identification task. (a) MSDs obtained by a fast LMS filter with step size $\mu_1 = 0.05$ (dotted), by a slow LMS filter with $\mu_2 = 0.005$ (dashed), and by their adaptive combination with parameters $\mu_a = 400$, $\alpha = 0.9$, and $\beta = 0.98$ (solid). (b) Evolution of the mixing parameter.



(a)



(b)

Fig. 3. Comparison of CLMS and AG-LMS performance. In both figures, CLMS ($\mu_1 = 0.05$, $\mu_2 = 0.005$, $\mu_a = 400$, $\alpha = 0.9$, and $\beta = 0.98$) is depicted using a solid line, while the rest stand for AG-LMS with $\mu_{\min} = 0$ and different values of $\mu_{\max}$ and ϵ: (a) fixed $\epsilon = 0.01$ for $\mu_{\max} = 0.01$ (dashed) and $\mu_{\max} = 0.05$ (dotted); (b) fixed $\mu_{\max} = 0.05$ for $\epsilon = 0.003$ (dashed) and $\epsilon = 0.0003$ (dotted).

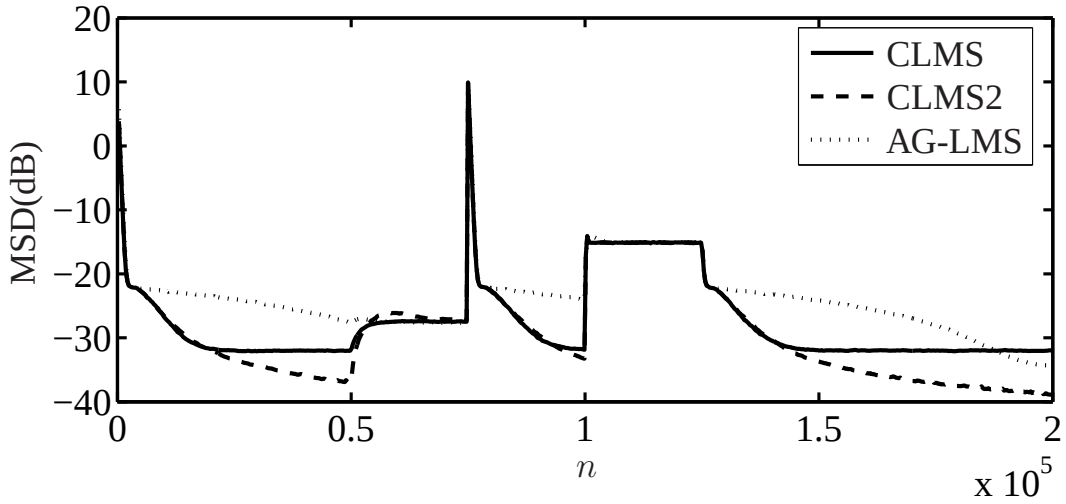


Fig. 4. MSDs achieved by a CLMS filter ($\mu_1 = 0.05$, $\mu_2 = 0.005$, $\mu_a = 400$, $\alpha = 0.9$, and $\beta = 0.98$), a CLMS2 filter (same settings, plus $\mu_{a2} = 10\mu_a$ and $r = 2$), and an AG-LMS filter ($\mu_{\min} = 0$, $\mu_{\max} = 0.05$, and $\epsilon = 0.0003$), respectively represented by solid, dashed, and dotted lines.

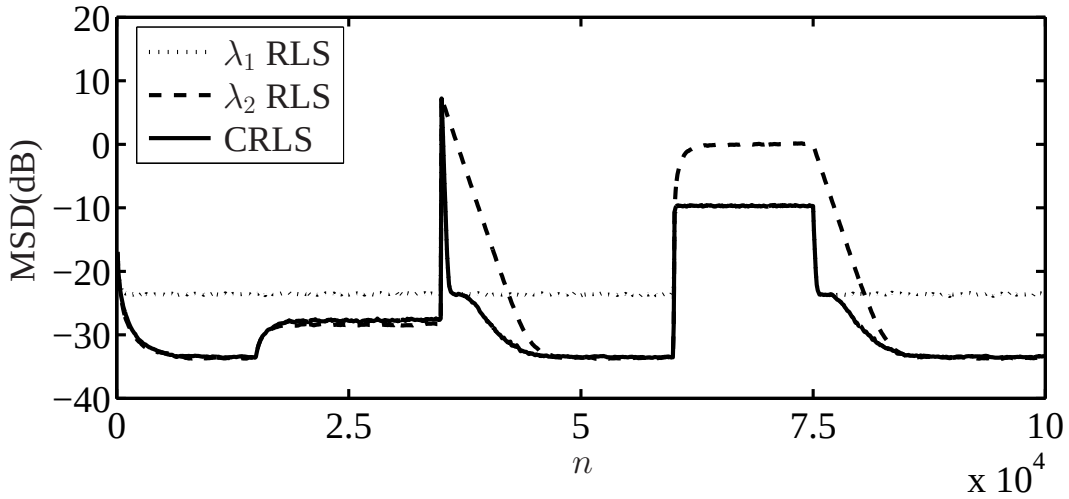


Fig. 5. Performance of a convex combination of two RLS filters in a time varying plant identification setup. MSD of the RLS components is depicted using dotted and dashed lines for $\lambda_1 = 0.995$ and $\lambda_2 = 0.9995$, respectively, while the MSD achieved by their combination is represented with a solid line.