Variable-length adaptive algorithms for sparse filters

Vítor H. Nascimento

Abstract— **Despite their qualities of robustness, low cost, and good tracking performance, for many applications the convergence of the LMS and the normalized LMS algorithms is too slow. Here we analyze a variation of a recently proposed method that speeds up this convergence rate by varying the length of the adaptive filter, taking advantage of the faster convergence rates obtained by short filters. Our results show that variable-length adaptive filters may improve the initial convergence rate of sparse filters when the correlation of the regressor input is not too high.**

We also provide simulations comparing the new algorithm with NLMS and sparse-signed LMS.

Keywords— **Sparse adaptive filters, convergence rate, LMS algorithm.**

I. Introduction

As is well known, the least-mean-square algorithm (LMS) has several desirable properties, which explain its widespread use: LMS is easily implemented, has a low computational cost, is robust to numerical errors, and has reasonable tracking performance. Its major drawback is its slow initial convergence, especially in situations where there is strong correlation between the entries of the regressor vector [1], [6].

In order to take better advantage of the qualities of LMS, several methods to speed up its convergence have been proposed, including the recent proportionate normalized LMS (PNLMS) [2] and sparse-signed LMS (SSLMS) [3] algorithms, that take advantage of sparsity in the optimal filter sought.

In this paper we give further results on another approach: variable-length filters, initially proposed by us in [4]. This approach can substantially accelerate the initial convergence of the LMS and even of the PNLMS and SSLMS algorithms. We present here a different strategy for filter length variation than the one proposed in [4], provide a new analysis showing the conditions under which our method achieves good results, and present simulations comparing our method with the normalized-LMS and SSLMS algorithms.

The additional cost in using the variable-length strategy described here includes both a larger program memory and a larger number of operations, and depends on the choice of the minimum sub-filter length. Depending on the choices made, this additional cost may be small when compared with the complexity of LMS.

A. Definitions and notation

We need approximations to a *desired sequence* $\{y(n) \in \mathbb{R}\}_{n=0}^{\infty}$ through linear combinations of the *regressor (column) vector sequence* $\{\boldsymbol{x}_n \in \mathbb{R}^M\}_{n=0}^{\infty}$, such that $\hat{y}(n) = \boldsymbol{w}_n^T\boldsymbol{x}_n$ (the superscript ^{T} denotes vector transposition). Both sequences are assumed zero-mean. The vector sequence $\{\boldsymbol{x}_n\}$ is formed from a scalar sequence $\{x(n) \in \mathbb{R}\}_{n=0}^{\infty}$, such that

$$\boldsymbol{x}_n = \begin{bmatrix} x(n) & x(n-1) & \dots & x(n-M+1)\end{bmatrix}^T.$$

The weight vector $\boldsymbol{w}_n$ is computed iteratively as (the positive constant μ is known as the *step-size*):

$$w_{n+1,i} = w_{n,i} + \mu e(n) x_{n,i} g[\boldsymbol{x}_n, \boldsymbol{w}_n],$$

where $w_{n,i}$ and $x_{n,i}$ are the i-th elements of each vector, $e(n) = y(n) - \boldsymbol{w}_n^T\boldsymbol{x}_n$, and

$$g[\boldsymbol{x}_n, \boldsymbol{w}_n] = \begin{cases} 1 & \text{for LMS,} \\ 1/(\epsilon_1 + \boldsymbol{x}_n^T\boldsymbol{x}_n) & \text{for NLMS,} \\ \epsilon_2 + \text{abs}(\boldsymbol{w}_{n,i}) & \text{for SSLMS,} \end{cases}$$

with small positive constants ϵ_1 and ϵ_2.

If the desired and regressor sequences are stationary (an assumption we make in this paper), and their second order statistics are known, it is possible to compute the optimal filter weights $\boldsymbol{\Omega}$ that minimize $\mathrm{E}\, e(n)^2$ ($\mathrm{E}(\cdot)$ is the expectation operator). Defining

$$R \overset{\Delta}{=} \mathrm{E}(\boldsymbol{x}_n\boldsymbol{x}_n^T) \quad \text{and } \boldsymbol{p} \overset{\Delta}{=} \mathrm{E}(y(n)\boldsymbol{x}_n),$$

we have $\boldsymbol{\Omega} = R^{-1}\boldsymbol{p}$ [1].

II. Variable-length NLMS algorithm

We can achieve a faster convergence rate if we split a length-M filter into several sub-filters of smaller length. These sub-filters are trained independently, and, since they have smaller lengths, their adaptation rate is faster than that of a full filter [4]. After a few iterations, the sub-filters are merged, with the training re-starting from a better initial condition than the original one. Several strategies for splitting, merging, and passing on the initial conditions to the merged filters are possible. The strategy described here is an evolution of the one presented in [4].

Consider a length-M adaptive filter. We begin (Stage 1) with K_1 length-M_1 sub-filters ($M = K_1M_1$), updated independently through the NLMS algorithm (in fig. 1 we have an example with $M = 4$, $M_1 = 1$, and $K_1 = 4$). After a pre-chosen number N_1 of iterations, we merge the length-M_1 sub-filters, forming $K_1/2$ length-$2M_1$ sub-filters (Stage 2). Again, after N_2 iterations, we merge these two sub-filters, and so on (see also [4]).

As seen in fig. 1, sub-filter j in stage k generates its own error, $e_j^k(n)$. We keep track of these errors to choose

Depto de Engenharia de Sistemas Eletrônicos Escola Politécnica, Universidade de São Paulo. E-mail: vitor@lps.usp.br. This work was supported in part by FAPESP (00/09569-6) and CNPq (300739/00-1).

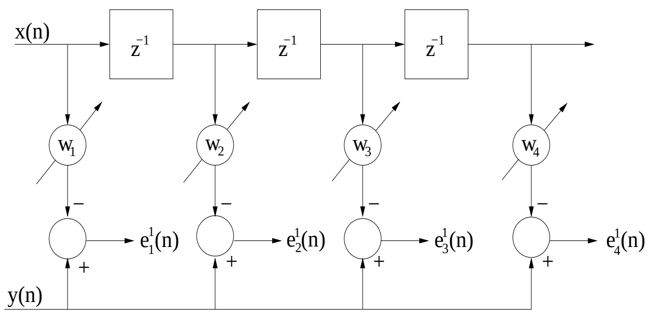

Fig. 1. *Stage1: 4 independent length-1 sub-filters forming a length-4 filter.*

the overall filter output, as follows. Choose $\lambda \in (0,1)$ and define, for $0 \le n < N_1 - 1$,

$$\boldsymbol{\sigma}_{n+1}^1 = \lambda \boldsymbol{\sigma}_n^1 + \left[\left(e_1^1(n)\right)^2 \quad \dots \quad \left(e_{K_1}^1(n)\right)^2 \right]^T, \qquad (1)$$

with initial condition $\boldsymbol{\sigma}_0^1 = \mathbf{0}$.

The overall output error $e(n)$ is chosen as the $e_{I(n)}^1(n)$ corresponding to the minimum entry of $\boldsymbol{\sigma}_n^1$. Denoting the i-th entry of $\boldsymbol{\sigma}_n^1$ by $\sigma_i^1(n)$, $e(n)$ and $I(n)$ are given by

$$I(n) = \arg\min_i \sigma_i^1(n), \qquad e(n) = e_{I(n)}^1(n). \qquad (2)$$

In the transition between two stages we need to keep the coefficients from the best sub-filter and zero the coefficients of all other sub-filters, otherwise there might appear a large spike at the estimation error $e(n)$ (see Sec. III and [5] for an explanation of this behavior).

The transition stragety described here is slightly different than that of [4]. For example, at the transition from stage 1 to stage 2 (at step N_1), we find the two minimum values of $\boldsymbol{\sigma}_{N_1-1}^1$ (corresponding to sub-filters i and j, say). In the second stage, the i-th and j-th sub-filters are merged together, i.e., the entries of $\boldsymbol{w}$ corresponding to the i-th and j-th sub-filters will be updated as a single NLMS filter. The other sub-filters are merged with their neighbors (see fig. II, where the smallest and second smallest entries of $\boldsymbol{\sigma}_{N_1-1}^1$ are the 2nd and 3rd, respectively).

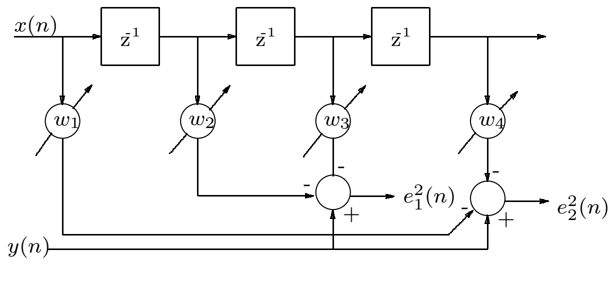

Fig. 2. Stage 2: 2 length-2 sub-filters

The second stage will therefore have $K_2 = K_1/2$ sub-filters, each with length $M_2 = 2M_1$. Their initial conditions are zero, except for the sub-filter corresponding to the best stage-1 sub-filter (for example, in the $M = 4$ filter in figures 1 and II, if $\boldsymbol{\sigma}_{N_1-1}^1 = [\,.8\ .01\ .7\ .9\,]^T$, all coefficients, except w_2, will be set to zero in step N_1). In our $M = 4$ example, the initial condition $\boldsymbol{\sigma}_{N_1-1}^2$ for stage 2 is as follows:

$$\boldsymbol{\sigma}_{N_1-1}^2 = \begin{bmatrix} \sigma_1^2(N_1-1) \\ \sigma_2^2(N_1-1) \end{bmatrix} = \begin{bmatrix} \min\limits_{i=2,3} \sigma_i^1(N_1-1) \\ \min\limits_{i=1,4} \sigma_i^1(N_1-1) \end{bmatrix} = \begin{bmatrix} .01 \\ .8 \end{bmatrix}. \qquad (3)$$

The procedure for the other stages is similar.

Note that at the last stage the algorithm returns to a standard NLMS recursion, and thus the steady-state, tracking, and robustness properties of NLMS are retained. The only modification occurs during the initial convergence.

Our example in figs. 1–II starts with length-1 sub-filters for convenience of presentation. In general, it would be a better policy to begin with longer sub-filters, for two reasons:

a) the algorithm will be more likely to choose the best filters to merge at the end of the first stage;
b) the computational complexity is reduced.

In fact, if we start with K_1 length-M_1 sub-filters, the extra work (*in addition* to the $3M+1$ multiplications, $3M$ additions and 1 division required by NLMS) that is needed at each step is:

i) $K_1 - 1$ comparisons to choose the the smallest entry of the length-K_1 vector $\boldsymbol{\sigma}_n^1$ (at the transition between the first two stages, the two smallest entries are needed, and finding them might require $2.5K_1 - 4$ comparisons);
ii) $K_1 - 1$ multiplications (to compute $\mu e_i^1(n)$; in NLMS we only need to compute $\mu e(n)$);
iii) $2(K_1 - 1)$ additions (to compute the $e_i^1(n)$ and the K_1 normalizing factors for the NLMS algorithm applied to each sub-filter),
iv) $K_1 - 1$ divisions,
v) $2K_1$ multiplications and K_1 additions to compute $\boldsymbol{\sigma}_{n+1}^k$.

This gives a total of $2.5K_1 - 4$ comparisons, $3K_1 - 1$ multiplications, $3K_1 - 2$ additions, and $K_1 - 1$ divisions in addition to the operations needed by plain NLMS. For the other stages, since the number of sub-filters will be smaller, the operation count will also be smaller.

III. Influence of sparsity and eigenvalue dispersion

The key to good performance of the variable-length algorithm is that the correct sub-filters are chosen at the transitions between stages. In this section we show how the sparsity of the optimum weight vector $\boldsymbol{\Omega}$ and the eigenvalue dispersion of the input autocorrelation matrix $R = \mathrm{E}\, \boldsymbol{x}_n \boldsymbol{x}_n^T$ influence the performance of the variable-length algorithm, affecting the probability of making wrong choices. Assume then that at stage 1 we have K_1 sub-filters of length M_1, for a total filter length $M = K_1 M_1$, and split $\boldsymbol{\Omega}$ into K_1 sub-vectors, such that

$$\boldsymbol{\Omega}^T \triangleq \left[\boldsymbol{\Omega}_1^T \quad \boldsymbol{\Omega}_2^T \quad \dots \quad \boldsymbol{\Omega}_{K_1}^T \right].$$

Similarly, split R as (recall that R is a Toeplitz matrix, and note that $R_k = R_{-k}^T$)

$$R \triangleq \mathrm{E}\, \boldsymbol{x}_n \boldsymbol{x}_n^T = \begin{bmatrix} R_0 & R_1 & \dots & R_{M_1-1} \\ R_{-1} & R_0 & \dots & R_{M_1-2} \\ \vdots & \vdots & \ddots & \vdots \\ R_{1-M_1} & R_{2-M_1} & \dots & R_0 \end{bmatrix}.$$

In [5] we prove that under these conditions, each sub-filter has optimal weights $\boldsymbol{\Omega}_i^s$ with minimum error $J_{0,i}^s$ given by

$$\boldsymbol{\Omega}_i^s = \boldsymbol{\Omega}_i + R_0^{-1} \sum_{k=1, k\neq i}^{K_1} R_{k-i} \boldsymbol{\Omega}_i^s \tag{4}$$

$$J_{0,i}^s = \sigma_0^2 + \sum_{k=1, k\neq i}^{K_1} \boldsymbol{\Omega}_k^{s\,T} R_0 \boldsymbol{\Omega}_k^s, \tag{5}$$

where $\sigma_0^2 = \mathrm{E}(y(n) - \boldsymbol{\Omega}^T \boldsymbol{x}_n)^2$.

To begin our discussion, assume that $x(n)$ is a white process, so that R is diagonal and $R_i = 0$ for $i \neq 0$. In this case, (4) implies that $\boldsymbol{\Omega}_i^s = \boldsymbol{\Omega}_i$, i.e., the overall filter obtained from the independent sub-filters with optimum weights is exactly the optimum length-M filter.

Let us assume also that $\boldsymbol{\Omega}$ is sparse, i.e. that for some small integer p, only the sub-filters corresponding to indices $i \in \mathcal{I} = \{i_1, i_2, \dots, i_p\}$ are nonzero (i.e., if $i \notin \mathcal{I}$, then $\boldsymbol{\Omega}_i = \mathbf{0}$). Under these conditions, it is easy to see that, whenever $i \notin \mathcal{I}$ and $j \in \mathcal{I}$, then $J_{0,i}^s > J_{0,j}^s$. In fact, since in this case $R_0 = \sigma_x^2 I$, whenever i and j are such that $\|\boldsymbol{\Omega}_i^s\| > \|\boldsymbol{\Omega}_j^s\|$, then $J_{0,i}^s > J_{0,j}^s$ ($\|\boldsymbol{\Omega}\|$ is the Euclidean norm of $\boldsymbol{\Omega}$).

A reasonable approximation to the steady-state mean-square errors (MSEs) of a sub-filter running NLMS is, for small step-size μ [7],

$$J_i^s \approx J_{0,i}^s \left[1 + \frac{\mu\, \mathrm{Tr}(R_0)}{2}\, \mathrm{E}\left(\frac{1}{\|\bar{\boldsymbol{x}}_{i,n}\|^2} \right) \right],$$

where

$$\bar{\boldsymbol{x}}_{i,n} = \begin{bmatrix} x(n-(i-1)M_1) & \dots & x(n-iM_1-1) \end{bmatrix}^T$$

and $\mathrm{Tr}(R_0)$ is the trace of matrix R_0. From the above expression for the MSE, we see that for small step-size, J_i^s is approximately equal to $J_{0,i}^s$.

From the above considerations, we conclude that (when $\boldsymbol{\Omega}$ is sparse and if R_i becomes small as i becomes larger), we can find the sub-filters that correspond to the largest entries in $\boldsymbol{\Omega}$ by finding the sub-filters with the smallest $J_{0,i}^s$. Of course, as one can see from (4) and (5), as the input correlation increases or the sparsity of $\boldsymbol{\Omega}$ decreases, the MSEs of the sub-filters will become worse indicators for the nonzero entries of $\boldsymbol{\Omega}$.

We will show this behavior through an example. Consider a filter with length $M = 128$, for which the only nonzero entries of $\boldsymbol{\Omega}$ are in positions 1, 10, 11, and 120 (all these entries are equal to 1). The input sequence $x(n)$ is generated from an autoregressive (AR) model, such that

$$x(n+1) = \alpha x(n) + \varepsilon(n),$$

where $\varepsilon(n)$ is a white Gaussian noise.

We computed the values of $J_{0,i}^s$ for $K_1 = 32$ sub-filters of length $M_1 = 4$, for $\alpha = 0.5$, 0.9, and 0.95 (fig. 3). The eigenvalue dispersions, $\lambda_{\max}(R)/\lambda_{\min}(R)$, are 9.0, 347, and 1337, respectively. The sub-filters with nonzero coefficients are the first, third, and 30th. One can see that at these sub-filters the MSE is indeed smaller than at adjacent sub-filters, but the difference becomes less pronounced as the input autocorrelation increases (thus decreasing the probability of choosing the correct sub-filters at stage transitions).

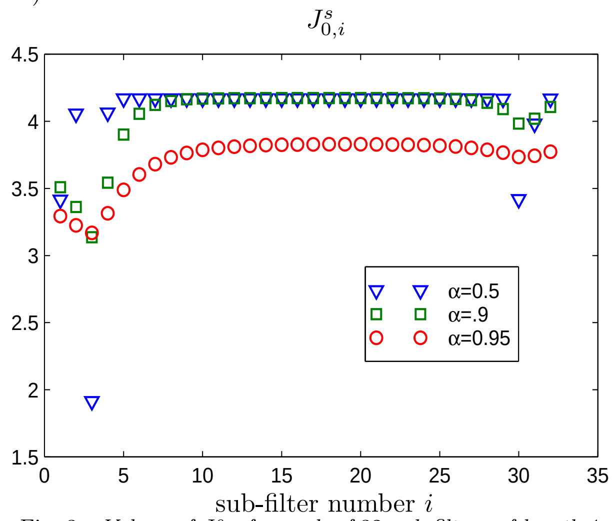

Fig. 3. *Values of* $J_{0,i}^s$ *for each of* 32 *sub-filters of length* 4.

IV. Simulations

We now present a few simulations comparing the variable-length approach with the NLMS and the SSLMS algorithms. The first example is a filter with total length $M = 2048$, whose input $x(n)$ is the output of a filter with white Gaussian (unit-variance) input, and transfer function

$$H(z) = \frac{1 + 0.7z^{-1}}{1 - 0.5z^{-1}}.$$

The NLMS and VLNLMS algorithms are used to identify a system with output

$$y(n) = \boldsymbol{\Omega}^T \boldsymbol{x}_n + e_0(n),$$

where $e_0(n)$ is a white Gaussian noise with variance $\sigma_0^2 = 10^{-4}$, and $\boldsymbol{\Omega}$ is obtained as the convolution of a vector of length 2048, whose only nonzero coefficients are at positions

$$1, \quad 100, \quad 111, \quad 1769, \quad \text{and} \quad 1800,$$

with amplitudes 1, 0.8, 0.7, -0.5, and 0.9; and a vector of length 21 whose k-th entry is given by $0.3^{|k-11|}$ (the first 10 and last 11 elements of the convolution are thrown away, so that the final vector has again length 2048).

We applied the NLMS algorithm with step-size $\mu = 1$ (the condition for maximum convergence speed for NLMS), and VLNLMS with transitions at points $N_i = \{100, 200, 300, 400, 600, 1200, 2400, 3600, 4800, 6400, \text{and } 9600\}$.

The step-sizes were chosen as follows: for the first stage (128 sub-filters with length 16), $\mu_1 = 0.02$, for the second stage, $\mu_2 = 0.1$, third stage, $\mu_3 = 0.2$, and for the remaining stages, $\mu_i = 1$ (so the last stage reverts to an NLMS filter with $\mu = 1$). For the computation of $\boldsymbol{\sigma}_n^k$ in (1), we used $\lambda = 0.999$.

The average of 5 simulations with initial condition $\boldsymbol{w}_0 = \mathbf{0}$ is presented in fig. 4. We see that VLNLMS reaches a point of reasonable MSE (between 0.1–0.01) in approximately half the number of iterations needed by NLMS.

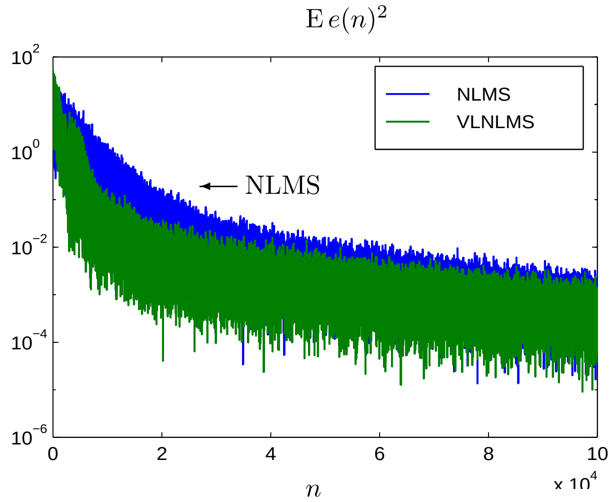

Fig. 4. *MSE for NLMS and VLLMS, average of 5 runs.*

The computational complexity of $NLMS$ for this example was 6,144 multiplications, 6,144 additions, and 1 division. For VLLMS, in the worst case (first stage), we have, in addition to the operations required by NLMS, 316 comparisons, 383 multiplications, 382 additions, and 127 divisions. Therefore, VLLMS would be approximately 44% slower than NLMS (considering that multiplications and additions can be performed simultaneously in one cycle, that comparisons have the same cost as additions and multiplications, and that divisions take 16 cycles in a digital signal processor).

For our next example, we compared the variable-length approach with SSLMS. Our filter $\boldsymbol{\Omega}$ had again $M = 2048$ coefficients, with nonzero entries only at taps 1, 600, 700, and 1700, and amplitudes of 0.1, 1, -0.5, and 0.1, respectively. The input regressor $x(n)$ was obtained as in the previous example, but the measurement noise had variance $\sigma_0^2 = 10^{-2}$. For this example, we chose the transition points at $N_i = \{300, 900, 1200, 1500, 1800, \text{and } 2100\}$, and again we started from a zero initial condition, with sub-filters with 16 coefficients. The step-sizes were 1 for the NLMS algorithm and 0.002 for the SSLMS algorithm. The first 3 stages of the variable-length algorithm employed NLMS, with step-sizes 0.02, 0.1, and 0.4. From the fourth stage on, the sub-filters used the SSLMS recursion, with step-sizes 0.02, 0.01, 0.005, 0.003, and 0.002. The average of 50 runs can be seen in fig. 5. The figure shows that again the variable-length algorithm reaches steady-state much faster than NLMS, and even than SSLMS. Since the first three stages in our filter used NLMS, the computational count for the worst case is as in the last example.

We stress that in both examples, we ran NLMS with step-size $\mu = 1$, which is a good approximation for the condition of maximum convergence speed for this algorithm. Therefore, we would not achieve faster convergence with NLMS even if we had used a variable step-size approach.

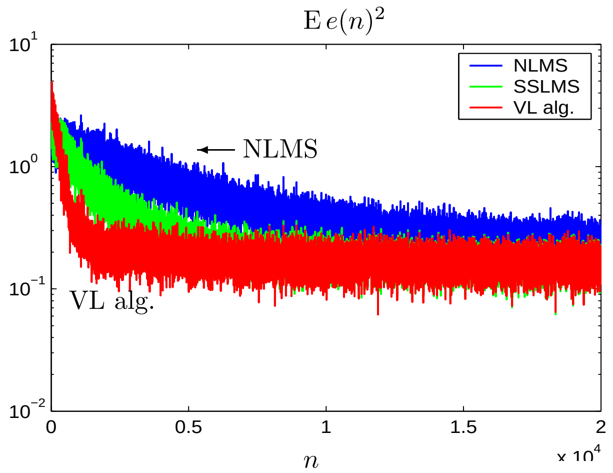

Fig. 5. *MSE for NLMS, SSLMS, and VLLMS – average of 50 runs.*

V. Conclusions

In this paper we studied the performance of variable-length algorithms, concluding that variable-length methods should give good performance for the identification of sparse filters, as long as the correlation of the input signal $x(n)$ is not too high. We compared the behavior of a variable-length NLMS algorithm with NLMS itself, and with the recently proposed sparse-signed LMS algorithm.

We are currently studying ways of improving the performance of the algorithm in choosing the best sub-filters at the end of each stage, and methods to choose on-line the transition points between stages and the step-sizes.

References

[1] P. S. R. Diniz. *Adaptive Filtering: Algorithms and Practical Implementation.* Kluwer Academic Publishers, 1997.

[2] D. L. Duttweiler. Proportionate normalized least-mean-squares adaptation in echo cancellers. *IEEE Transactions on Speech and Audio Processing*, 8(5):508–518, set. 2000.

[3] R. K. Martin, W. A. Sethares, R. C. Williamson, e C. R. Johnson Jr. Exploiting sparsity in adaptive filters. *IEEE Transactions on Signal Processing*, 50(8):1883–1894, ago. 2002.

[4] V. H. Nascimento. Improving the initial convergence of adaptive filters: the variable-length LMS filter. In *Proceedings of the 14th IEEE International Conference on Digital Signal Processing (DSP2002), Santorini, Greece*, volume 2, pages 667–670. IEEE/EURASIP, jun. 2002.

[5] V. H. Nascimento. Analysis of the hierarchical LMS algorithm. *IEEE Signal Processing Letters*, 10(3):78–81, mar 2003.

[6] A. H. Sayed. *Fundamentals of Adaptive Filtering.* Wiley-Interscience, 2003.

[7] D. T. M. Slock. On the convergence behavior of the LMS and the normalized LMS algorithms. *IEEE Transactions on Signal Processing*, 41(9):2811–2825, set. 1993.