IMPROVING THE INITIAL CONVERGENCE OF ADAPTIVE FILTERS: VARIABLE-LENGTH LMS ALGORITHMS

Vítor H. Nascimento

Electronic Systems Eng. Dept., Escola Politécnica, Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, nº 158, CEP 05508-900 — São Paulo, SP, Brazil
vitor@lps.usp.br

Abstract: Despite its qualities of robustness, low cost, and good tracking performance, in many situations the LMS algorithm suffers from slow initial convergence. We propose a method to speed up this convergence rate by varying the length of the adaptive filter, taking advantage of the larger step-sizes allowed for short filters. The results presented here show that variable-length adaptive filters have the potential to achieve quite fast convergence rates, with a modest increase in the computational complexity.

1. INTRODUCTION

The least-mean square algorithm (LMS) is probably the most widely used adaptive filtering algorithm, due to its low computational cost, robustness [5], and good tracking performance (which is comparable to that of the recursive least-squares algorithm, RLS) [4]. The major drawback of LMS is surely its slow initial convergence, especially in situations where there is strong correlation between the entries of the regressor vector [3, 6].

In order to take better advantage of its qualities, several methods to speed up the LMS convergence have been proposed, most notably the NLMS algorithm (normalized-LMS) [1, 7, 9], and transform-domain algorithms [6]. These methods achieve faster convergence by reducing the dispersion of the eigenvalues of R (the autocorrelation matrix of the input vector sequence, see Sec. 1.1 below). As is well known [3, 6], the best condition for fast LMS convergence is if R is a multiple of the identity. When the number of taps used is large, however, even in this ideal situation the convergence may be slow: it is known that the maximum allowable step-size for stable LMS behavior is inversely proportional to the filter length (for independent, Gaussian regressors a proof for this can be found in [10], and for more general regressors, in [2]). This constraint on the step-size forces slower convergence as the filter length is increased.

In this paper we describe a new approach that can substantially accelerate the initial convergence of the LMS and even of the NLMS algorithms. Contrary to what is normally done, we do not attempt to reduce the eigenvalue dispersion of R. Instead, we propose a way of taking advantage of the superior convergence rates attainable by *short* filters, employing variable-length filters.

We present here one possible strategy, leading to a simple variable-length LMS algorithm. Other, more complex strategies are possible and are being studied. We show that our variable-length filter is less sensitive to eigenvalue dispersion, although it is sensitive to the relative values of the optimum (Wiener) weights, as explained further on.

The additional cost in using the variable-length strategy described here includes both a larger program memory and a larger number of operations (up to $2M$ multiplications and M comparisons beyond what is required by LMS, where M is the filter length, depending on the how the algorithm is implemented.)

In the remainder of this section, we describe our notation, and remind the reader of expressions relating convergence rate, eigenvalue dispersion, and filter length. In the next sections, we describe our variable-length LMS algorithm, comment on its behavior, and present a few simulations to illustrate the analysis.

1.1. Definitions and notation

Recall that the LMS algorithm computes 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 usually 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*):

$$\begin{aligned} \boldsymbol{w}_{n+1} &= \boldsymbol{w}_n + \mu e(n) \boldsymbol{x}_n, \quad \text{where} \\ e(n) &= y(n) - \boldsymbol{w}_n^T \boldsymbol{x}_n. \end{aligned}$$

If the desired and regressor sequences are stationary, and their auto- and cross-correlations are known, it is possible to compute the filter weights that minimize the expected value of $e(n)^2$ (these optimum weights are known as the *Wiener solution*), as follows: define the autocorrelation of $\boldsymbol{x}_n$ and the cross-correlation between $\boldsymbol{x}_n$ and $y(n)$ as $R \triangleq \mathrm{E}(\boldsymbol{x}_n \boldsymbol{x}_n^T)$, and $\boldsymbol{p} \triangleq \mathrm{E}(y(n)\boldsymbol{x}_n)$, respectively ($\mathrm{E}(\cdot)$ represents the expectation operator). Then, $\boldsymbol{w}_o = R^{-1}\boldsymbol{p}$ [3].

1.2. Convergence rate and filter length

The behavior of LMS is strongly dependent on R — in particular, it is well known [3, 6] that the convergence speed of LMS depends on the dispersion of the eigenvalues of R, which can be measured by the ratio

$$\kappa(R) \triangleq \frac{\lambda_{\max}(R)}{\lambda_{\min}(R)}.$$

The convergence speed is fastest when $\kappa(R) = 1$, and decreases as $\kappa(R) \to \infty$.

Even in the best situation, when R is a multiple of the identity and $\kappa(R) = 1$, the convergence rate will be considerably slow if the filter is long. As an example, consider the case where the regressor sequence $\{\boldsymbol{x}_n\}$ is *independent, identically distributed* (iid) and has a normal distribution, with $R = \mathrm{E}\,\boldsymbol{x}_n\boldsymbol{x}_n^T = \sigma_x^2 I$. In this idealized situation, it is possible to exactly evaluate the learning curve of the algorithm (the graph of the mean-square error, or MSE, i.e., $\mathrm{E}\,e(n)^2$) [10, 8], and it can be shown that the fastest convergence rate (in the mean-square sense) is achieved by choosing

$$\mu = \frac{1}{(M+2)\sigma_x^2}. \qquad (1)$$

The time constant of the algorithm will then be [3]

$$\tau = \left| \frac{1}{\ln\left(1 - \frac{1}{(M+2)\sigma_x^2}\sigma_x^2\right)} \right| = \frac{1}{\ln\left(\frac{M+2}{M+1}\right)}. \qquad (2)$$

For large M, the time constant is well approximated by $\tau \approx M$, while for $M = 1$, $\tau \approx 2.5$. Note that even in this case, in which the entries of $\boldsymbol{x}_n$ are mutually independent, the step-sizes (and thus the maximum convergence rate) must be reduced as the filter length is increased.

2. VARIABLE-LENGTH (VL LMS) ALGORITHM

We can achieve a faster convergence rate by splitting a length-$M = 2^K$ filter into several sub-filters of smaller length. These sub-filters are trained independently, and, since they have smaller lengths, they can be updated using larger step-sizes, allowing faster convergence rates. 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: different specific applications will require different strategies. We describe here a straightforward strategy, in order to demonstrate the potential of the variable-length approach.

Let us consider then a length-4 LMS filter. We shall begin (Stage 1) with 4 length-one sub-filters updated independently (see Fig. 1). After a pre-chosen number N_1 of iterations, we merge the length-one sub-filters, forming 2 length-2 sub-filters (Stage 2, see Fig. 2). Again, after N_2 iterations, we merge these two sub-filters, and obtain a standard, length-4 LMS filter (Stage 3, see Fig. 3). In each stage we use a *different* step-size μ_k, and we should choose $\mu_1 > \mu_2 > \mu_3$. As seen in Figs. 1–3, sub-filter

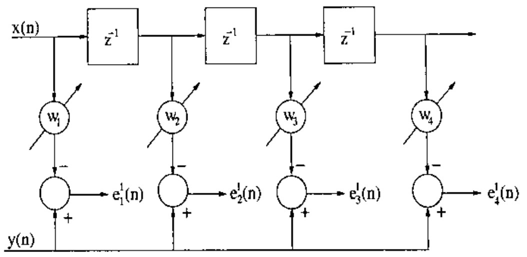

Fig. 1. *Length-one sub-filters (Stage 1).*

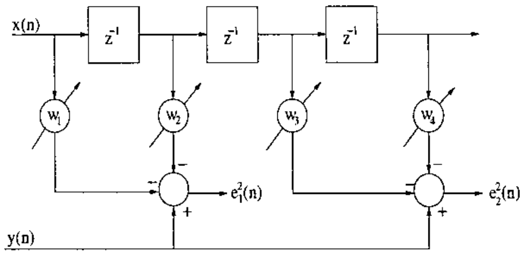

Fig. 2. *Length-two sub-filters (Stage 2).*

j in stage k generates its own error, $e_j^k(n)$. The update equations for each stage are as follows: define the vectors $\boldsymbol{e}_n^1 \in \mathbb{R}^4$, $\boldsymbol{e}_n^2 \in \mathbb{R}^2$, and $\boldsymbol{e}_n^3 \in \mathbb{R}$, where

$$\boldsymbol{e}_n^1 = \begin{bmatrix} e_1^1(n) \\ e_2^1(n) \\ e_3^1(n) \\ e_4^1(n) \end{bmatrix}, \quad \boldsymbol{e}_n^2 = \begin{bmatrix} e_1^2(n) \\ e_2^2(n) \end{bmatrix}, \quad \boldsymbol{e}_n^3 = e(n).$$

Then the update equations for stage 1 ($0 \le n < N_1 - 1$) are ($w_i(n)$ is the i-th entry of vector $\boldsymbol{w}_n$)

$$\begin{bmatrix} w_1(n+1) \\ w_2(n+1) \\ w_3(n+1) \\ w_4(n+1) \end{bmatrix} = \begin{bmatrix} w_1(n) \\ w_2(n) \\ w_3(n) \\ w_4(n) \end{bmatrix} + \mu_1 \begin{bmatrix} e_1^1(n)x(n) \\ e_2^1(n)x(n-1) \\ e_3^1(n)x(n-2) \\ e_4^1(n)x(n-3) \end{bmatrix},$$

and for stage 2 ($N_1 - 1 \le n < N_2 - 1$)

$$\begin{bmatrix} w_1(n+1) \\ w_2(n+1) \\ w_3(n+1) \\ w_4(n+1) \end{bmatrix} = \begin{bmatrix} w_1(n) \\ w_2(n) \\ w_3(n) \\ w_4(n) \end{bmatrix} + \mu_2 \begin{bmatrix} e_1^2(n)x(n) \\ e_1^2(n)x(n-1) \\ e_2^2(n)x(n-2) \\ e_2^2(n)x(n-3) \end{bmatrix},$$

For stage 3, we have a standard LMS recursion.

We now describe how to choose the overall output er-

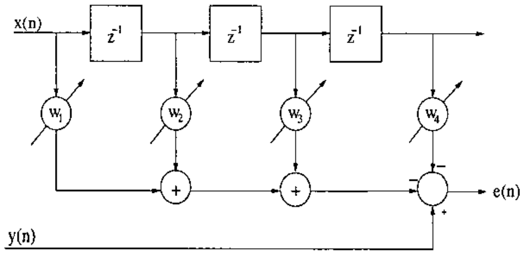

Fig. 3. *Length-four LMS filter (Stage 3).*

ror $e(n)$ in stages 1 and 2. Choose $\lambda \in (0,1)$ and define

$$\sigma^1_{n+1} = \lambda\sigma^1_n + (1-\lambda)\begin{bmatrix}(e^1_1(n))^2\\(e^1_2(n))^2\\(e^1_3(n))^2\\(e^1_4(n))^2\end{bmatrix},\ 0 \le n < N_1 - 1,$$

$$\sigma^2_{n+1} = \lambda\sigma^2_n + (1-\lambda)\begin{bmatrix}(e^2_1(n))^2\\(e^2_2(n))^2\end{bmatrix},\ N_1 \le n < N_2 - 1,$$

with $\sigma^1_0 = \mathbf{0}$, and $\sigma^2_{N_1}$ given below — sec (4). In stage k, the overall output error $e(n)$ is chosen as the $e^k_{J(n)}(n)$ corresponding to the minimum entry of σ^k_n. Denoting the i-th entry of σ^k_n by $\sigma^k_i(n)$, $e(n)$ and $J(n)$ are given by

$$J(n) = \arg\min_i \sigma^k_i(n), \qquad e(n) = e^k_J(n). \quad (3)$$

The transition between stages can be acomplished in several ways. A possibility that gives smooth transitions is the following: At step N_1, zero out all entries of $\boldsymbol{w}_{N_1}$, *except* the $J(N_1)$-th, *before* computing $\boldsymbol{w}_{N_1+1}$. In addition, choose the initial conditions $\sigma^2_{N_1}$ for stage 2 as follows:

$$\sigma^2_{N_1} = \begin{bmatrix}\sigma^2_1(N_1)\\\sigma^2_2(N_1)\end{bmatrix} = \begin{bmatrix}\min_{i=1\ldots2}\sigma^1_i(N_1)\\\min_{i=3\ldots4}\sigma^1_i(N_1)\end{bmatrix}. \quad (4)$$

Similarly, at step N_2, we zero out the entries $\boldsymbol{w}_{N_2}$ corresponding to all sub-filters, except the $J(N_2)$-th sub-filter. Since in this example stage 3 is already the standard LMS algorithm, σ^3_n is not defined. If a larger number of stages were being used, one should proceed with the choice of $\sigma^3_{N_2}$ as done in (4). This procedure may be easily extended to any filter length of the form $M = 2^K$. Note that, since the last stage always returns to a standard LMS filter, the steady-state, tracking, and robustness properties of LMS are retained. The only modification occurs during the initial convergence.

3. SIMULATIONS

In this section, we demonstrate the potential utility of the algorithm above, and the situations in which it may not perform well by means of a few examples (in all of which $\lambda = 0.9$ was used). We consider first the case of iid *vectors* $\boldsymbol{x}_n$, i.e., the situation for which expressions (1)–(2) hold. Let $\{\boldsymbol{x}_n \in \mathbb{R}^{32}\}_{n=0}^{\infty}$ be a zero-mean Gaussian iid sequence, with autocorrelation $R = I$. Let also $y(n) = \boldsymbol{w}_o^T\boldsymbol{x}_n + v(n)$, where $\{v(n)\}_{n=0}^{\infty}$ is zero-mean, Gaussian, iid, independent of $\{\boldsymbol{x}_n\}$, and with variance 10^{-4}. The i-th entry of $\boldsymbol{w}_o$ equal to $e^{-0.3\times(i-1)}$, $1 \le i \le M$. For this $M = 32$ filter, we chose first

$$\begin{aligned}&N_1 = 5, \quad N_2 = 10, \quad N_3 = 30, \quad N_4 = 60,\\&N_5 = 120, \quad N_6 = 200, \qquad\qquad (5)\\&\mu_1 = 1/3, \quad \mu_2 = 1/4, \quad \mu_3 = 1/6, \quad \mu_4 = 1/10,\\&\mu_5 = 1/18, \quad \mu_6 = 1/34.\end{aligned}$$

With these choices, and with initial condition $\boldsymbol{w}_0 = \mathbf{0}$, we obtain the learning curve presented in Fig. 4. The LMS step-size was $\mu = \mu_6$, and the NLMS step-size, $\bar{\mu} = 1$. Note how, since $\kappa(R) = 1$ in this case, the LMS and NLMS curves converge with practically the same speed, although the variable-length algorithm converges much faster. In Fig. 5, one can see the same example, both with

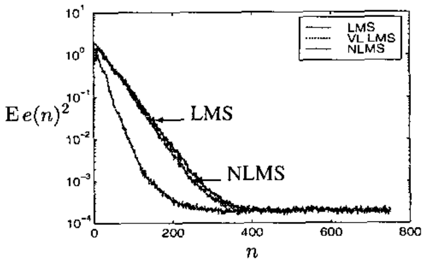

Fig. 4. *MSE* ($\mathrm{E}\,e(n)^2$) *computed for LMS, VL LMS, and NLMS, average of 300 runs.*

$N_i = i \cdot 100$, a choice of transition points that should be avoided in general, but that highlights how the variable-length algorithm works. Remark that at each stage (6 in this case), the algorithm is able to decrease a little further the average error $e(n)$. Note also how the convergence rate decreases as the sub-filter length is increased.

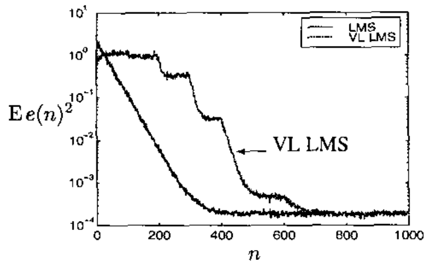

Fig. 5. *MSE* ($\mathrm{E}\,e(n)^2$) *computed for LMS and VL LMS (300 runs). The transition points were chosen to explain how the variable-length algorithm works.*

In the next example, $\boldsymbol{w}_o$ was changed so that its i-th entry is $e^{-0.01(i-1)}$, for $1 \le i \le 32$. The sequences $\{\boldsymbol{x}_n\}$ and $\{v(n)\}$ are the same, and the transition steps are again given by (5). Now this choice gives poor results, as one can see in the upper curve of Fig. 6. Nevertheless, if one rotates the vector $\boldsymbol{x}_n$ (multiplying it by an orthogonal matrix Q, $Q^TQ = I$) prior to application of the LMS and variable-length LMS algorithms, the lower curve is obtained. Note that in both situations, the LMS learning curve is the same — LMS is not affected by orthogonal rotations in this case. The rotation was chosen so that $\boldsymbol{w}_o$ has only one non-zero entry (its 15-th). This example shows that this variable-length strategy is more affected by the relative values of the entries of $\boldsymbol{w}_o$ than by the eigenvalue dispersion of R.

Next, we apply the variable-length algorithm to an equalization example. Now the (length $M = 32$) regressor $\boldsymbol{x}_n$ is formed from a tapped-delay line: a Gaussian iid signal $s(n)$ with variance 1 passes through a channel with z-transform $H(z) = 1/(1 + 0.7z^{-1})$ and is added to a Gaussian noise with variance 10^{-4} to form the entries $x(n)$ of $\boldsymbol{x}_n$. We used the variable-length filter as

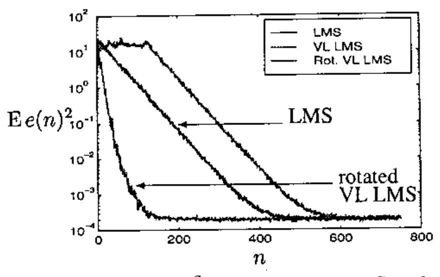

Fig. 6. *MSE* ($\mathrm{E}\,e(n)^2$) *computed for LMS and the new algorithm, with rotated and non-rotated regressor vector (300 runs).*

an equalizer, with the transition points in (5), and with $y(n) = s(n)$. The step-sizes used were $\mu_i = 10^{-2} \times \{0.9,\ 4.5,\ 2.25,\ 1.125,\ 0.9375,\ 0.46875\}$ (Fig. 7).

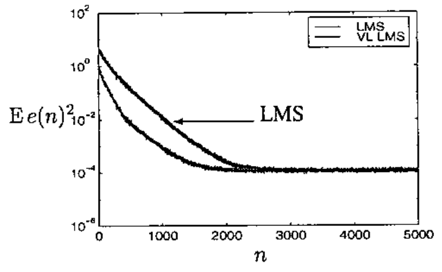

Fig. 7. *MSE* ($\mathrm{E}\,e(n)^2$) *computed for LMS and VL LMS for the first equalizer example (300 runs).*

In Fig. 8, we have the same situation, but the channel is now described by $H(z) = (1 + z^{-1} + 0.9z^{-2}/(1 + 0.7z^{-1})$. This results in a autocorrelation matrix with larger $\kappa(R)$, and a vector w_o with more entries with similar amplitude. Only after the VL LMS algorithm is switched

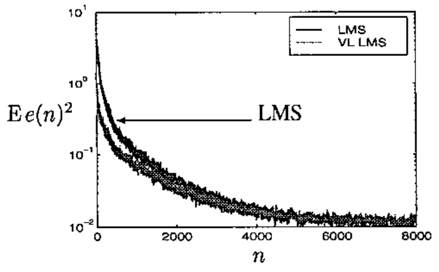

Fig. 8. *MSE* ($\mathrm{E}\,e(n)^2$) *computed for LMS and VL LMS, average of 300 curves for the second equalizer example.*

to standard LMS does the convergence slow down. Better results might be obtained if sub-filter lengths larger than 16 were allowed, or if a smarter merging strategy were used.

4. CONCLUSIONS

In this paper, variable-length LMS filters were described, and some of their properties analysed. We showed that these algorithms have interesting convergence properties, that they are less affected by the eigenvalue dispersion of the input autocorrelation matrix than LMS, but are sensitive to the optimum weight vector, w_o (the Wiener solution), since the convergence is faster when w_o has few dominating entries.

Many modifications of the variable-length algorithm may be devised, which may lead to less sensitivity to w_o: different strategies for merging sub-filters, choosing the overall output error, choosing initial conditions for each stage. One may start, as we did here, with length-one sub-filters, and double the sub-filter lengths at each stage, but it would also be possible to start with larger sub-filters (which would result in a smaller computational cost). It is also possible to use sub-filters with lengths larger than $M/2$ taps, and use more stages before turning to standard LMS.

Another interesing possibility is to choose the switching points between stages dynamically, and even to allow the filter to return to a previous stage in rapidly-varying situations, where the error $e(n)$ may increase suddenly. These possibilities are now being pursued.

REFERENCES

[1] N.J. Bershad. Analysis of the normalized LMS algorithm with Gaussian inputs. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, ASSP-34:793–806, 1986.

[2] H.J. Butterweck. A wave theory of long adaptive filters. *IEEE Trans. on Circuits and Systems–I*, 48(6):739–747, June 2001.

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

[4] E. Eweda. Comparison of RLS, LMS, and sign algorithms for tracking randomly time-varying channels. *IEEE Trans. on Signal Processing*, 42(11):2937–2944, November 1994.

[5] B. Hassibi, A.H. Sayed, and T. Kailath. LMS is H^∞-optimal. In *Proc. Conference on Decision and Control*, vol. 1, pages 74–79, San Antonio, TX, Dec. 1993.

[6] S. Haykin. *Adaptive Filter Theory*. Prentice-Hall, 3rd edition, 1996.

[7] V.H. Nascimento. The normalized LMS algorithm with dependent noise. In *Anais do 19º Simpósio Brasileiro de Telecomunicações*, Fortaleza, Brazil, 2001.

[8] V.H. Nascimento and A.H. Sayed. On the learning mechanism of adaptive filters. *IEEE Trans. on Signal Processing*, 48(6):1609–1625, June 2000.

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

[10] V. Solo and X. Kong. *Adaptive Signal Processing Algorithms*. Prentice-Hall, NJ, 1995.