PARTIAL-UPDATE NLMS ALGORITHMS WITH DATA-SELECTIVE UPDATING

Stefan Werner,[1] *Marcello L. R. de Campos,*[2] *Paulo S. R. Diniz*[2]

[1]Helsinki University of Technology
Signal Processing Laboratory
Stefan.Werner@hut.fi

[2]Universidade Federal do Rio de Janeiro
COPPE/Prog. Engenharia Elétrica
[campos,diniz]@lps.ufrj.br

ABSTRACT

Partial-update adaptive filtering algorithms only update part of the filter coefficients at each time instant, leading to reduced computational complexity as compared with their conventional counterparts. In this paper, the ideas of the partial-update NLMS-type algorithms found in the literature are extended to the framework of set-membership filtering, from which data-selective NLMS type of algorithms with partial update are derived. The new algorithms combine data-selective updating from set-membership filtering with the reduced computational complexity from partial updating. Simulation results verify the good performance of the new algorithms in terms of convergence speed, final misadjustment, and reduced computational complexity.

1. INTRODUCTION

With the choice of algorithms ranging from the simple LMS algorithm to the more complex RLS algorithm, tradeoffs between performance criteria such as, e.g., computational complexity and convergence rate have to be made. In certain applications, the use of the RLS algorithm is prohibitive due to the high computational complexity and in such cases we must resort to simpler algorithms. As an example, consider an acoustic echo cancellation application where the adaptive filter might require thousands of coefficients [1]. For this large number of filter coefficients, it is possible that with the given resources even the implementation of low computational-complexity algorithms, such as the NLMS algorithm, could be impractical. Some means of reducing the computational complexity even further are to apply quantized-error algorithms like, for example, the sign-error and sign-data algorithms [2]. Alternatively, instead of reducing the filter order, or applying quantized-error algorithms, one may choose to update only part of the filter vector at each time instant. Such algorithms, referred to as partial-update algorithms, can reduce the computational complexity per iteration while performing close to their full update counterparts in terms of convergence rate and final MSE. In the literature one can find several variants of the LMS and the NLMS algorithms with partial updates, see, e.g., [1], [3]–[5].

Another efficient approach to reduce the computational complexity is to employ set-membership filtering (SMF) algorithms [6]. Algorithms derived within the framework of SMF employ a deterministic objective function related to a bounded error constraint on the filter output. The resulting adaptation algorithms are data selective, which in turn can reduce considerably the average computational complexity. Data-selective algorithms with low computational complexity per update are the set-membership NLMS (SM-NLMS) [6], the set-membership binormalized data-reusing (SM-BNDRLMS) [7], and the set-membership affine projection (SM-AP) [8] algorithms.

The objective of this paper is to propose a framework which allows the combination of set-membership normalized data-reusing algorithms with partial-update algorithms. The resulting algorithms benefit from the data-selective updating related to the set-membership framework reducing the average computational complexity, and also from the reduced computational complexity obtained with the partial update of the coefficient vector.

2. SET-MEMBERSHIP FILTERING

In set-membership filtering (SMF), the filter $\mathbf{w}$ is designed to achieve a specified bound on the magnitude of the output error. Assuming a sequence of input vectors $\{\mathbf{x}_k\}_{k=1}^{\infty}$, a desired-signal sequence $\{d_k\}_{k=1}^{\infty}$, we can write the sequence of output errors $\{e_k\}_{k=1}^{\infty}$ as,

$$e_k = d_k - \mathbf{w}^{\mathrm{T}}\mathbf{x}_k \tag{1}$$

where $\mathbf{x}_k$ and $\mathbf{w} \in \mathbb{R}^N$, and d_k and $e_k \in \mathbb{R}$. For a properly chosen bound γ on the estimation error, there are several valid estimates of $\mathbf{w}$. Let $\mathcal{H}_k$ denote the set containing all vectors $\mathbf{w}$ for which the associated output error at time instant k is upper bounded in magnitude by γ. In other words,

$$\mathcal{H}_k = \{\mathbf{w} \in \mathbb{R}^N : |d_k - \mathbf{w}^{\mathrm{T}}\mathbf{x}_k| \leq \gamma\} \tag{2}$$

The set $\mathcal{H}_k$ is referred to as the *constraint set* and its boundaries are hyperplanes. Finally, define the *feasibility set* ψ_k to be the intersection of the constraint sets over the time instants $i = 1, \ldots, k$, i.e.,

$$\psi_k = \bigcap_{i=1}^{k} \mathcal{H}_i \tag{3}$$

The idea of set-membership adaptive recursion techniques (SMART) is to adapt in a way as to remain within the feasibility set. The set-membership NLMS (SM-NLMS) algorithm uses the information provided by constraint set $\mathcal{H}_k$ to construct a set of feasible solutions.

3. THE SET-MEMBERSHIP PARTIAL-UPDATE NLMS ALGORITHM

In this section we extend the idea of partial update to the framework of SMF. The goal is to combine the advantages of SMF and partial update to obtain an algorithm with sparse updating and low computational complexity per update. Let the L coefficients updated at time instant k be specified by an index set $\mathcal{I}_L(k) = \{i_0(k), \ , \ldots, \ i_{L-1}(k)\}$ with $\{i_j\}_{j=0}^{L-1}$ taken from the set $\{0, \ \ldots, \ N\}$. Note that $\mathcal{I}_L(k)$ depends on the time instant k. As a consequence, the L coefficients to be updated can change between consecutive time instants. A question that naturally arises is "Which L coefficients should be updated?". The answer can be related to the optimization criterion chosen for the algorithm derivation. Our approach is to seek a coefficient vector update that minimizes the Euclidean distance $\|\mathbf{w}_{k+1} - \mathbf{w}_k\|^2$ subject to the constraint that $\mathbf{w}_{k+1} \in \mathcal{H}_k$ with the additional constraint of updating only L coefficients. This means that if $\mathbf{w}_k \in \mathcal{H}_k$, the minimum distance is zero and no update is required. However, when $\mathbf{w}_k \notin \mathcal{H}_k$, the new update is obtained as the solution to the optimization problem in Equation (4) below. Introduce the diagonal matrix $\mathbf{A}_{\mathcal{I}_{L(k)}}$ having L ones in the positions indicated by $\mathcal{I}_{L(k)}$ and zeros elsewhere. Defining the complementary matrix $\tilde{\mathbf{A}}_{\mathcal{I}_{L(k)}} = \mathbf{I} - \mathbf{A}_{\mathcal{I}_{L(k)}}$ will give $\tilde{\mathbf{A}}_{\mathcal{I}_{L(k)}}\mathbf{w}_{k+1} = \tilde{\mathbf{A}}_{\mathcal{I}_{L(k)}}\mathbf{w}_k$ when only L coefficients are updated. With these notations, the optimization criterion for the partial update can be formulated as

$$\begin{aligned} \mathbf{w}_{k+1} = \min_{\mathbf{w}} \|\mathbf{w} - \mathbf{w}_k\|^2 & \text{ subject to:} \\ d_k - \mathbf{x}_k^T \mathbf{w} &= g_k \\ \tilde{\mathbf{A}}_{\mathcal{I}_L(k)} (\mathbf{w} - \mathbf{w}_k) &= \mathbf{0} \end{aligned} \tag{4}$$

where g_k is a parameter that determines a point within the constraint set $\mathcal{H}_k$, therefore, $|g_k| \leq \gamma$. Herein g_k is chosen such that the updated vector belongs to the closest bounding hyperplane in $\mathcal{H}_k$, i.e., $g_k = \gamma \text{sign}(e_k)$. Applying the method of Lagrange multipliers gives the recursive updating rule

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha_k \frac{e_k \mathbf{A}_{\mathcal{I}_L(k)} \mathbf{x}_k}{\|\mathbf{A}_{\mathcal{I}_L(k)} \mathbf{x}_k\|^2} \tag{5}$$

where α_k is a data dependent step size given by

$$\alpha_k = \begin{cases} 1 - \gamma/|e_k| & \text{if } |e_k| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

We see from (5) that only the coefficients of $\mathbf{w}_k$ indicated by the index set $\mathcal{I}_L(k)$ are updated, whereas the remaining coefficients are not changed. As with the PU-NLMS algorithm [5], we can conclude through substitution of (5) in (4) that the index set $\mathcal{I}_L(k)$ minimizing (4) is the set associated with the L coefficients in the input vector $\mathbf{x}_k$ having the largest norm. The algorithm is similar in form to the PU-NLMS algorithm, but not in philosophy or in derivation.

A graphical view of the SM-PU-NLMS algorithm update is given in Figure 1 for the case of $N + 1 = 3$ filter coefficients and $L = 1$ coefficient in the partial update. In the figure, the component x_3 is the element of maximum magnitude in $\mathbf{x}_k$, and the matrix $\mathbf{A}_{\mathcal{I}_{L_3(k)}}$ which decides the direction of update $\mathbf{w}_{k+1}$ in $\mathbb{R}^3$ is, therefore, in this example given by $\mathbf{A}_{\mathcal{I}_{L_3(k)}} = \text{diag}(0\,0\,1)$. The solution $\mathbf{w}^\perp$ in Figure 1 is the solution of the SM-NLMS algorithm obtained by an orthogonal projection of $\mathbf{w}_k$ onto the closest boundary of $\mathcal{H}_k$. The angle θ shown in Figure 1 denotes the angle between the direction of update between $\mathbf{A}_{\mathcal{I}_{L_3(k)}}\mathbf{x}_k$ and the input vector $\mathbf{x}_k$. The angle θ in $\mathbb{R}^N$ is given by $\cos\theta = \frac{\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|}{\|\mathbf{x}_k\|}$. In order to take the solution of the SM-PU-NLMS algorithm after the update closer to the orthogonal projection than the solution before the update, consider the bound given by the following lemma:

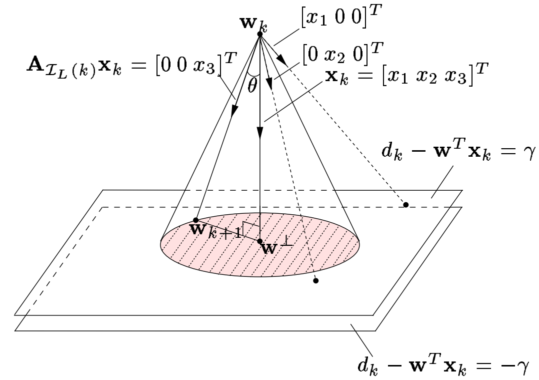

Fig. 1. Geometric illustration of an update in $\mathbb{R}^3$ using $L = 1$ coefficient in the partial update, and with $|x_3| > |x_2| > |x_1|$, the direction of the update is along the vector $[0\ 0\ x_3]^T$ forming an angle θ with the input vector $\mathbf{x}_k$.

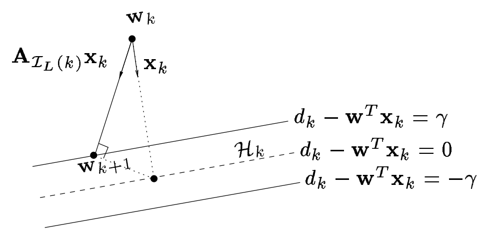

Fig. 2. General projection solution.

Lemma: $\|\mathbf{w}_{k+1} - \mathbf{w}^\perp\| \leq \|\mathbf{w}^\perp - \mathbf{w}_k\|$ for $\frac{\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|}{\|\mathbf{x}_k\|} \geq \frac{1}{\sqrt{2}}$.

Proof: The orthogonal projection $\mathbf{w}^\perp = \mathbf{w}_k + \alpha_k \frac{e_k \mathbf{x}_k}{\|\mathbf{x}_k\|^2}$ [6] where α_k is the data-dependent step size given by Equation (6). Consequently, $\|\mathbf{w}^\perp - \mathbf{w}_k\|^2 = \frac{\alpha_k^2 e_k^2}{\|\mathbf{x}_k\|^2}$. Since $\mathbf{w}_{k+1}$ and $\mathbf{w}^\perp$ lie in the same hyperplane, we have $(\mathbf{w}_{k+1} - \mathbf{w}^\perp) \perp (\mathbf{w}^\perp - \mathbf{w}_k)$. Therefore, $\|\mathbf{w}_{k+1} - \mathbf{w}^\perp\|^2 = \|\mathbf{w}_{k+1} - \mathbf{w}_k\|^2 - \|\mathbf{w}^\perp - \mathbf{w}_k\|^2 = \frac{\alpha_k^2 e_k^2}{\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|^2} - \frac{\alpha_k^2 e_k^2}{\|\mathbf{x}_k\|^2}$. For $\|\mathbf{w}_{k+1} - \mathbf{w}^\perp\|^2 \leq \|\mathbf{w}^\perp - \mathbf{w}_k\|^2$ to hold, $\frac{\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|^2}{\|\mathbf{x}_k\|^2} \geq \frac{1}{2}$ is required.

The lemma tells us that if the instantaneous power in the input vector corresponding to the partial update is larger than half of the total instantaneous power, the SM-PU-NLMS update will be closer to the orthogonal solution than the current solution. For white input signals we can make the approximations $\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|^2 = \sigma_x^2 L$ and $\|\mathbf{x}_k\|^2 = \sigma_x^2(N+1)$ for large $N+1$ and L. Using these approximations a lower bound on the number of coefficients in the partial update is $L > (N+1)/2$.

Unlike the PU-NLMS algorithm, the solution to the SM-PU-NLMS algorithm is required to belong to the constraint set; the introduction of a step size to prevent divergence is outside the framework of SMF. On the other hand, stability problems for the SM-PU-NLMS algorithm may arise when L is small, and as a consequence, θ is increased. In order to address this problem, consider the following update strategy.

Proposition 1: *Increase the number of filter coefficients in the partial update vector until the relation* $\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|^2 \geq \alpha_k \|\mathbf{x}_k\|^2$ *is true.*

Proposition 1 gives a solution where the number of coefficients in the update varies with time. In case of equality we have

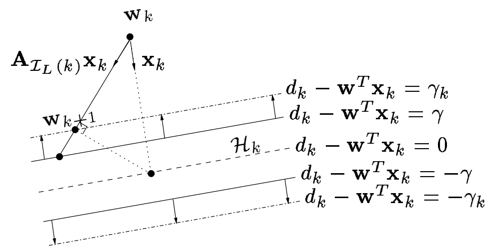

Fig. 3. Projection solution with temporary expansion of the constraint set $\mathcal{H}_k$ using a new threshold γ_k

.

Table 1. SM-PU-NLMS algorithm, with $L \leq L_{max}$.

```
for k = 1:K
    y = w' * x
    e = d - y + n
    P = P + x(k)^2 - x(k-N)^2
    if abs(e) > gamma
        [z, i] = sort(abs(x))
        i = i(N:-1:1)
        l = 0
        p = 0
        alpha = 1 - gamma/abs(e)
        b = alpha*P
        while p < b & l < Lmax
            l = l + 1
            p = p + x(i(l))
        end;
        if p < b
            c = 1 - p/P
            alpha = 1 - c
        end;
        a = z(i(1:l))
        t = a' * a
     w(i(1:l)) = w(i(1:l)) + alpha*e/t*x(i(1:l))
    end;
end;
```

$\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|^2 = \alpha_k\|\mathbf{x}_k\|^2$, and the update can be viewed as the projection of the zero *a posteriori* solution onto $\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k$, as illustrated in Figure 2. No upper bound on L is guaranteed, and the proposed strategy would most likely result in an L close to $N+1$ during the initial adaptation. This is clearly not desirable for the case of partial-update algorithms, where in many cases $L << (N+1)$ is required. Therefore, we consider the following alternative proposition:

Proposition 2: *Increase the number of filter coefficients in the partial update vector until the relation* $\|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|^2 \geq \alpha_k\|\mathbf{x}_k\|^2$ *is true or* $L = L_{max}$. *If* $L = L_{max}$, *increase the threshold* γ *temporarily at the kth iteration to* $\gamma_k = \frac{\|\mathbf{x}_k\|^2 - \|\mathbf{A}_{\mathcal{I}_L(k)}\mathbf{x}_k\|^2}{\|\mathbf{x}_k\|^2}|e_k|$.

As illustrated in Figure 3, Proposition 2 will temporarily expand the constraint set in order to provide a feasible solution if the required number of coefficients to fulfill Proposition 1 exceeds a predefined maximum number of coefficients L_{max} set at the design stage. Tables 1 and 2 show two different versions of the SM-PU-NLMS algorithm. The version in Table 1 implements Proposition 2 and the number of coefficients are allowed to vary freely such that $L \leq L_{max}$, where $L_{max} \leq N+1$ is a predefined value. If $L_{max} = N+1$ the algorithm will be the same as the one in Proposition 1. Table 2 implements a version where L is fixed all the time and is not allowed to vary during the adaptation. The choice between the two versions is application dependent.

Table 2. SM-PU-NLMS algorithm, L fixed during adaptation.

```
for k = 1:K
    y = w' * x
    e = d - y + n
    P = P + x(k)^2 -x(k-N)^2
    if abs(e) > gamma
        [z, i] = sort(abs(x))
        i = i(N:-1:1)
        a = z(i(1:L))
        p = a' * a
        alpha = 1 - gamma/abs(e)
        b = alpha*P
        if p < b
            c = 1 - p/P
            alpha = 1 - c
        end;
        w(i(1:l)) = w(i(1:l)) + alpha*e/p*x(i(1:l))
    end;
end;
```

Table 3. Computational complexity.

ALG.	MULT.	ADD.	DIV.
NLMS	$2N+5$	$2N+4$	1
SM-NLMS	$2N+5$	$2N+5$	2
PU-NLMS [5]	$N+L+4$	$N+L+3$	1
SM-PU-NLMS	$N+L+4$	$N+L+4$	2

4. COMPUTATIONAL COMPLEXITY

The computational complexities per update in terms of the number of additions, multiplications, and divisions for the NLMS, SM-NLMS, PU-NLMS, and SM-PU-NLMS (L fixed) algorithms are shown in Table 3. Although the PU-NLMS and SM-PU-NLMS algorithms have a similar complexity per update, the gain of applying the SM-PU-NLMS algorithm comes through the reduced number of required updates. For time instants where no updates are required, the complexity of the SM-PU-NLMS algorithm is due to filtering, i.e., N additions and multiplications. In the operation counts, the value of $\|\mathbf{x}_{k-1}\|^2$ was assumed known at iteration k such that $\|\mathbf{x}_k\|^2$ can be computed as $\|\mathbf{x}_k\|^2 = \|\mathbf{x}_{k-1}\|^2 + x_k^2 - x_{k-N}^2$, which requires only two multiplications and two additions.

In order to find the L largest-norm elements in $\mathbf{x}_k$, *comparison-sort* algorithms can be used, which require a maximum number comparisons of order $O(N \log N)$, see, e.g., [5]. Both the PU-NLMS and the SM-PU-NLMS algorithms require additional memory to store the pointers to the sorted list. The amount of additional memory can be reduced by partitioning the coefficient and input vectors into blocks and perform block updates as proposed in [4], but at the expense of a decrease in convergence speed.

5. SIMULATIONS

In this section, the two SM-PU-NLMS algorithms are applied to a system identification problem. The order of the plant was $N - 1 = 50$ and colored noise input signal was used with SNR set to 60dB. The colored noise was generated by passing a white noise sequence through an one-pole filter with pole at $z_p = 0.8238$. The bound on the output error was set to $\gamma = \sqrt{5\sigma_n^2}$. Figure 4 shows the learning curves averaged over 500 simulations for the SM-PU-NLMS algorithm using the algorithm shown in Table 2,

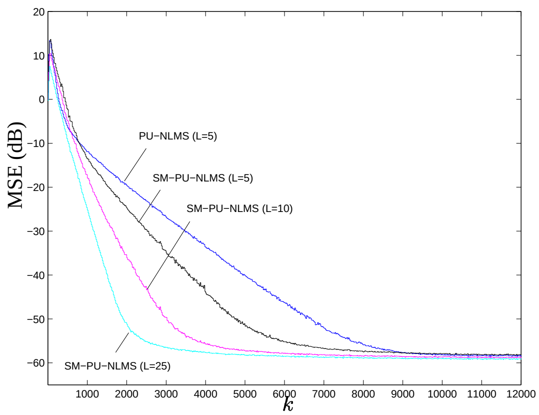

Fig. 4. Learning curves.

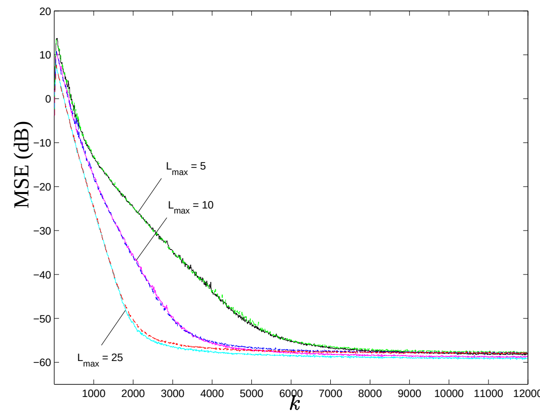

Fig. 5. Learning curves, $L \leq L_{max}$ (dashed) and L fixed (solid).

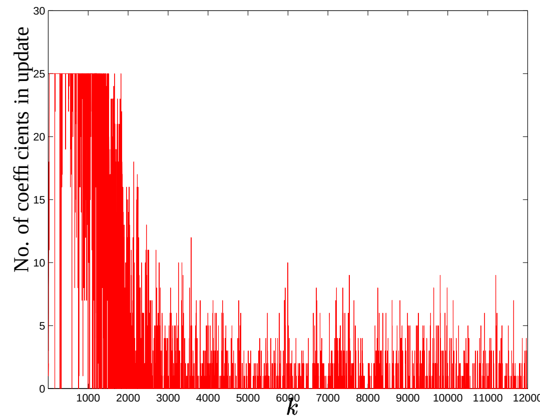

Fig. 6. Number of coefficients updated in the partial update versus time in a single realization for the SM-PU-NLMS algorithm with L variable and $L \leq 25$.

i.e, with fixed L. The learning curve for the PU-NLMS algorithm with $L = 5$ was included as a reference. The step size in the PU-NLMS algorithm was $\mu = 0.3676$ which resulted in the same level of misadjustment as the SM-PU-NLMS algorithm with $L = 5$. In 12000 iterations the number of times an update took place for $L = 5$, $L = 10$ and $L = 25$ were 4950, 3340, and 2420 respectively. As can be seen from Figure 4 the SM-PU-NLMS algorithm converges faster than the PU-NLMS algorithm for the same level of misadjustment.

Figure 5 shows the learning curves for the SM-PU-NLMS algorithm with variable L. The results for the SM-PU-NLMS algorithm obtained previously are included in Figure 5 for reference. As can be seen from the figure, the SM-PU-NLMS algorithm with variable L converges to a slightly higher steady-state value than the SM-PU-NLMS algorithm using fixed L. In 12000 iterations the number of times an update took place for $L_{max} = 5$, $L_{max} = 10$, and $L_{max} = 25$ were 5070 and 3640, and 2840 respectively, which is slightly higher than when L is fixed. However, the number of coefficients in the partial update was also smaller for most of the time instants, which can be observed from Figure 6 where for $L \leq 25$, the number of coefficients in the partial update versus time is shown during one realization. As can be seen from the figure, close to $L_{max} = 25$ coefficients are used during the initial iterations whereas later on this value decreases considerably. The same trend was observed for the case of $L_{max} = 5$ and $L_{max} = 10$ but the results were not included due to limited space.

6. CONCLUSIONS

In this paper, novel data-selective normalized adaptation algorithms with partial updating were derived based on the concept of set-membership filtering. The new algorithms benefit from the reduced average computational complexity from the set-membership filtering framework and the reduced computational complexity resulting from partial updating. Simulations were presented for a system identification application. It was verified that not only the data-selective adaptation algorithms with partial updating can further reduce the computational complexity as compared with the partial-update NLMS algorithm, but they also retain fast convergence without increasing the excess of MSE.

7. REFERENCES

[1] S. C. Douglas, "Adaptive filters employing partial updates," *IEEE Trans. Circuits Syst. II*, vol. 44, pp. 209–216, Mar. 1997.

[2] P. S. R. Diniz, *Adaptive Filtering — Algorithms and Practical Implementation*, Boston, MA: Kluwer Academic Press, 1997.

[3] T. Aboulnasr and K. Mayyas, "Complexity reduction of the NLMS algorithm via selective coefficient update," *IEEE Trans. Signal Processing*, vol. 47, pp. 1421–1427, May 1999.

[4] T. Schertler, "Selective block update of NLMS type algorithms," *ICASSP'98*, May 1998, vol. 3, pp. 1717–1720.

[5] K. Doğançay and O. Tanrıkulu, "Selective-partial-update NLMS and affine projection algorithms for acoustic echo cancellation," *ICASSP'2000*, June 2000, vol. 1, pp. 448–451.

[6] S. Gollamudi, S. Nagaraj, S. Kapoor, and Y.-F. Huang, "Set-membership filtering and a set-membership normalized LMS algorithm with an adaptive step size," *IEEE Signal Processing Lett.*, vol. 5, pp. 111-114, May 1998.

[7] P. S. R. Diniz and S. Werner, "Set-membership binormalized LMS data-reusing algorithms," in *Proc. IFAC SYSID'2000*, Santa Barbara, USA, June 2000, vol. 3, pp. 869–874.

[8] S. Werner and P. S. R. Diniz, "Set-membership affine projection algorithm," *IEEE Signal Processing Lett.*, vol. 8, pp. 231–235, Aug. 2001.