Hindawi
Shock and Vibration
Volume 2017, Article ID 3864951, 11 pages
https://doi.org/10.1155/2017/3864951

Research Article

Multichannel Filtered-X Error Coded Affine Projection-Like Algorithm with Evolving Order

J. G. Avalos, A. Rodriguez, H. M. Martinez, J. C. Sanchez, and H. M. Perez

Instituto Politecnico Nacional, ESIME Culhuacan, Av. Santa Ana No. 1000, Coyoacan, 04260 Ciudad de Mexico, Mexico

Correspondence should be addressed to J. G. Avalos; javaloso@ipn.mx

Received 4 November 2016; Revised 15 March 2017; Accepted 3 April 2017; Published 8 June 2017

Academic Editor: M. I. Herreros

Affine projection (AP) algorithms are commonly used to implement active noise control (ANC) systems because they provide fast convergence. However, their high computational complexity can restrict their use in certain practical applications. The Error Coded Affine Projection-Like (ECAP-L) algorithm has been proposed to reduce the computational burden while maintaining the speed of AP, but no version of this algorithm has been derived for active noise control, for which the adaptive structures are very different from those of other configurations. In this paper, we introduce a version of the ECAP-L for single-channel and multichannel ANC systems. The proposed algorithm is implemented using the conventional filtered-x scheme, which incurs a lower computational cost than the modified filtered-x structure, especially for multichannel systems. Furthermore, we present an evolutionary method that dynamically decreases the projection order in order to reduce the dimensions of the matrix used in the algorithm's computations. Experimental results demonstrate that the proposed algorithm yields a convergence speed and a final residual error similar to those of AP algorithms. Moreover, it achieves meaningful computational savings, leading to simpler hardware implementation of real-time ANC applications.

1. Introduction

Active noise control (ANC) [1, 2] is a field that still presents challenges and open problems for researchers [3, 4], especially with regard to the limitations of the hardware used in its implementation. ANC systems are commonly developed using FIR adaptive filters; therefore, for real-time applications, it is very important to choose a suitable adaptive algorithm. One of the most widely used algorithms for this purpose is the filtered-x least mean square (FXLMS) algorithm [2]; however, its convergence speed is slow, which makes it unsuitable for some applications, especially for multichannel systems. The authors of [5] proposed the modified filtered-x LMS (MFXLMS) algorithm, which provides a slightly better convergence speed but at a greatly increased computational cost. Therefore, implementation of the MFXLMS algorithm for multichannel systems demands considerable computational resources, making certain applications unfeasible. Variants of the FXLMS algorithm that improve the convergence speed using a variable step-size are proposed in [6, 7]; nevertheless, they also require more computations. Other approaches enhance the convergence using different strategies such as convex combination of adaptive filters [8], adaptive IIR filters [9], training mechanism based on recursive least square algorithm [10], and the use of two adaptive filters [11]; however, those strategies involve a higher computational burden than the classic adaptive filters.

The affine projection algorithm (APA) [12] is widely used in ANC systems because it offers higher convergence speed than LMS algorithms and presents good stability and robustness. The APA updates the weight vector using the most recent L input vectors instead of using the current input vector. The parameter L is usually called the projection order; if it is increased, the convergence speed also increases, but the computational cost and final residual error (misadjustment) increase as well. Another critical concern is the several matrix inversions required during the process, which complicate the hardware implementation.

Several authors have published various strategies with the intent of improving the performance or reducing the computational burden of the APA in multichannel ANC systems. In [13], a fast affine projection algorithm called the

MFXFAP-RLS algorithm is presented, which uses a modified structure and a sliding window; however, this method does not guarantee stability, and the use of the "modified filtered-x structure" increases the computational cost of implementation. Moreover, for small projection orders, its complexity may be similar to that of the APA, which is a great disadvantage since ANC applications typically function well for projection orders of lower than 5 [14].

Other proposed approaches use different strategies to decrease the cost of matrix inversion. In [15], a scheme based on the Gauss-Seidel method is used; nevertheless, it remains necessary to compute at least one matrix inversion, which may result in instability for large matrices. In [16], dichotomous coordinate descent (DCD) iterations are applied for coefficient updating; however, most variants of this approach are based on the fast affine projection (FAP) algorithm, and as mentioned above, for lower projection orders, the APA offers simpler implementation.

In [17], efficient versions of the FAP algorithm that incur low computational loads are developed. However, the recursive approach used for matrix inversion does not present any improvement over the original version for low projection orders and adaptive filters with few coefficients, and therefore, it is a good option only for long-length filters.

Some researchers have also proposed variable-step-size AP algorithms. Such modifications result in better final errors in the steady state, but the computational cost remains the same; furthermore, most of these proposals are intended for acoustic echo canceler applications [17–21].

Considering the problems encountered when using high projection orders, algorithms have been proposed that dynamically adjust the number of input vectors used to update the coefficients [22–25]. Algorithms of this type have a reduced computational burden as the process advances and achieve lower misadjustment. The authors of [26] presented a multichannel ANC system using variants of AP algorithms with a variable step size and a projection order that evolves. In [27], the error signal of the AP algorithm is encoded and the projection order is also calculated dynamically, the resulting algorithm is applied to a single-channel ANC system; the reported results in both works demonstrate the efficiency of those strategies; nevertheless, it is still necessary to perform matrix inversion to update the algorithm.

To overcome some of the problems mentioned above, [28] introduced the Error Coded Affine Projection-Like (ECAP-L) algorithm, which is an algorithm that does not require either direct or indirect inversion of the input signal matrix; moreover, the filter coefficients are updated only when the output estimation error is higher than a predetermined threshold, thereby considerably reducing the amount of computation required per iteration.

In this work, we present an efficient multichannel version of the ECAP-L algorithm for active noise control applications. To compensate for the negative effects of the secondary path in an ANC system, we use the filter-x structure; moreover, based on the schemes of evolving order algorithms, we develop a strategy to adjust the projection order in each iteration. The result is the new FXECAP-L algorithm with evolving order.

In Section 2, we describe the original ECAP-L algorithm and analyze its stability. The development of the FXECAP-L algorithm with evolving order for ANC is presented in Section 3. In Section 4, the computational complexity of the proposed algorithm is evaluated. Simulation results for single-channel and multichannel ANC systems are discussed in Section 5. Finally, the conclusions are outlined in Section 6.

2. Error Coded Affine Projection-Like Algorithm

The ECAP-L algorithm [28] is a good option for hardware implementation because its convergence speed is similar to that of the APA, it does not require matrix inversion operations, and the coefficients are not updated in each iteration. The filter coefficient update equation for the algorithm is given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n)\,\mathbf{X}(n)\,C[\mathbf{e}(n)], \tag{1}$$

where $\mathbf{w}(n)$ is the weight vector at moment n;

$$\mathbf{X}(n) = \begin{bmatrix}\mathbf{x}(n) & \mathbf{x}(n-1) & \cdots & \mathbf{x}(n-L+1)\end{bmatrix} \tag{2}$$

is the input signal matrix consisting of the recent input vectors $\mathbf{x}(n)$, where L is the projection order; and the error vector $\mathbf{e}(n)$ is expressed as

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{X}^t(n)\,\mathbf{w}(n-1), \tag{3}$$

where $\mathbf{d}(n) = \begin{bmatrix}d(n) & d(n-1) & \cdots & d(n-L+1)\end{bmatrix}^t$ is the input vector that contains the desired signal and the superscript t denotes transposition. Here, the computational load is reduced through the error encoding, for which it is necessary to quantize the signal and then assign a digital code to each quantized sample. The error is encoded as follows:

$$C[\mathbf{e}(n)] = \text{round}\left(\mathbf{e}(n)\,\frac{1}{\text{Res}}\right), \tag{4}$$

where round represents the quantization process, in which the obtained values are rounded to the closest integer, and Res is the encoder resolution and is determined by dividing the maximum probable error ($e_{\max}$) by the number of quantization levels, as shown in

$$\text{Res} = \frac{e_{\max}}{2^b - 1}. \tag{5}$$

Here, $e_{\max}$ is considered to be no larger than 90% of the maximum signal amplitude, and b denotes the number of bits of resolution used. Using a larger number of bits provides more information on the error magnitude, which allows the algorithm to converge more rapidly, but if this number is too high, the amount of information to be processed may be too great, and the algorithm may diverge.

The step size $\mu(n)$ is given by

$$\mu(n) = \frac{\|\mathbf{X}(n)\,\mathbf{e}(n)\|^2}{\|\mathbf{X}^t(n)\,\mathbf{X}(n)\,\mathbf{e}(n)\|^2}\text{Res}. \tag{6}$$

As seen above, to calculate the step size, it is necessary to use the resolution; this is because of the effects of the quantization process. To analyze this issue, we substitute expression (4) into (1) to obtain

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n)\,\mathbf{X}(n)\,\frac{Q[\mathbf{e}(n)]}{\text{Res}}, \tag{7}$$

where $Q[\mathbf{e}(n)]$ represents the quantized error. Now, by substituting (5) into (7), we can write the following:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{X}(n)\,\mu(n)\left(\frac{2^b-1}{e_{\max}}\right)Q[\mathbf{e}(n)]. \tag{8}$$

From (8), it can be observed that the error magnitude $\mathbf{e}(n)$ and the step size are increased; therefore, to avoid a very large value of $\mu(n)$, the step size must be regularized directly with respect to the resolution.

The most important feature of the ECAP-L algorithm is the reduction of the computational burden, which is achieved through the use of a rule for updating the coefficients. This rule is based on the fact that an adaptive filter does not need to be updated when the error signal is small. As the adaptation process advances, the error magnitude diminishes, and if the encoded error is "0" or "1," the filter coefficient is not updated. Thus, the update threshold can be defined as

$$\mathbf{w}(n+1) = \begin{cases} \mathbf{w}(n) & \text{if } C[\mathbf{e}(n)] = 0 \text{ or } 1 \\ \mathbf{w}(n) + \mu(n)\,\mathbf{X}(n)\,C[\mathbf{e}(n)] & \text{if } C[\mathbf{e}(n)] \neq 0 \text{ or } 1. \end{cases} \tag{9}$$

3. Multichannel Filtered-X Error Coded Affine Projection-Like Algorithm for Active Noise Control

This section describes the modifications to the ECAP-L algorithm that are necessary to apply this algorithm for active noise control applications. The first issue to consider is the effect caused by the presence of an unavoidable system response at the adaptive filter output (secondary path), which will affect the algorithm's performance. The filtered-x scheme is commonly used to overcome this problem because of its good convergence properties; moreover, it avoids the additional filtering process used in the modified filtered-x scheme, and therefore, its computational cost is lower.

Figure 1 shows the block diagram of an ANC system using the conventional filtered-x scheme, the main feature of which is the placement of a system with a response identical to that of the secondary path to filter the reference signal and update the algorithm. In this way, the misalignment between the error signal and the reference signal is compensated.

Incorporating the filtered-x scheme into the ECAP-L algorithm yields the Filtered-X ECAP-L (FXECAP-L) algorithm. To describe the FXECAP-L algorithm, the notation in Notations is used.

Then, the adaptive filter output is obtained as follows:

$$y_j(n) = \sum_{i=1}^{I} y_{ji}(n), \quad j = 1, 2, \ldots, J \tag{10}$$

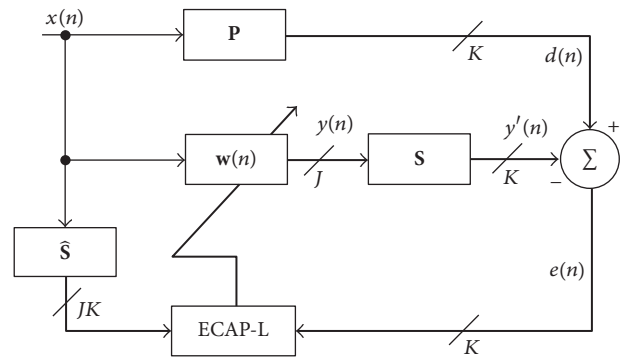

Figure 1: Block diagram of the Filtered-X ECAP-L algorithm for an ANC system [2].

with

$$y_{ji}(n) = \mathbf{w}_{ji}^t(n)\,\mathbf{x}_i(n), \tag{11}$$

where $I \times J$ adaptive filter coefficients are defined as

$$\mathbf{w}_{ji}(n) = \left[w_{j,i,1}(n) \;\; w_{j,i,2}(n) \;\; \cdots \;\; w_{j,i,N}\right]^t, \quad j = 1, 2, \ldots, J, \;\; i = 1, 2, \ldots, I, \tag{12}$$

and the common reference signal vector $\mathbf{x}_i(n)$ is defined as

$$\mathbf{x}_i(n) = \left[x_i(n) \;\; x_i(n-1) \;\; \cdots \;\; x_i(n-N+1)\right]^t, \quad i = 1, 2, \ldots, I. \tag{13}$$

In ANC systems that use the AP algorithm based on filtered-x structure the desired signal $\mathbf{d}(n)$ is unavailable in practice and only the error signal $e(n)$ can be accessed. Therefore, it is necessary to use an approximation to estimate the error signal vector $\mathbf{e}(n)$. The estimation can be done using past samples of the error signals $e_k(n)$ [17]. Thus, the error vectors can be obtained as follows:

$$\mathbf{e}_k(n) \approx \left[e_k(n) \;\; e_k(n-1) \;\; \cdots \;\; e_k(n-L+1)\right]^t, \quad k = 1, 2, \ldots, K. \tag{14}$$

The prefiltering that is necessary to compensate for the effects caused by $\mathbf{s}_{kj}(n)$ can be calculated using

$$x'_{ijk}(n) = \hat{\mathbf{s}}_{kj}^t(n)\,\mathbf{v}_i(n), \tag{15}$$

where

$$\mathbf{v}_i(n) = \left[x_i(n) \;\; x_i(n-1) \;\; \cdots \;\; x_i(n-M+1)\right]^t, \quad i = 1, 2, \ldots, I. \tag{16}$$

Commonly, $\hat{\mathbf{s}}_{kj}(n)$ is a fixed FIR filter estimate of $\mathbf{s}_{kj}(n)$ and generates the reference signals used to calculate the coefficient update equation for the multichannel FXECAP-L algorithm, which is described by

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \sum_{k=1}^{K} \mu_{ijk}\mathbf{X}_{ijk}(n)\,\mathbf{C}\left[\mathbf{e}_k(n)\right], \quad j = 1, 2, \ldots, J, \;\; i = 1, 2, \ldots, I, \tag{17}$$

where $\mathbf{X}_{ijk}(n) = \left[\mathbf{x}'_{ijk}(n) \;\; \mathbf{x}'_{ijk}(n-1) \;\; \cdots \;\; \mathbf{x}'_{ijk}(n-L+1)\right]$ is a matrix whose columns are the current and previous L vectors of the filtered-x signals $\mathbf{x}'_{ijk}(n)$ given by

$$\mathbf{x}'_{ijk}(n) = \left[x'_{ijk}(n) \;\; x'_{ijk}(n-1) \;\; \cdots \;\; x'_{ijk}(n-N+1)\right]^t. \tag{18}$$

The step size μ_{ijk} is computed as

$$\mu_{ijk} = \frac{\left\|\mathbf{X}_{ijk}(n)\,\mathbf{e}_k(n)\right\|^2}{\left\|\mathbf{X}^t_{ijk}(n)\,\mathbf{X}_{ijk}(n)\,\mathbf{e}_k(n)\right\|^2}\text{Res}\cdot\text{sf}, \tag{19}$$

where sf is a scaling factor in the interval $[0, 1]$ that is introduced to facilitate convergence. Specifically, in the case of $\widehat{\mathbf{s}}_{kj}(n) = \mathbf{s}_{kj}(n)$, we could use a value of 1 (i.e., no scaling factor would be required); however, for a larger mismatch between $\widehat{\mathbf{s}}_{kj}(n)$ and $\mathbf{s}_{kj}(n)$, a smaller value of sf would be required.

3.1. Evolutionary Order. The performance of algorithms based on affine projection subspaces is related to the number of previous L input vectors that are used to update the taps of the adaptive filter. With a larger number of input vectors, the convergence speed is increased, but the misadjustment is larger, and the computational cost rises. For this reason, several authors have proposed various approaches for modifying the projection order during the adaptation process [22–25], with the primary purpose of reducing the computational burden.

In this work, we present a strategy based on the evolutionary method proposed in [23], in which the number of input vectors is adjusted dynamically according to a threshold established based on the instantaneous residual error power at each error sensor. In this way, the projection order of each adaptive filter in the multichannel ANC system is calculated as follows:

$$L_{ij}(n) = \begin{cases} \min\left\{L_{ij}(n-1)+1, L_{\max}\right\} & \eta_{ij}(n) < \mathbf{e}_k^2(n) \\ L_{ij}(n-1) & \theta_{ij}(n) < \mathbf{e}_k^2(n) \le \eta_{ij}(n) \\ \max\left\{L_{ij}(n-1)-1, 1\right\} & \mathbf{e}_k^2(n) \le \theta_{ij}(n), \end{cases} \tag{20}$$

where L_{ij} is the projection order for each of the IJ adaptive filters, $L_{\max}$ is the maximum projection order, $\theta_{ij}(n)$ is the lower threshold, and $\eta_{ij}(n)$ is the upper threshold.

The thresholds are determined using the steady-state MSE of the Affine Projection-Like I algorithm [29]; when the output error is smaller than the MSE, the projection order is reduced by one, and when it is larger, the projection order is increased. The derived thresholds can be expressed as

$$\theta_{ij}(n) = \frac{2L_{ij}}{2L_{ij}-1}\sigma_v^2, \qquad \eta_{ij}(n) = \frac{2L_{ij}+1}{2L_{ij}}\sigma_v^2, \tag{21}$$

where σ_v^2 is the variance of a noise signal that is uncorrelated with the disturbance signal. When the evolutionary method is applied to the algorithm, the use of high projection orders is not required throughout the entire process; therefore, the average complexity of the algorithm is reduced.

4. Computational Complexity

Table 1 lists the computational complexity of the FXECAP-L algorithm expressed as the number of multiplications and the number of additions per iteration. The total number of multiplications needed in each iteration of the FXECAP-L algorithm presented in Section 3 is $[((3N+1)L + 2N + 2 + M)IJK + (N)IJ + L(K)]$; the FXAP algorithm requires $[(NL + N + M + NL^2 + L^2 + L^3)IJK + (N)IJ]$ multiplications in each iteration; and finally, the FXAPL-I algorithm requires $[((2N+1)L + 2N + M + 1)IJK + (N)IJ]$ multiplications per iteration. The total number of additions needed in each iteration of the FXECAP-L algorithm is $[(3NL + M - 2)IJK + (N)IJ + J(I-1)]$; the FXAP algorithm requires $[((L + NL - N)N + L^2(L+1) + M)IJK + (N)IJ + J(I-1)]$ additions in each iteration; and finally, the FXAPL-I algorithm requires $[((2L+1)N + M - 2)IJK + (N)IJ + J(I-1)]$ additions per iteration.

A few multiplications are added to the computational complexity of the FXECAP-L algorithm because it uses the thresholds described in (21). The proposed algorithm requires $3IJ$ additional multiplications and $2IJ$ additions in each iteration to determine the projection order. However, once the steady state is reached, the projection order will remain constant, and thus, the computational burden will be greatly reduced. Table 2 compares the memory requirements of the FXAP, FXAPL-I, and the FXECAP-L algorithms.

5. Results

To validate the proposed FXECAP-L algorithm with evolving order, a single-channel ANC system and a 1:2:2 multichannel ANC system were simulated using MATLAB™, and the results were compared with those of the Filtered-X Affine Projection (FXAP) algorithm and the Filtered-X Affine Projection-Like (FXAPL-I) algorithm. Notably, in order to use the APL-I algorithm in an ANC system, it is necessary to use the scaling factor introduced in (19) because this algorithm presents the same mismatch as does the FXECAP-L algorithm.

Several experiments were performed using white Gaussian noise with unit variance as the reference noise signal $x(n)$. The acoustic transfer functions for the primary and secondary paths were obtained from [2] and were measured in a duct. The paths were modeled as FIR filters with tap-weight lengths of 256 and 128, respectively. The secondary path was estimated offline, and the length of the adaptive filters was set to $N = 100$. The projection orders used were $L = 5$ and $L = 10$. The algorithms were run 50 times, each with 50,000 iterations.

Table 1: Description of the FXECAP-L algorithm, the number of multiplications, and the number of additions required (I reference sensors, J actuators, and K error sensors).

Computational step	Number of multiplications	Number of additions
$y_j(n) = \sum_{i=1}^{I} \mathbf{w}_{ji}^t(n)\mathbf{x}_i(n)$	NIJ	$NIJ + J(I-1)$
$x'_{ijk}(n) = \hat{\mathbf{s}}_{kj}^t(n)\mathbf{v}_i(n)$	$MIJK$	$MIJK$
$\mathbf{C}[\mathbf{e}_k(n)]$	LK	No additions required
$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \sum_{k=1}^{K} \mu_{ijk}\mathbf{X}_{ijk}(n)\mathbf{C}[\mathbf{e}_k(n)]$	$((3N+1)L + 2N + 2)IJK$	$(3NL-2)IJK$

Table 2: Memory requirements for the FXAP, FXAPL-I, and FXECAP-L algorithms.

Algorithm	Memory
FXAP	$(2L^2 + NL)IJK + (2N+1)IJ + MJK + MI + LK + J + 2$
FXAPL-I	$(NL + N + L^2 + L + 2)IJK + (2N+1)IJ + MJK + MI + LK + J + 2$
FXECAP-L with evolving order	$(NL + 2N + L^2 + L + 2)IJK + (2N+1)IJ + MJK + MI + 2LK + J + 2$

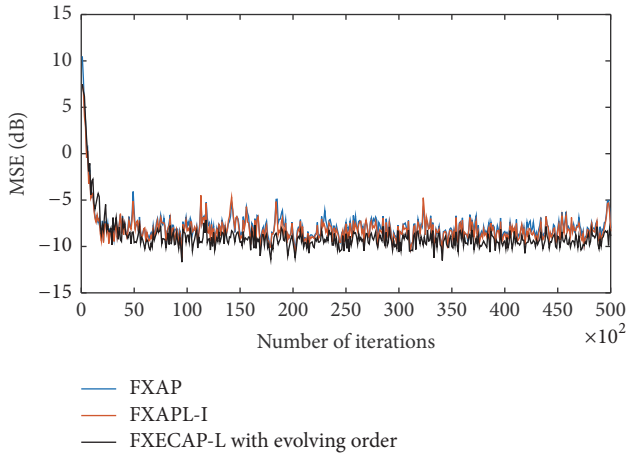

Figure 2: MSE values of the tested AP algorithms for a projection order of $L = 5$ when $b = 2$ bits are used to encode the error.

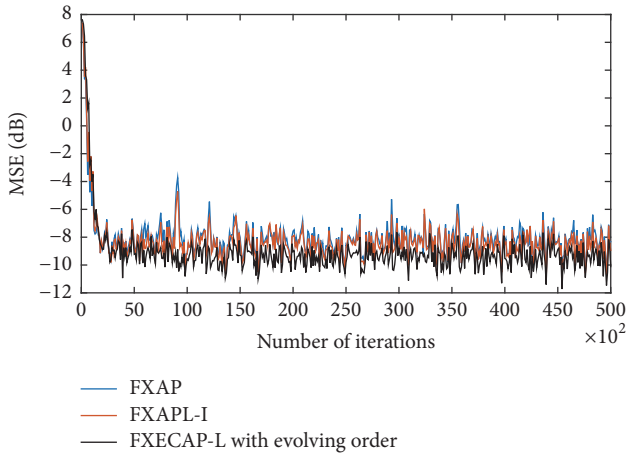

Figure 3: MSE values of the tested AP algorithms for a projection order of $L = 5$ when $b = 8$ bits are used to encode the error.

5.1. Single-Channel ANC System. The first experiment considered a single-channel ANC system. To evaluate the efficiency of the algorithms, the Mean Square Error (MSE) at the error microphone was measured. Figures 2 and 3 show the performance of the FXAP algorithm, the FXAPL-I algorithm, and the proposed algorithm for a projection order of $L = 5$, where the numbers of bits used to encode the error were $b = 2$ and $b = 8$, respectively. The FXAP step size and the scaling factors for the FXAPL-I and FXECAP-L were adjusted by trial and error; the values producing the fastest convergence speed were selected. Thus, for the FXAP algorithm, $\mu = 0.08$, for the FXAPL-I algorithm, sf $= 0.08$, and for the FXECAP-L algorithm when $b = 2$ and $b = 8$, sf $= 0.1$. From Figure 2, it can be seen that the proposed algorithm showed slightly slower convergence than the FXAP and FXAPL-I algorithms; however, when the error was encoded using 8 bits, the convergence was nearly identical. Furthermore, the proposed algorithm does not require calculations using high-order matrices because it adapts to reduce the number of input vectors; this can be observed in Figure 4, which shows that the proposed algorithm needs to work with high-order matrices only in the first 1500 iterations, whereas for the remainder of the process,

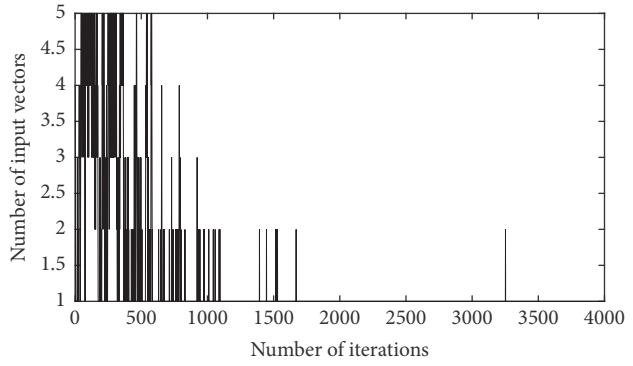

Figure 4: Zoomed view of the number of input vectors used in the proposed FXECAP-L algorithm when $L = 5$ and $b = 2$.

the algorithm works with only one input vector. For $b = 8$, the projection order was reduced to one even before 1500 iterations.

Figure 5 shows the performances of the tested algorithms for a projection order of $L = 10$ when the error is encoded using $b = 2$ bits. Since the projection order is modified, the

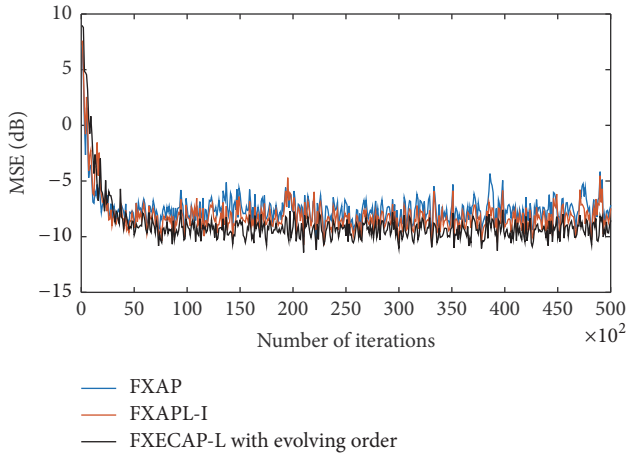

Figure 5: MSE values of the tested AP algorithms for a projection order of $L = 10$ when $b = 2$ bits are used to encode the error.

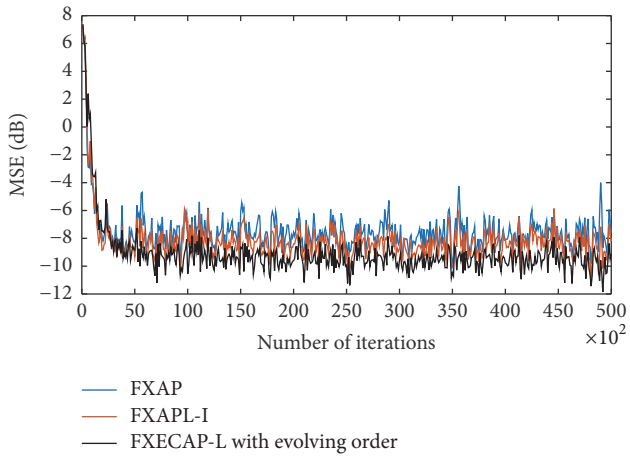

Figure 6: MSE values of the tested AP algorithms for a projection order of $L = 10$ when $b = 8$ bits are used to encode the error.

step size and the scaling factor must be adjusted, the values were chosen by trial and error and were found to be as follows: for the FXAP algorithm, $\mu = 0.07$, for the FXAPL-I algorithm, sf = 0.07, and for the FXECAP-L algorithm when $b = 2$ and $b = 8$, sf = 0.1. As expected, when the projection order increases, the convergence speed also increases, but the proposed algorithm remains slightly slower than the FXAP and FXAPL-I algorithms. For the case of $b = 8$, the convergence speed of the proposed algorithm is very similar to that of the other algorithms; see Figure 6. For all the experiments presented the FXECAP-L algorithm achieved a reduction in the MSE of about 1 and 2 dB with respect to the FXAP-L and FXAP algorithms.

Figure 7 presents the number of input vectors used for $L = 10$ and $b = 2$. As seen from the results, the projection order was reduced to one after approximately 2500 iterations. When the error was encoded using $b = 8$ bits, the behavior was very similar.

Another important advantage that the proposed algorithm possesses over the FXAP and FXAPL-I algorithms is

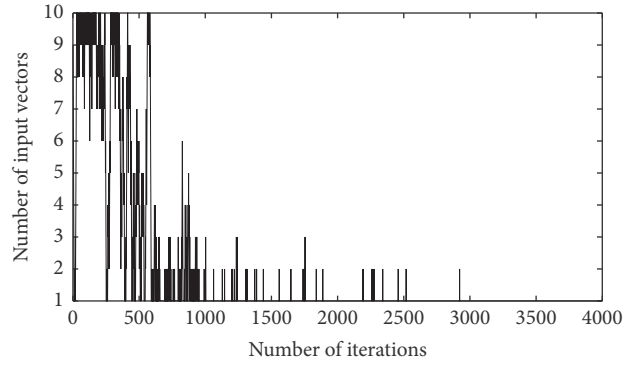

Figure 7: Zoomed view of the number of input vectors used in the proposed FXECAP-L algorithm when $L = 10$ and $b = 2$.

the number of times that the adaptive algorithm updates the filter coefficients. Table 3 shows that the proposed algorithm was updated only 21,992 and 21,931 times for $L = 5$ and $L = 10$, respectively, when the number of bits used to encode the error was $b = 2$. When the error was encoded using a higher number of bits, the resolution resulted in a larger step size; therefore, the convergence speed increased, but the error signal also increased. For this reason, the threshold established for updating the coefficients was not reached, and the algorithm was updated in almost every iteration.

In Table 4, the number of multiplications per iteration required by the FXECAP-L algorithm is compared with the number of multiplications for the FXAP and FXAPL-I algorithms. Table 4 shows the results for both experiments, $L = 5$ and $L = 10$.

The results presented in Table 4 show the number of multiplications for a constant projection order; for that reason the FXECAP-L algorithm presents a higher number than the FXAP-L algorithm. As the process advances the FXECAP-L algorithm changes its projection order, also the algorithm is not updated in each iteration; in this way the overall complexity is greatly reduced.

To test the tracking behavior of the proposed algorithm, the primary path coefficients were multiplied by −1 at iteration 25000. Furthermore, at the microphone sensor, white Gaussian noise was added until a 30 dB signal to noise ratio (SNR) was achieved. The FXAP step size and the scaling factors for the FXAPL-I and FXECAP-L were adjusted by trial and error; the values producing the fastest convergence speed were selected. Thus, for the FXAP algorithm, $\mu = 0.0025$, for the FXAPL-I algorithm, sf = 0.03, and for the FXECAP-L algorithm when $b = 2$, sf = 0.08. Figure 8 shows the MSE for a projection order of $L = 5$, and the number of bits used to encode the error was $b = 2$. As it can be seen, the convergence rate and steady-state estimation error are not degraded. It is worth mentioning that in this case the number of updates was only 9094 (18.18%), since the error signal was smaller and so the threshold established for updating the coefficients was reached sooner. Figure 9 shows the number of input vectors obtained from the tracking experiment, from which it can be observed that the proposed algorithm is able to adjust the projection order.

Table 3: Numbers of updates and update percentages for the single-channel ANC system.

Algorithm	Updates (percentage)
AP	50,000 (100%)
APL-I	50,000 (100%)
ECAP-L with evolving order, $b = 2$ and $L = 5$	21,992 (43.984%)
ECAP-L with evolving order, $b = 2$ and $L = 10$	21,931 (43.862%)
ECAP-L with evolving order, $b = 8$ and $L = 5$	49,583 (99.166%)
ECAP-L with evolving order, $b = 8$ and $L = 10$	49,571 (99.142%)

Table 4: Comparison of the computational burden of the FXECAP-L algorithm with those of the FXAP and FXAPL-I algorithms for ANC systems with $N = 100$ and $M = 128$.

Algorithm	$I = 1, J = 1, K = 1, L = 5$	$I = 1, J = 1, K = 1, L = 10$
FXAP	3478	12428
FXAPL-I	1434	2439
FXECAP-L	1943	3453

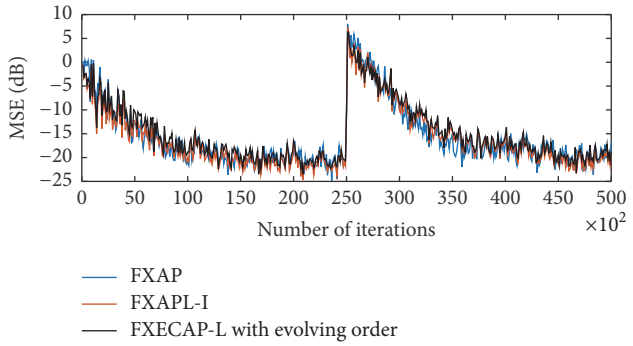

Figure 8: MSE values of the tested AP algorithms in a variant single-channel system. The projection order used is $L = 5$ and $b = 2$ bits are used to encode the error.

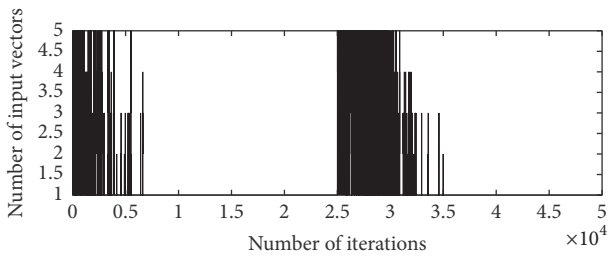

Figure 9: Zoomed view of the number of input vectors used in the proposed FXECAP-L algorithm when $L = 5$ and $b = 2$ in a variant single-channel system.

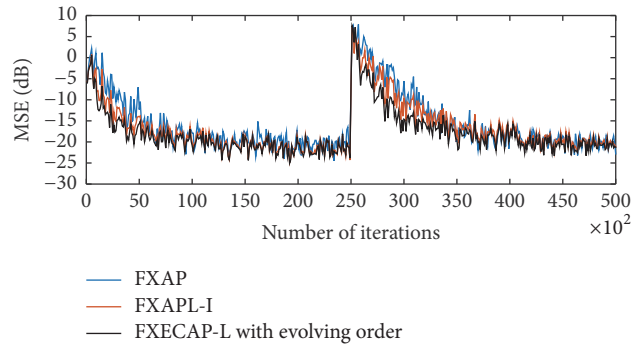

Figure 10: MSE values of the tested AP algorithms in a variant single-channel system. The projection order used is $L = 5$ and $b = 8$ bits are used to encode the error.

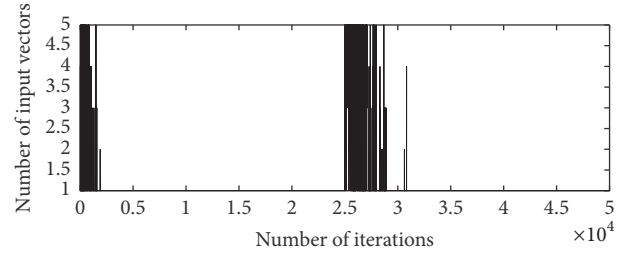

Figure 11: Zoomed view of the number of input vectors used in the proposed FXECAP-L algorithm when $L = 5$ and $b = 8$ in a variant single-channel system.

The tracking experiment was also done for the case $b = 8$. The scaling factor used for the FXECAP-L algorithm was $sf = 0.08$. Figure 10 shows the MSE obtained, as it can be seen, the algorithm also maintained its tracking performance and as expected the convergence speed was increased. The number of updates of the proposed algorithm was 48784 (97.56%); in this case the error signal is increased and the threshold established for updating the coefficients is not reached. Figure 11 shows the number of input vectors obtained from the tracking experiment, from which it can be observed that the proposed algorithm is able to adjust the projection order faster than the case $b = 2$, since the convergence speed is increased.

5.2. Multichannel ANC System. The second experiment considered a multichannel (1:2:2) ANC system. To evaluate the performance, the Mean Square Error (MSE) was measured at each error sensor. The results obtained at error sensor 1 are

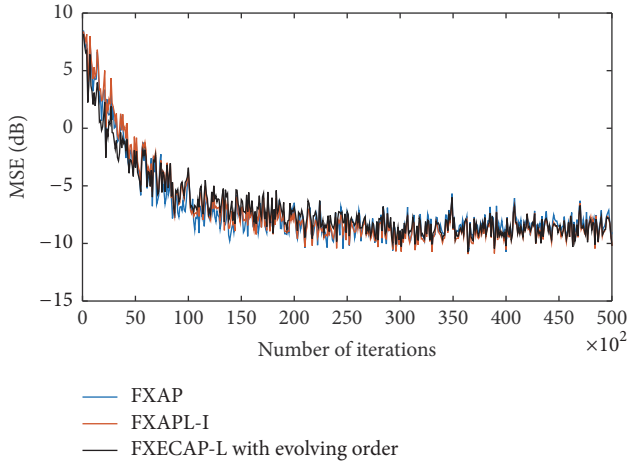

Figure 12: MSE values of the tested AP algorithms as measured at error sensor 1 for a projection order of $L = 5$ when $b = 2$ bits are used to encode the error.

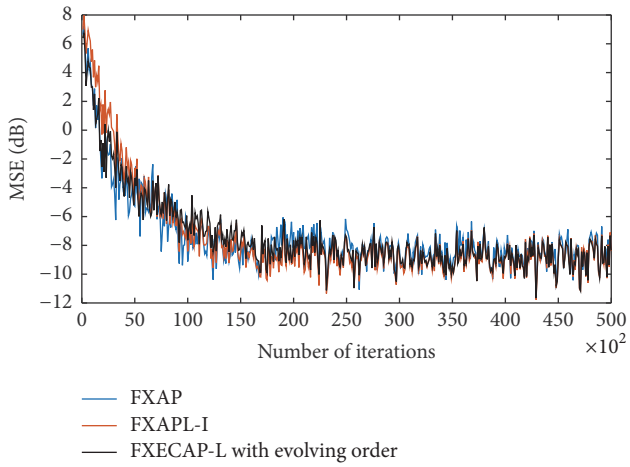

Figure 13: MSE values of the tested AP algorithms as measured at error sensor 1 for a projection order of $L = 5$ when $b = 8$ bits are used to encode the error.

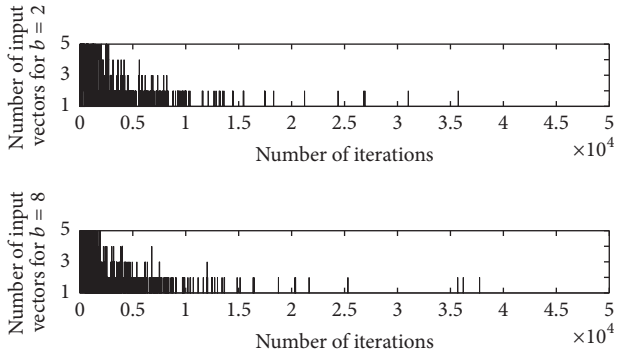

Figure 14: Numbers of input vectors used in the proposed FXECAP-L algorithm when $L = 5$ and $b = 2$ or $b = 8$.

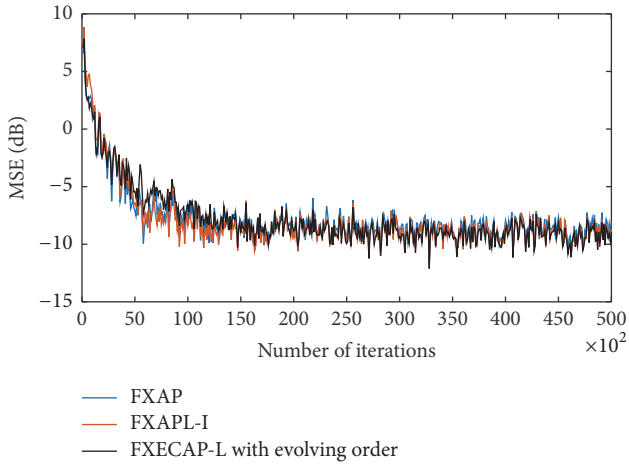

Figure 15: MSE values of the tested AP algorithms as measured at error sensor 1 for a projection order of $L = 10$ when $b = 2$ bits are used to encode the error.

presented. The results obtained at error sensor 2 were very similar to the presented results.

Figures 12 and 13 show the performances of the algorithms for a projection order of $L = 5$, where the numbers of bits used to encode the error were $b = 2$ and $b = 8$, respectively. The FXAP step size and the scaling factors for the FXAPL-I and FXECAP-L were adjusted by trial and error, and the values producing the fastest convergence speed were selected. Thus, for the FXAP algorithm, $\mu = 0.03$, for the FXAPL-I algorithm, sf = 0.01, for the FXECAP-L algorithm when $b = 2$, sf = 0.02, and for the FXECAP-L algorithm when $b = 8$, sf = 0.01. In this experiment, the proposed algorithm exhibited a very similar convergence speed and misalignment compared with the FXAP and FXAPL-I algorithms. The most important features were the reductions in the number of updates and the projection order.

Table 5 shows that the FXECAP-L algorithm with $b = 2$ performed almost 30% fewer updates compared with the other algorithms; however, when $b = 8$, the algorithm was updated in almost every iteration because of the effect of the resolution.

Figure 14 shows the numbers of input vectors used by the proposed algorithm in the experiments with an initial projection order of $L = 5$ and error encoding using $b = 2$ and $b = 8$ bits, from which it can be observed that the adaptive algorithm worked with high-order matrices only during the first approximately 5,000 iterations; subsequently, the algorithm worked with only one or two input vectors.

Figures 15 and 16 show the MSE values for $L = 10$ and error encoding using $b = 2$ and $b = 8$ bits, respectively. The FXAP step size and the scaling factors for the FXAPL-I and FXECAP-L were adjusted by trial and error; the values producing the fastest convergence speed were selected. Thus, for the FXAP algorithm, $\mu = 0.02$, for the FXAPL-I algorithm, sf = 0.008, for the FXECAP-L algorithm when $b = 2$, sf = 0.02, and for the FXECAP-L algorithm when $b = 8$, sf = 0.01. In these cases, the convergence speed slightly increased, and for $b = 2$, the algorithm again avoided updating the coefficients in each iteration, as shown in Table 5.

For a projection order of $L = 10$, the number of updates and the convergence speed were similar to those achieved for

Table 5: Numbers of updates and update percentages for the multichannel ANC system.

Algorithm	Updates (percentage)
AP	50,000 (100%)
APL-I	50,000 (100%)
ECAP-L with evolving order, $b = 2$ and $L = 5$	35,812 (71.624%)
ECAP-L with evolving order, $b = 2$ and $L = 10$	35,212 (70.424%)
ECAP-L with evolving order, $b = 8$ and $L = 5$	49,992 (99.984%)
ECAP-L with evolving order, $b = 8$ and $L = 10$	49,990 (99.8%)

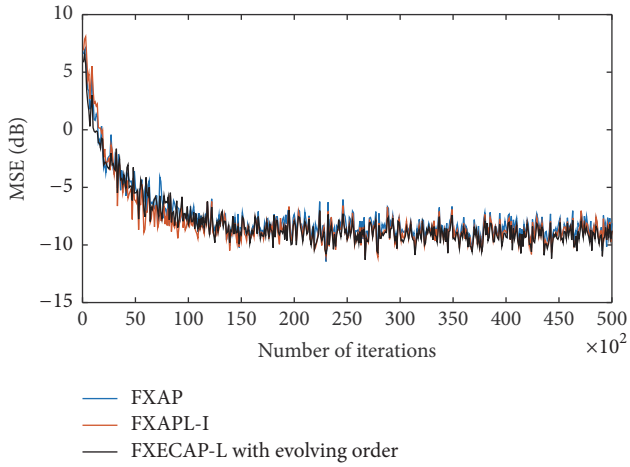

Figure 16: MSE values of the tested AP algorithms as measured at error sensor 1 for a projection order of $L = 10$ when $b = 8$ bits are used to encode the error.

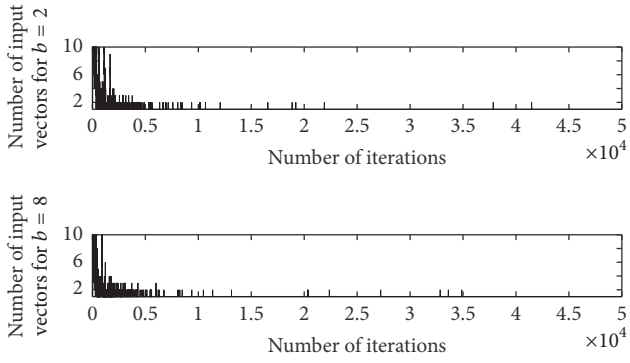

Figure 17: Numbers of input vectors used in the proposed FXECAP-L algorithm when $L = 10$ and $b = 2$ or $b = 8$.

$L = 5$, whereas the number of input vectors was reduced considerably (see Figure 17) because the algorithm converged faster and the thresholds for the evolution order were reached sooner.

In Table 6, the number of multiplications per iteration required by the FXECAP-L algorithm is compared with the numbers of multiplications for the FXAP and FXAPL-I algorithms. Table 6 shows the results for both experiments, $L = 5$ and $L = 10$.

As in the single-channel experiments, the FXECAP-L algorithm presents a higher number of multiplications than the FXAP-L algorithm because the calculation is performed for a constant projection order; nevertheless, using the evolving projection order and the thresholds for the update of the algorithm, the number of multiplications is reduced.

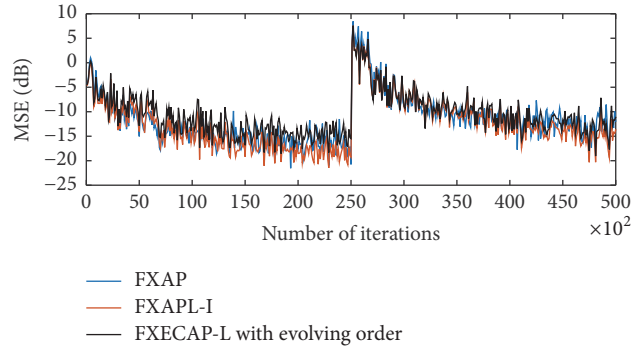

Figure 18: MSE values of the tested AP algorithms in a variant multichannel system. The projection order used is $L = 5$ and $b = 2$ bits are used to encode the error.

Figure 18 shows the tracking performance of the proposed algorithm when the primary path coefficients are multiplied by −1 at iteration 25000. For this case, at the microphone sensor, white Gaussian noise was added until a 30 dB signal to noise ratio (SNR) was achieved. The projection order for this experiment is $L = 5$ and $b = 2$ bits are used to encode the error. The FXAP step size and the scaling factors for the FXAPL-I and FXECAP-L were adjusted by trial and error; the values producing the fastest convergence speed were selected. Thus, for the FXAP algorithm, $\mu = 0.001$, for the FXAPL-I algorithm, sf = 0.008, and for the FXECAP-L algorithm, sf = 0.02. As it can be seen, the proposed algorithm maintained its tracking performance without degrading the convergence rate or steady-state estimation error. The number of updates of the proposed algorithm was 25784 (51.56%), since the error signal was smaller and so the threshold established for updating the coefficients was reached with less iterations.

Figure 19 shows the MSE obtained when 8 bits are used to encode the error in the proposed algorithm. The scaling factor used in this case was sf = 0.01. As it can be seen, there is only a slightly increased on the convergence speed. On the other hand, the number of updates of the algorithm was 49981 (99.96%).

Figure 20 shows the number of input vectors obtained from the tracking experiment with an initial projection order of $L = 5$ and error encoding using $b = 2$ and $b = 8$ bits.

Table 6: Comparison of the computational burden of the FXECAP-L algorithm with those of the FXAP and FXAPL-I algorithms for ANC systems with $N = 100$ and $M = 128$.

Algorithm	$I = 1, J = 2, K = 2, L = 5$	$I = 1, J = 2, K = 2, L = 10$
FXAP	13712	49512
FXAPL-I	5536	9556
FXECAP-L	7556	13586

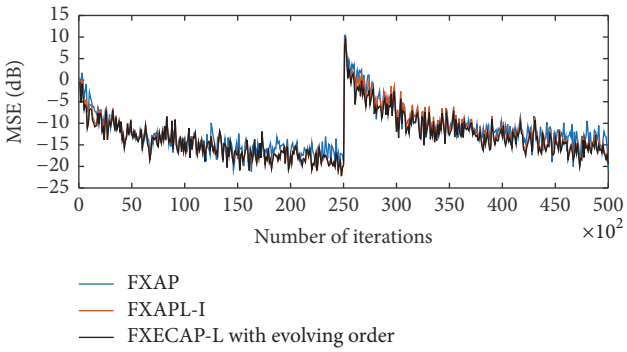

Figure 19: MSE values of the tested AP algorithms in a variant multichannel system. The projection order used is $L = 5$ and $b = 8$ bits are used to encode the error.

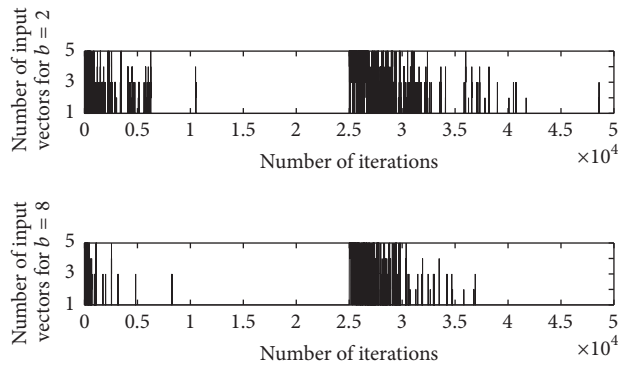

Figure 20: Numbers of input vectors used in the proposed FXECAP-L algorithm when $L = 5$ and $b = 2$ or $b = 8$.

For both cases, after the variation of the path the algorithm uses a higher number of input vectors during more iterations; nevertheless, it is able to reduce the projection order to one. For the case $b = 8$, the number of input vectors was reduced considerably because the algorithm converged faster and the thresholds for the evolution order were reached sooner.

The experiments performed for single-channel and multichannel ANC systems show that using a lower number of bits to encode the error reduces the number of updates performed by the algorithm. The number of updates required in the case of the multichannel system was greater than that required for the single-channel system; however, the convergence speed for the multichannel system was nearly identical to those of both the FXAP algorithm and the FXAPL-I algorithm. Moreover, the evolutionary method of the proposed algorithm allowed the projection order to decrease as the algorithm progressed, whereas the other algorithms used the same projection order throughout the entire process.

However, when the error was encoded with a higher number of bits, although the convergence speed increased, the algorithm updated the coefficients in almost every iteration. Another important observation from both experiments is that using a high projection order results in a faster reduction of the number of input vectors.

6. Conclusions

In this paper, we introduce the Filtered-X Error Coded Affine Projection-Like algorithm with evolving order for single-channel and multichannel ANC systems. The proposed algorithm updates its coefficients only when the error is higher than some predetermined threshold. Furthermore, the projection order changes dynamically by means of an evolutionary method that compares the error against two thresholds derived from the instantaneous value of the steady-state MSE. Simulation results demonstrate that the proposed algorithm inherits the fast convergence speed and misadjustment level of the APA and does not require matrix inversion operations. Moreover, the proposed algorithm reduces the number of coefficient updates by more than 50% for a single-channel system and by almost 30% for a multichannel system, and it avoids the use of high projection orders throughout the entire computation process. The FXECAP-L algorithm can be an excellent alternative for the implementation of ANC systems because it has a low overall computational complexity compared with other algorithms based on affine subspace projections.

Notations

I: Number of reference sensors
J: Number of actuators
K: Number of error sensors
N: Length of the adaptive FIR filters
L: Projection order
M: Length of the fixed FIR filters
$x_i(n)$: ith reference signal
$y_j(n)$: jth canceling signal
$e_k(n)$: kth error signal
$\mathbf{w}_{ji}(n)$: Weight vectors of the adaptive filters
$\mathbf{s}_{kj}(n)$: Impulse responses of the secondary paths
$\hat{\mathbf{s}}_{kj}(n)$: Estimates of the secondary paths.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] S. J. Elliott and P. A. Nelson, "Active noise control," *IEEE Signal Processing Magazine*, vol. 10, no. 4, pp. 12–35, 1993.

[2] S. M. Kuo and D. Morgan, *Active Noise Control Systems: Algorithms and DSP Implementations*, John & Wiley Sons, Inc, New York, NY, USA, 1st edition, 1995.

[3] Y. Kajikawa, W. S. Gan, and S. M. Kuo, "Recent advances on active noise control: open issues and innovative applications," *APSIPA Transactions on Signal and Information Processing*, vol. 1, article e3, 21 pages, 2012.

[4] N. V. George and G. Panda, "Advances in active noise control: a survey, with emphasis on recent nonlinear techniques," *Signal Processing*, vol. 93, no. 2, pp. 363–377, 2013.

[5] E. Bjarnason, "Active noise cancellation using a modified form of the filtered-x lms algorithm," in *Proceeding of the 6th European Signal Processing Conference*, vol. 2, pp. 1053–1056, 1992.

[6] D.-C. Chang and F.-T. Chu, "Feedforward active noise control with a new variable tap-length and step-size filtered-X LMS algorithm," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 22, no. 2, pp. 542–555, 2014.

[7] B. Huang, Y. Xiao, J. Sun, and G. Wei, "A variable step-size FXLMS algorithm for narrowband active noise control," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 21, no. 2, pp. 301–312, 2013.

[8] M. Ferrer, A. Gonzalez, M. De Diego, and G. Pinero, "Convex combination filtered-X algorithms for active noise control systems," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 21, no. 1, pp. 154–165, 2013.

[9] C. Mosquera and F. Pérez-González, "Convergence analysis of the multiple-channel filtered-U recursive LMS algorithm for active noise control," *Signal Processing*, vol. 80, no. 5, pp. 849–856, 2000.

[10] R. Shah, S. Reddy, V. Patel, and N. V. George, "Improving convergence in finite word length nonlinear active noise control systems," in *Proceeding of the IEEE International Conference on Digital Signal Processing, DSP 2015*, pp. 562–565, July 2015.

[11] Z. Zecevic, B. Krstajic, and M. Radulovic, "Frequency-domain adaptive algorithm for improving the active noise control performance," *IET Signal Processing*, vol. 9, no. 4, pp. 349–356, 2015.

[12] K. Ozeki and T. Umeda, "An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties," *Electronics and Communications in Japan (Part I: Communications)*, vol. 67, no. 5, pp. 19–27, 1984.

[13] M. Bouchard, "Multichannel affine and fast affine projection algorithms for active noise control and acoustic equalization systems," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 1, pp. 54–60, 2003.

[14] M. Ferrer, M. De Diego, A. González, and G. Pinero, "Efficient implementation of the affine projection algorithm for active noise control applications," in *Proceeding of the 12th European Signal Processing Conference*, pp. 929–932, IEEE, 2004.

[15] M. Bouchard and F. Albu, "The Gauss-Seidel fast affine projection algorithm for multichannel active noise control and sound reproduction systems," *International Journal of Adaptive Control and Signal Processing*, vol. 19, no. 2-3, pp. 107–123, 2005.

[16] F. Albu, M. Bouchard, and Y. Zakharov, "Pseudo-affine projection algorithms for multichannel active noise control," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 3, pp. 1044–1052, 2007.

[17] M. Ferrer, A. Gonzalez, M. De Diego, and G. Piñero, "Fast affine projection algorithms for filtered-x multichannel active noise control," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 8, pp. 1396–1408, 2008.

[18] L. Rey Vega, H. Rey, and J. Benesty, "A robust variable step-size affine projection algorithm," *Signal Processing*, vol. 90, no. 9, pp. 2806–2810, 2010.

[19] F. Albu, C. Paleologu, and S. Ciochina, "New variable step size affine projection algorithms," in *Proceeding of the 9th International Conference on Communications, COMM 2012*, pp. 63–66, IEEE, June 2012.

[20] F. Albu and C. Paleologu, "The variable step-size gauss-seidel pseudo affine projection algorithm," *World Academy of Science, Engineering and Technology*, vol. 37, pp. 642–645, 2009.

[21] J.-M. Song and P. Park, "An optimal variable step-size affine projection algorithm for the modified filtered-x active noise control," *Signal Processing*, vol. 114, pp. 100–111, 2015.

[22] S.-J. Kong, K.-Y. Hwang, and W.-J. Song, "An affine projection algorithm with dynamic selection of input vectors," *IEEE Signal Processing Letters*, vol. 14, no. 8, pp. 529–532, 2007.

[23] S.-E. Kim, S.-J. Kong, and W.-J. Song, "An affine projection algorithm with evolving order," *IEEE Signal Processing Letters*, vol. 16, no. 11, pp. 937–940, 2009.

[24] J. W. Yoo, J. W. Shin, H. Choi, and P. G. Park, "An affine projection algorithm with evolving order using variable step-size," *International Journal of Computer and Electrical Engineering*, vol. 5, no. 1, article 5, 2013.

[25] R. Arablouei and K. Dogançay, "Affine projection algorithm with selective projections," *Signal Processing*, vol. 92, no. 9, pp. 2253–2263, 2012.

[26] A. Gonzalez, F. Albu, M. Ferrer, and M. de Diego, "Evolutionary and variable step size strategies for multichannel filtered-x affine projection algorithms," *IET Signal Processing*, vol. 7, no. 6, pp. 471–476, 2013.

[27] A. Rodriguez, J. G. Avalos, and J. C. Sanchez, "Filtered-x error coded affine projection algorithm with evolving order for active noise control in," in *Proceeding of the 8th Latin American Symposium on Circuits & Systems (LASCAS)*, IEEE, 2017.

[28] A. Rodriguez, J. C. Sanchez, and J. G. Avalos, "Error coded affine projection like algorithm," in *Proceeding of the International Conference on Mechatronics, Electronics, and Automotive Engineering, ICMEAE 2015*, pp. 70–75, mex, November 2015.

[29] M. Z. A. Bhotto and A. Antoniou, "Affine-projection-like adaptive-filtering algorithms using gradient-based step size," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 7, pp. 2048–2056, 2014.