applied sciences

MDPI

Brief Report

A Time-Efficient Method for Determining an Optimal Scaling Factor and the Encoder Resolution in the Multichannel FXECAP-L Algorithm with Evolving Order for Active Noise Control

Ángel A. Vázquez, Xochitl Maya, Juan G. Avalos *, Giovanny Sánchez, Juan C. Sánchez, Hector M. Pérez and Gabriel Sánchez

Instituto Politécnico Nacional, ESIME Culhuacan, Av. Santa Ana No. 1000, Ciudad de México 04260, Mexico; vapa1995@hotmail.com (Á.A.V.); xomaro94@hotmail.com (X.M.); giovas666@hotmail.com (Gi.S.); jcsanchezgarcia@gmail.com (J.C.S.); hmperezm@ipn.mx (H.M.P.); caaann@gmail.com (Ga.S.)

* Correspondence: javaloso@ipn.mx; Tel.: +52-55-2101-9551

Received: 5 December 2018; Accepted: 1 February 2019; Published: 8 February 2019

Abstract: Affine projection (AP) algorithms have demonstrated faster convergence speed than conventional least mean square (LMS) algorithms, thus providing an attractive solution in the active noise control (ANC) field. However, the AP algorithms demand high computational cost, restricting their practical use in real-time ANC applications. Recently, a multichannel filtered-x error-coded affine projection-like (FXECAP-L) algorithm with evolving order has been proposed to reduce the computational burden by maintaining the convergence speed of AP algorithms. In order to obtain an efficient and robust FXECAP-L algorithm with evolving order, the scaling factor and encoder resolution need to be adjusted manually, which is a time-consuming and costly effort that must be carried out by expert designers. To reduce these costs and efforts, we introduce, for the first time, a strategy for automatic adjustment of the scaling factor and encoder resolution that benefits the rapid development of practical ANC applications. To demonstrate its practical use, we applied the proposed strategy for controlling the noise in an acoustic duct. The practical results demonstrate the automatic adjustment of the FXECAP-L algorithm which maintains high convergence speed at the expense of a small compromise in terms of processing time.

Keywords: active noise control; filtered-x error-coded affine projection-like algorithm; acoustic duct

1. Introduction

Active noise control (ANC) systems have been intensively studied in recent years [1,2]. These systems commonly use adaptive filters to cancel a primary noise by means of a cancelling noise. One of the most popular adaptive algorithms is the affine projection (AP) algorithm [3] that has been widely used for computing ANC systems since AP algorithms present faster convergence speed compared with least mean square (LMS) algorithms. However, AP algorithms demand high computational power to process a large number of input vectors L (called projection order) to update the weight vector. Nevertheless, the use of a high projection order increases the convergence speed. A challenging task is to find an optimal projection order whilst maintaining fast convergence speed.

Some authors have proposed techniques to minimize the computational complexity of the AP algorithms applied to ANC systems [4,5]. However, many of them have some restrictions rendering their use unfeasible in practical applications. Recently, Avalos et al. [6] introduced a new variant of the AP algorithms based on an affine projection-like-I (APL-I) algorithm. The authors proposed a strategy to update the filter coefficients only when the estimation error is higher than a pre-determined

threshold and it uses a method to adjust the projection order dynamically. Their results demonstrate that the filtered-x error-coded affine projection-like FXECAP-L algorithm with evolving order reduces the computational burden and maintains the high-speed convergence. This potentially allows the use of this algorithm in ANC systems, requiring lower computational power compared with AP algorithms whilst maintaining high convergence speed. However, the scaling factor and the encoder resolution of the algorithm need to be adjusted manually, which represents a great effort for the designers, in order for it to be used in practical applications.

In particular, several authors have employed the FXLMS algorithm in ANC systems for acoustic ducts due to its low computational cost [7–9]. However, in these ANC systems, fast convergence and stability are vital. Therefore, the use of the APL-I algorithm could improve these factors. In addition, the development of APL-I algorithms with automatic adjustment is still a big challenge [10,11]. In this paper, we present, for the first time, a strategy to automatically adjust the scaling factor and the encoder resolution of the FXECAP-L algorithm with evolving order. The proposed strategy to automatically adjust the scaling factor is based on the estimation of the mean squared value of the error vector. Meanwhile, the automatic adjustment of the encoder resolution was solved using the instantaneous value of the steady-state mean square error. We intend to provide an adaptive algorithm that can be used in practical ANC applications, such as automotive, appliance, industrial and transportation applications. Here, we prove its practicality by developing an ANC system for controlling the noise in an acoustic duct. Our results demonstrate that the automatic scaling factor and encoder resolution have helped us to rapidly develop a particular ANC application.

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

The FXECAP-L algorithm with evolving order was introduced by Avalos et al. [6]. This algorithm aims at reducing the computational cost by employing two techniques. The first technique is linked to the update of the filter coefficients based on the estimation error, i.e., there is an update process when the error is higher than a pre-determined threshold. The second technique is based on the calculation of projection order in each iteration [6]. The block diagram of a multichannel ANC system using the FXECAP-L algorithm is shown in Figure 1. The blocks P and S represent the primary and secondary paths, respectively, K is the number of error sensors, J the number of actuators, $x_i(n)$ the ith reference signal obtained from I reference sensors and $y_j(n)$ is the adaptive filter output given by

$$y_j(n) = \mathbf{w}_{ji}^t(n)\mathbf{x}_i(n). \tag{1}$$

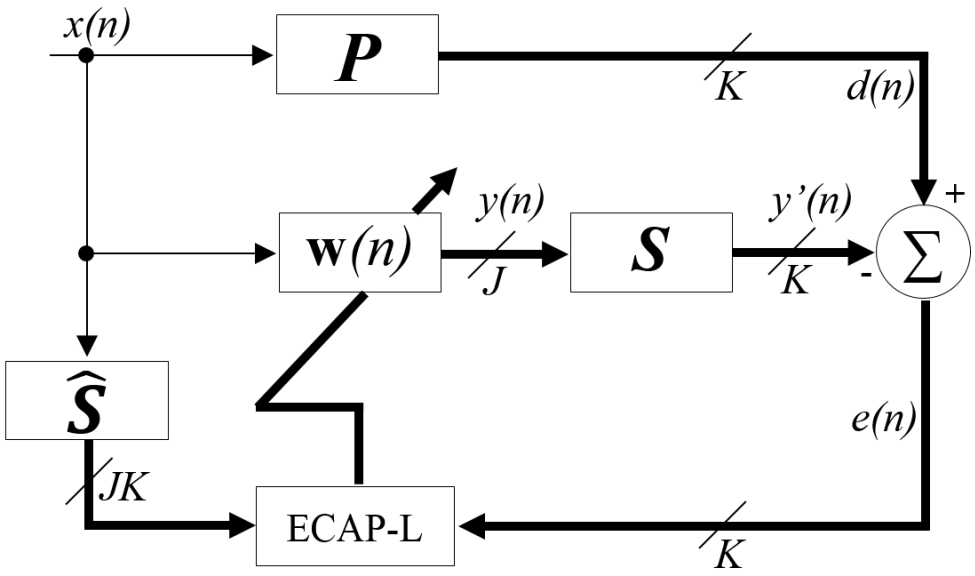

Figure 1. Block diagram of the FXECAP-L algorithm for active noise control.

The reference signal is defined as $\mathbf{x}_i(n) = \begin{bmatrix} x_i(n) & x_i(n-1) & \dots & x_i(n-N+1) \end{bmatrix}^t$ and the adaptive filter coefficients as $\mathbf{w}_{ji}(n) = \begin{bmatrix} w_{j,i,1}(n) & w_{j,i,2}(n) & \dots & w_{j,i,N} \end{bmatrix}^t$, where N is the length of the adaptive filter. In practice, the desired signal $\mathbf{d}(n)$ is unavailable, and thus the error signal vector $\mathbf{e}(n)$ is estimated using an approximation. Therefore, the estimation is obtained using past samples of the error signals and is expressed as $\mathbf{e}_k(n) \approx \begin{bmatrix} e_k(n) & e_k(n-1) & \dots & e_k(n-L+1) \end{bmatrix}^t$, where L is the projection order. To compensate the effects caused by the secondary paths $\mathbf{s}_{kj}(n)$, the reference signal $x_i(n)$ is filtered using the following expression:

$$x'_{ijk}(n) = \hat{\mathbf{s}}^t_{kj}(n)\mathbf{v}_i(n), \tag{2}$$

where $\hat{\mathbf{s}}_{kj}$ is a fixed FIR filter of length M that models $\mathbf{s}_{kj}(n)$ and $v_i(n) = \begin{bmatrix} x_i(n) & x_i(n-1) & \dots & x_i(n-M+1) \end{bmatrix}^t$.

To calculate the coefficient update equation, the filtered-x signals are arranged in a matrix $\mathbf{X}_{ijk}(n) = \begin{bmatrix} \mathbf{x}'_{ijk}(n) & \mathbf{x}'_{ijk}(n-1) & \dots & \mathbf{x}'_{ijk}(n-L+1) \end{bmatrix}$, composed of the filtered-x signals $\mathbf{x}'_{ijk}(n) = \begin{bmatrix} x'_{ijk}(n) & x'_{ijk}(n-1) & \dots & x'_{ijk}(n-N+1) \end{bmatrix}^t$. Thus, the filter update equation for the FXECAP-L algorithm is expressed as follows:

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \sum_{k=1}^{K} \mu_{ijk}\mathbf{X}_{ijk}(n)\mathbf{C}[\mathbf{e}_k(n)], \tag{3}$$

where the error is encoded using $\mathrm{C}[\mathbf{e}_k(n)] = round(\mathbf{e}_k(n)/Res)$, and the resolution is obtained as follows:

$$Res = \frac{\mathrm{e}_{\max}}{2^b - 1}. \tag{4}$$

Here, $\mathrm{e}_{\max}$ denotes the maximum probable error and b is the number of bits used to encode the error. The step size μ_{ijk} can be calculated using the following expression:

$$\mu_{ijk} = \frac{\| \mathbf{X}_{ijk}(n)\mathbf{e}_k(n) \|^2}{\| \mathbf{X}^t_{ijk}(n)\mathbf{X}_{ijk}(n)\mathbf{e}_k(n) \|^2} Res \cdot sf, \tag{5}$$

where sf is a scaling factor used to compensate for the mismatch between $\hat{\mathbf{s}}_{kj}(n)$ and $\mathbf{s}_{kj}(n)$. Commonly, the value of sf can be defined in the interval $[0, 1]$ and the designer can choose it by trial and error. The ECAP-L algorithm reduces the computational burden by updating the filter coefficients when the error signal is larger than a pre-established threshold. Such a threshold is determined by the error magnitude; in this way, when the error signal is small, the coefficients are not updated. The update threshold is defined as follows:

$$\mathbf{w}_{ji}(n+1) = \begin{Bmatrix} \mathbf{w}_{ji}(n), & if \quad \mathrm{C}[\mathbf{e}_k(n)] = 0 \, or \, 1, \\ \text{update the coefficient}, & if \quad \mathrm{C}[\mathbf{e}_k(n)] \neq 0 \, or \, 1. \end{Bmatrix} \tag{6}$$

As mentioned in [6], when the error is encoded with a lower number of bits, the convergence speed is slower. However, using a large number of bits increases the convergence speed, but the update threshold is not reached, and thus the algorithm updates its coefficients in almost all the iterations.

The FXECAP-L algorithm dynamically calculates the projection order using a strategy based on the instantaneous residual error power at each error sensor. The rule of variation of the projection order is defined as

$$L_{ij}(n) = \begin{Bmatrix} min\{L_{ij}(n-1)+1, L_{max}\}, & \eta_{ij}(n) < \mathbf{e}^2_k(n), \\ L_{ij}(n-1), & \theta_{ij}(n) < \mathbf{e}^2_k(n) \leq \eta_{ij}(n), \\ max\{L_{ij}(n-1)-1, 1\}, & \mathbf{e}^2_k(n) \leq \theta_{ij}(n), \end{Bmatrix} \tag{7}$$

where L_{ij} is the projection order used in each iteration; L_{max} is the maximum projection order; $\theta_{ij}(n)$ and $\eta_{ij}(n)$ are thresholds derived from the steady-state mean square error of the algorithm, and can be expressed as

$$\theta_{ij}(n) = \frac{2L_{ij}}{2L_{ij}-1}\sigma_v^2 \tag{8}$$

and

$$\eta_{ij}(n) = \frac{2L_{ij}+1}{2L_{ij}}\sigma_v^2, \tag{9}$$

where σ_v^2 is the variance of a noise signal.

3. Time-Efficient Method to Determine an Optimal Scaling Factor and the Encoder Resolution

The most critical parameters of the existing FXECAP-L algorithm are the scaling factor and the encoder resolution since the convergence can be affected by the designer choosing a wrong value. To date, there is no established method to determine either the number of bits that are used to encode the error or scaling factor in ANC applications. The following section presents a new time-efficient method to dynamically adjust such parameters in order to optimize the designing time.

3.1. Dynamic Adjustment of the Number of Bits for the Encoder Resolution

Rodriguez et al. [12] proposed a method to adjust the encoder resolution of the ECAP-L algorithm for acoustic echo cancellation applications. Nevertheless, this variant has not been applied for active noise control, for which the adaptive configuration is very different. To adjust the number of bits for the encoder in each iteration, the instantaneous value of the steady-state mean square error (see [6]) was used. Thus, the number of bits is adjusted as follows:

$$b_{ij}(n) = \begin{Bmatrix} min\{b_{ij}(n-1)+1, b_{max}\}, & \eta_{ij}(n) < \mathbf{e}_k^2(n), \\ b_{ij}(n-1), & \theta_{ij}(n) < \mathbf{e}_k^2(n) \leq \eta_{ij}(n), \\ max\{b_{ij}(n-1)-1, 1\}, & \mathbf{e}_k^2(n) \leq \theta_{ij}(n), \end{Bmatrix} \tag{10}$$

where b_{ij} is the number of bits for each adaptive filter and b_{max} is the maximum number of bits. In this way, at the beginning of the process, a large number of bits is used to encode the error, and thus the algorithm provides a fast convergence speed. After that, the number of bits is reduced as long as the residual error power is less than $\theta_{ij}(n)$. Therefore, this mechanism guarantees a low computational cost since the FXECAP-L algorithm is not updated during all the iterations. The resolution of the encoder must be calculated in each iteration because the number of bits changes over the time, yielding a resolution $Res_{ij}(n)$ for each adaptive filter.

3.2. Automatic Scaling Factor

The time-efficient method that is used to automatically adjust the scaling factor is based on the estimation of an auxiliary error vector $\epsilon_{ijk}(n)$ [13]. For the FXECAP-L algorithm, the auxiliary vector is defined as $\epsilon_{ijk}(n) = \mathbf{X}_{ijk}(n)\mathbf{C}[\mathbf{e}_k(n)]$. Thus, the algorithm for the coefficient update can be rewritten as

$$\mathbf{w}_{ji}(n+1) = \mathbf{w}_{ji}(n) + \sum_{k=1}^{K} \mu_{ijk}\epsilon_{ijk}(n). \tag{11}$$

In this way, the proposed automatic scaling factor can be expressed as

$$sf_{ijk} = Res_{max}\frac{\|p_{ijk}(n)\|^2}{\|p_{ijk}(n)\|^2 + R}, \tag{12}$$

where Res_{max} is the initial value of the encoder resolution, $p_{ijk}(n)$ is an estimation of $\epsilon_{ijk}(n)$ that is obtained as $p_{ijk}(n) = \alpha p_{ijk}(n-1) + (1-\alpha)\epsilon_{ijk}(n)$ with a smoothing factor $(0 < \alpha < 1)$ and R

is a parameter related to the projection order and is approximated by L_{ij}/SNR, where SNR is the signal-to-noise ratio. The proposed time-efficient method uses Res_{max} (see Equation (12)) to guarantee the stability of the filter while another method uses a maximum step-size parameter. Therefore, the proposed time-efficient method establishes that the variation of sf (see Equation (12)) does not depend on the value of sf_{max} in order to avoid the divergence of the algorithm. Hence, the convergence is achieved by regularizing the step size μ_{ijk} by means of the resolution. The proposed time-efficient method to compute the step size μ_{ijk} is as follows:

$$\mu_{ijk} = \frac{\| \mathbf{X}_{ijk}(n)\mathbf{e}_k(n) \|^2}{\| \mathbf{X}_{ijk}^t(n)\mathbf{X}_{ijk}(n)\mathbf{e}_k(n) \|^2} Res_{ij} \cdot sf_{ijk}. \tag{13}$$

Using the previous time-efficient method, the proposed algorithm requires $(3N + 1)IJK$ multiplications more than the existing FXECAP-L algorithm with an evolving order at each iteration. Therefore, it is clear that this strategy contributes little to the overall computational complexity. Here, the computational complexity is defined in terms in the number of additions and multiplications per iteration. As can be observed in Table 1, the proposed method requires 1204 and 3218 more multiplications than the conventional FXECAP-L algorithm and the filtered-x affine projection-like-I (FXAPL-I) algorithm, respectively, by considering a typical case ($M = 128$, $N = 100$, $L = 5$, $I = 1$ and $J = K = 2$). On the other hand, the conventional FXECAP-L and FXAPL-I algorithms demand 800 and 2400 fewer additions, respectively, compared with the proposed method, as shown in Table 2. However, the computational cost of the proposed method decreases when the steady-state is reached, while the computational cost of the FXAPL-I algorithm is always constant.

Table 1. Number of multiplications per iteration of the FXAPL-I algorithm, the conventional FXECAP-L algorithm and the proposed algorithm. Typical case $M = 128$, $N = 100$, $L = 5$, $I = 1$ and $J = K = 2$.

Algorithm	Number of Multiplications	Typical Case
FXAPL-I algorithm	$[((2N + 1)L + 2N + M + 1)IJK + (N)IJ]$	5536
Conventional FXECAP-L algorithm	$[((3N + 1)L + 2N + 2 + M)IJK + (N)IJ + L(K)]$	7550
Proposed method	$[((3N + 1)L + 5N + 3 + M)IJK + (N)IJ + L(K)]$	8754

Table 2. Number of additions per iteration of the FXAPL-I algorithm, conventional FXECAP-L algorithm and the proposed algorithm. Typical case $M = 128$, $N = 100$, $L = 5$, $I = 1$ and $J = K = 2$.

Algorithm	Number of Additions	Typical Case
FXAPL-I algorithm	$[((2L + 1)N + M - 2)IJK + (N)IJ + J(I - 1)]$	5104
Conventional FXECAP-L algorithm	$[(3NL + M - 2)IJK + (N)IJ + J(I - 1)]$	6704
Proposed method	$[(3NL + M + 2N - 2)IJK + (N)IJ + J(I - 1)]$	7504

4. Simulation Results

We evaluate the performance of the proposed algorithm by considering a 1:2:2 multichannel ANC system. The tap length of the primary path, the secondary path, and the adaptive filter was of 256, 128 and 100 coefficients, respectively. These values were defined previously by considering a typical case ($M = 128$, $N = 100$, $L = 5$, $I = 1$ and $J = K = 2$). Therefore, the computational cost of the proposed algorithm is defined in Tables 1 and 2. The impulse response of the acoustics paths was obtained from [14]. In order to evaluate the behavior of the proposal, we performed 50 independent trials each with 50,000 iterations. Furthermore, to test the tracking capabilities, we caused an abrupt change to the impulse response of the acoustic primary path at the middle of the iterations. As the reference signal, we used white Gaussian noise with unit variance and we also added white Gaussian noise to the error microphones with SNR 30 dB. Figure 2 shows a comparison of the mean square error (MSE) at the error microphone 1, for the filtered-x affine projection-like-I (FXAPL-I) algorithm, the conventional FXECAP-L algorithm and the proposed algorithm. For this experiment, the projection order was $L = 5$

for all the algorithms and the error was encoded with $b = 6$ bits in the FXECAP-L algorithm and the proposed algorithm. The scaling factor for the FXAPL-I algorithm and the FXECAP-L algorithm was $sf = 0.008$, and $sf = 0.015$, respectively. As can be observed from Figure 2, the convergence speed for all the algorithms is almost identical.

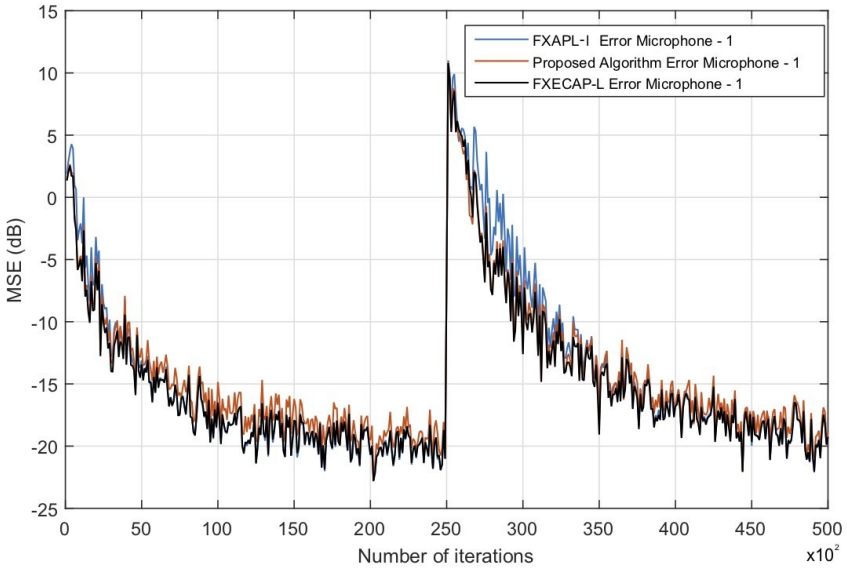

Figure 2. Comparison of the MSE curves for the FXAPL-I, FXECAP-L and the proposed algorithm.

Figure 3 shows the number of bits that are used to encode the error of the proposed algorithm. As can be seen, the number of bits decreases as the process advances. This allows the algorithm to stay within the update threshold, avoiding the computation of the algorithm in each iteration. In this way, the average number of updates performed by the proposed algorithm was 19,966 (39.93%), while the FXECAP-L algorithm updated its coefficients 49,610 times (99.2%).

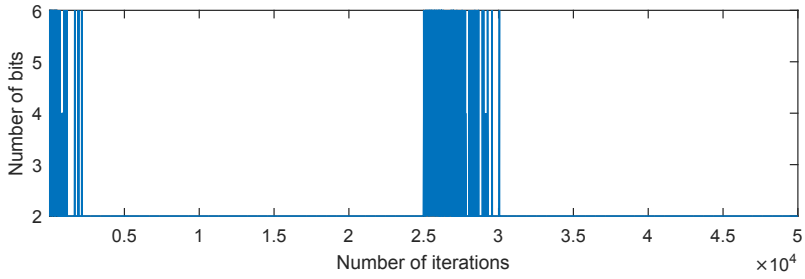

Figure 3. Number of bits used to encode the error in the proposed algorithm.

Figure 4 shows the number of input vectors for the proposed algorithm. As can be seen, the proposed strategy does not affect the evolving behavior; therefore, the computational burden is reduced.

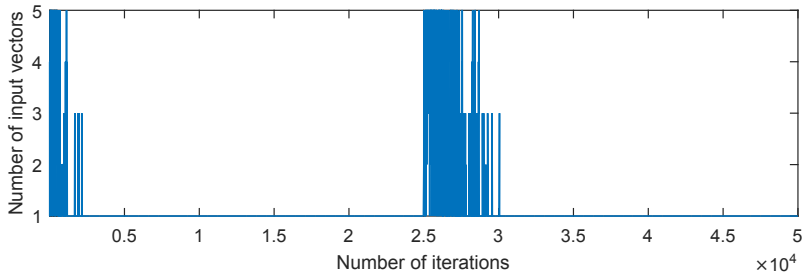

Figure 4. Number of input vectors of the proposed algorithm.

5. Experimental Verification

Once the control algorithm used in the multichannel ANC system was verified by simulations, we performed 1:1:1 ANC experiments in a wooden duct to validate it. The aim of this experiment was to show how the inclusion of the automatic scaling factor reduces the time required to develop real-time practical ANC applications since the manual setting of this factor is very time consuming. In addition, the proposed automatic adjustment of the scaling factor allows an easy adaptation of the control algorithm when abrupt changes in the reference signal occur. As a consequence, real-time analysis can be performed. The length and width of the duct are 1.21 m and 0.09 m, respectively, as shown in Figure 5.

In addition, the experimental set-up involves analogue low-pass filters, microphones, speakers, an audio power amplifier and a DSK board TMSC3206713, as shown in Figure 6. In general, we use analog low-pass filters to eliminate signals with high frequencies obtained from the microphones (#1 and #2). Here, the error microphone (#1) and reference microphone (#2) were employed to measure the residual error signal and reference signal, respectively. On the other hand, the noise source loudspeaker (#1) and the cancelling loudspeaker (#2) were used to generate a primary disturbance signal and anti-noise signal, respectively. Here, we used an audio power amplifier along with a loudspeaker (#2) to generate the anti-noise signal. The control algorithm was implemented in the DSK TM320C6713 from Texas Instruments (TI). In general terms, the error signal was measured by means of a microphone (#1), and then filtered by a low-pass filter. The resulting analog signal was converted to a digital signal to be processed by the DSK TM320C6713 board. Once the control algorithm generated the anti-noise signal, the DSK TM320C6713 board sent the analog signal to the power amplifier to ensure the acoustic cancellation via loudspeaker (#2).

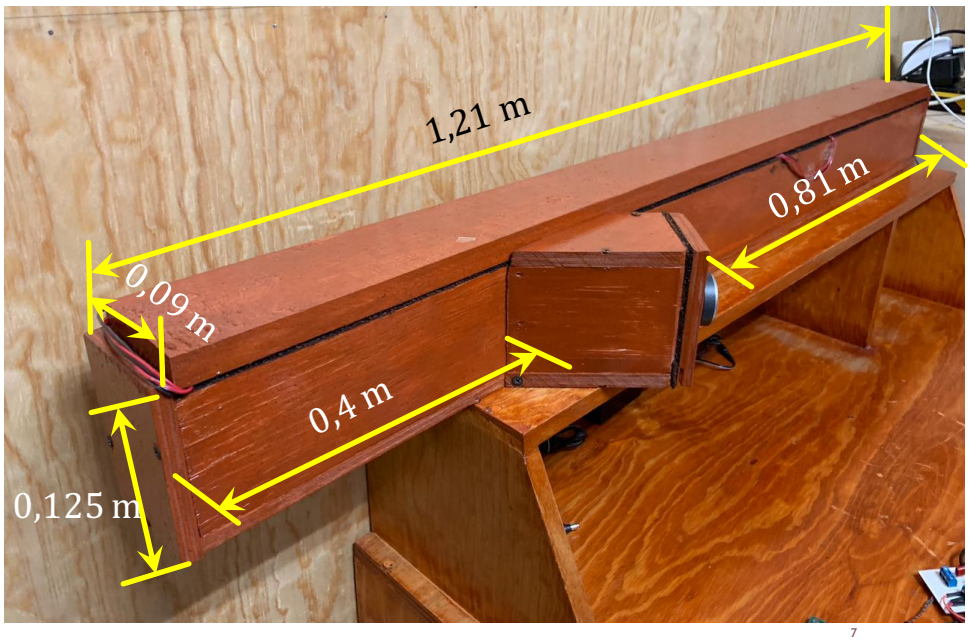

Figure 5. Acoustic duct used in ANC experiments.

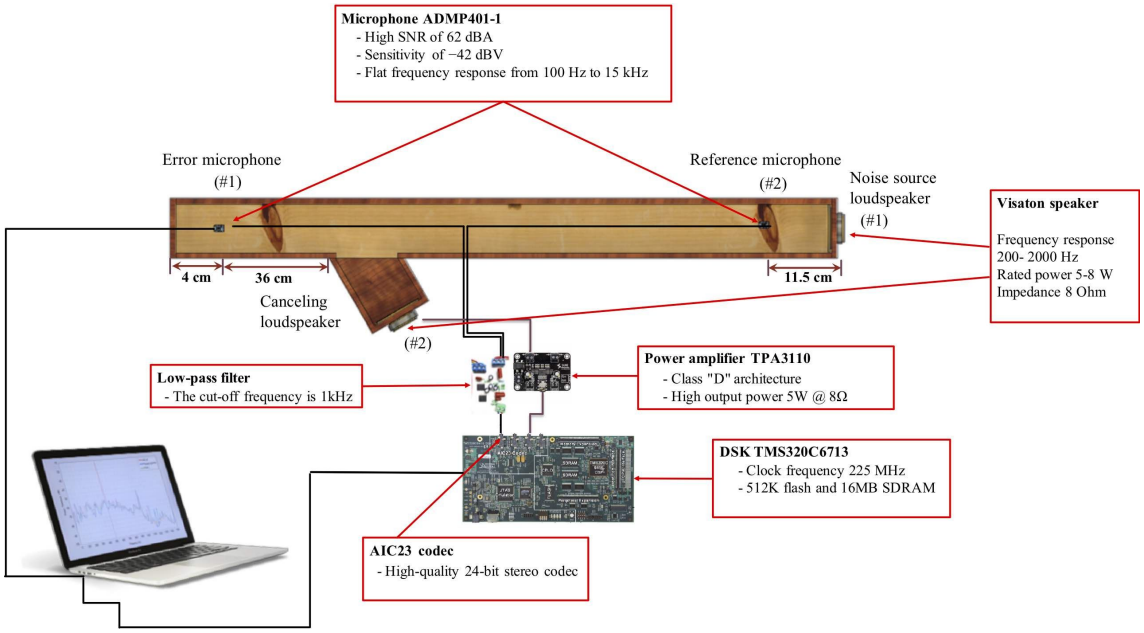

Figure 6. Experimental set-up for active noise reduction in a duct.

In this experiment, the transfer function of the secondary path was estimated offline using an LMS algorithm with 128 coefficients. Using this strategy, we can easily obtain the acoustic characteristics between the microphone (#1) and loudspeaker (#2). On the other hand, we considered two types of reference signal, a single-tonal input of 500 Hz and a multi-tonal input of 600, 750 and 900 Hz, in order to evaluate the ANC system in the acoustic duct. In addition, the variance of the noise and SNR were obtained by means of acoustic sensors.

Case 1: Single-tone as a reference signal

Figure 7 shows a comparison between the FXECAP-L algorithm, the FXAPL-I algorithm and the proposed algorithm in terms of the power spectrum of the error signal at the microphone (#1). In all algorithms, the projection order was $L = 5$ and the error was encoded with $b = 10$ bits. As can be observed from Figure 7, all algorithms can effectively attenuate noise. However, the automatic adjustment of the scaling factor has allowed the ANC system to be rapidly developed. In addition, the average number of updates performed by the proposed algorithm and the conventional FXECAP-L algorithm were 52,164 (48.07%) and 78,131 (72%), respectively. For simplicity, Figure 8 shows only how the projection order changes over time to reduce the computational complexity of the proposed algorithm. Here, the proposed algorithm and the conventional FXECAP-L algorithm exhibit similar behavior.

Case 2: Multi-tone as a reference signal

In this case, we compared the FXECAP-L algorithm, the FXAPL-I algorithm and the proposed algorithm. Figure 9 shows how the proposed algorithm performs the noise cancellation of a multi-tone signal. All of the algorithms effectively cancel multi-tone signals. However, the proposed algorithm and the conventional FXECAP-L algorithm require 51.18% (95,302) and 81% (150,829) fewer updates when compared with the FXAPL-I algorithm. As in the previous case, Figure 10 shows only the dynamic adjustment of the projection order of the proposed algorithm.

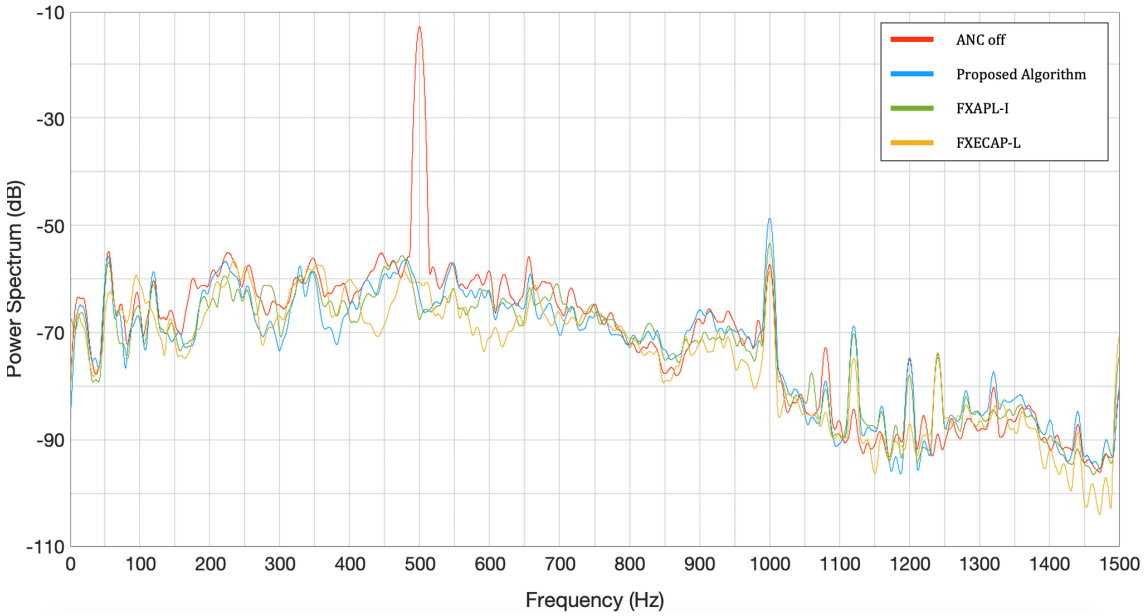

Figure 7. Power spectrum of the error signal at microphone (#1) considering a single-tone as a reference signal.

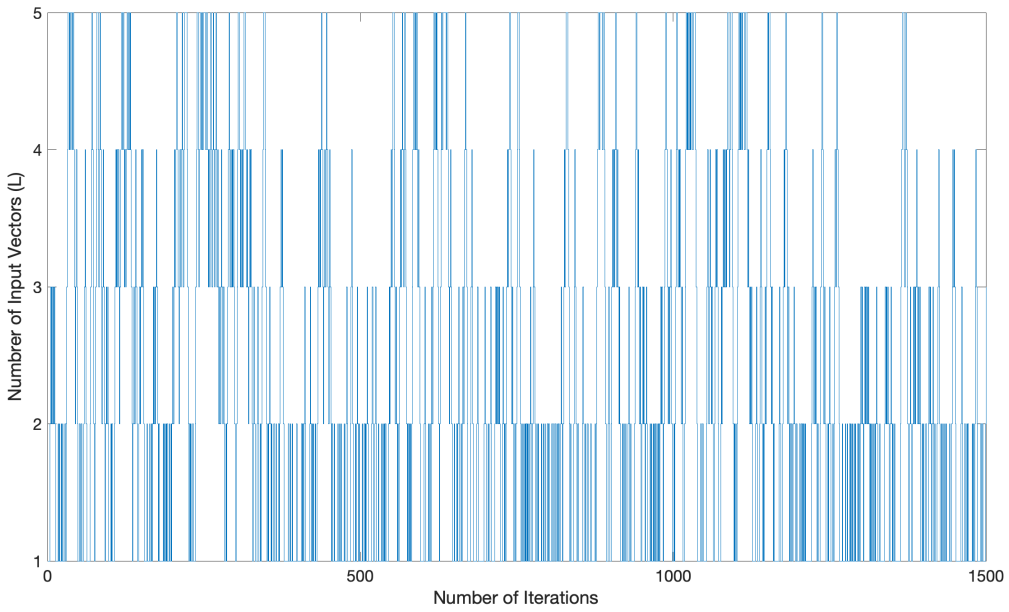

Figure 8. Zoomed view of the number of input vectors for case 1.

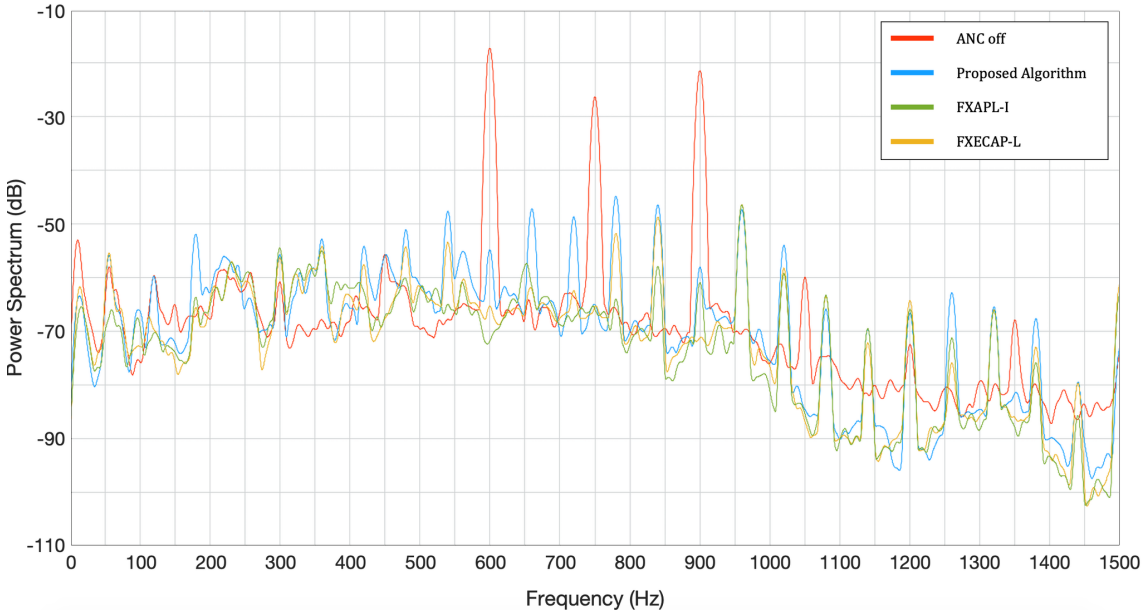

Figure 9. Power spectrum of the error signal at microphone (#1) considering a multi-tone as a reference signal.

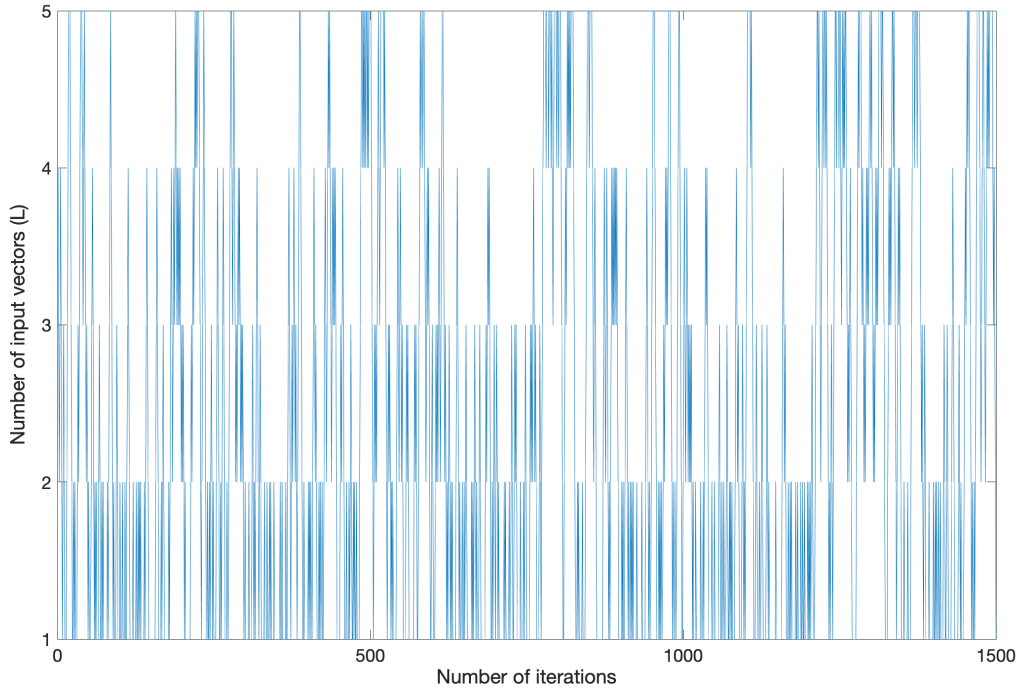

Figure 10. Zoomed view of the number of input vectors for case 2.

6. Conclusions

In this paper, we have presented a strategy to automatically adjust the value of the scaling factor and the encoder resolution for the multichannel FXECAP-L with evolving order. The simulation results confirm that the proposed algorithm maintains the high convergence speed and the computational savings of the original version. Furthermore, the number of updates is greatly reduced. In addition, the proposed ANC algorithm has been used in an acoustic duct to demonstrate its potential practical use.

Therefore, we intend to provide a useful tool for the development of practical real-time active noise control (ANC) applications. This could potentially be valuable to the ANC engineering communities.

Author Contributions: Validation and Investigation, A.A.V. and X.M.; Formal Analysis, Methodology, Supervision and Writing—Original Draft Preparation, J.G.A. and G.S. (Giovanny Sánchez); Project Administration, Conceptualization, J.C.S. and H.M.P.; Funding Acquisition, G.S. (Gabriel Sánchez).

Acknowledgments: The authors would like to thank the Consejo Nacional de Ciencia y Tecnologia (CONACYT) and the IPN for the financial support to realize this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kajikawa, Y.; Gan, W.S.; Kuo, S.M. Recent advances on active noise control: Open issues and innovative applications. *APSIPA Trans. Signal Inf. Process.* **2012**, *1*. [CrossRef]
2. George, N.V.; Panda, G. Advances in active noise control: A survey, with emphasis on recent nonlinear techniques. *Signal Process.* **2013**, *93*, 363–377. [CrossRef]
3. Ozeki, K.; Umeda, T. An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties. *Electron. Commun. Jpn.* **1984**, *67*, 19–27. [CrossRef]
4. Gonzalez, A.; Albu, F.; Ferrer, M.; de Diego, M. Evolutionary and variable step size strategies for multichannel filtered-x affine projection algorithms. *IET Signal Process.* **2013**, *7*, 471–476. [CrossRef]
5. Song, J.M.; Park, P. An optimal variable step-size affine projection algorithm for the modified filtered-x active noise control. *Signal Process.* **2015**, *114*, 100–111. [CrossRef]
6. Avalos, J.G.; Rodriguez, A.; Martinez, H.M.; Sanchez, J.C.; Perez, H.M. Multichannel Filtered-X Error Coded Affine Projection-Like Algorithm with Evolving Order. *Shock Vib.* **2017**, *2017*, 3864951. [CrossRef]
7. Kim, H.W.; Park, H.S.; Lee, S.K.; Shin, K. Modified-filtered-u LMS algorithm for active noise control and its application to a short acoustic duct. *Mech. Syst. Signal Process.* **2011**, *25*, 475–484. [CrossRef]
8. Ardekani, I.T.; Abdulla, W.H. On the convergence of real-time active noise control systems. *Signal Process.* **2011**, *91*, 1262–1274. [CrossRef]
9. Ardekani, I.T.; Abdulla, W.H. Effects of imperfect secondary path modeling on adaptive active noise control systems. *IEEE Trans. Control Syst. Technol.* **2012**, *20*, 1252–1262. [CrossRef]
10. Antoñanzas, C.; Ferrer, M.; de Diego, M.; Gonzalez, A. Affine-projection-like algorithm for active noise control over distributed networks. In Proceedings of the 2016 IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM), Rio de Janerio, Brazil, 10–13 July 2016; pp. 1–5.
11. Avalos, J.G.; Sanchez, G.; Rpdriguez, A.; Guevara, J.; Avalos, G. Multichannel Filtered-X Set-Membership Affine Projection-Like Algorithm. *IEEE Lat. Am. Trans.* **2018**, *16*, 2131–2137. [CrossRef]
12. Rodriguez, E.I.; Avalos, J.G.; Sanchez, J.C. Error coded affine projection-like algorithm with evolving order and variable resolution for acoustic echo cancellation. In Proceedings of the 2018 IEEE 9th Latin American Symposium on Circuits and Systems (LASCAS), Puerto Vallarta, Mexico, 25–28 February 2018; pp. 1–4.
13. Shin, H.C.; Sayed, A.H.; Song, W.J. Variable step-size NLMS and affine projection algorithms. *IEEE Signal Process. Lett.* **2004**, *11*, 132–135. [CrossRef]
14. Kuo, S.M.; Morgan, D.R. *Active Noise Control Systems: Algorithms and DSP Implementations*; Wiley: New York, NY, USA, 1996.