IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 6 August 2022, accepted 15 August 2022, date of publication 23 August 2022, date of current version 31 August 2022.

*Digital Object Identifier 10.1109/ACCESS.2022.3201135*

RESEARCH ARTICLE

# Filtered-x Set Membership Algorithm With Time-Varying Error Bound for Nonlinear Active Noise Control

**DINH CONG LE AND HOANG HUU VIET**
School of Engineering and Technology, Vinh University, Vinh 43108, Vietnam

Corresponding author: Dinh Cong Le (ldcong@vinhuni.edu.vn)

This work was supported by the Vietnam's Ministry of Education and Training under Grant B2021-TDV-03.

**ABSTRACT** In order to reduce the computational complexity for nonlinear Active Noise Control (ANC) systems, we propose a filtered-x set membership with a time-varying error bound (Fx-SM-VEB) algorithm based on the adaptive approximation principle and data-selective update strategy. By introducing the time-varying error bound (VEB) to replace a pre-specified threshold when nonlinearity exists in the components of the ANC system, the performance of the proposed algorithm is significantly improved. Moreover, the VEB is also easily expanded against the impulsive noise that is normally encountered in actual ANC systems. Based on the impulsive-free estimation, we develop a robust Fx-SM-VEB (RFx-SM-VEB) algorithm for the ANC system which is corrupted by impulsive noise in the reference input signal. Besides, we provide analyses of the steady-state behavior, stability conditions, and computational complexity of the proposed algorithms. Many simulation results in different scenarios of the ANC system have shown that the proposed algorithms are efficient under the nonlinear environment and impulsive noise.

**INDEX TERMS** Nonlinear active noise control, set membership, filtered-x least mean square (Fx-LMS) algorithm, impulsive noise.

## I. INTRODUCTION

For the purpose of noise reduction at low frequencies ($f$ <= 500 Hz), the ANC system has shown many outstanding advantages such as low cost, and simple implementation [1], [2], [3]. Its principle is based on the superposition of two sound wave sources, the primary path (unwanted noise source) and the secondary path (anti-noise source) to suppress unwanted noise. The noise cancellation quality of the ANC system depends not only on the correct coordination between the audio domain and the electrical domain, but also on the problem of reducing nonlinear distortion occurring in the primary path, secondary path, and noise source [4], [5]. Many studies have shown that the ANC system using a linear controller suffers from performance degradation because it does not take into account the influence of nonlinear distortion [4], [5], [6], [7], [8].

In recent literature [4], [5], many controllers based on linear-in-the-parameters (LIP) filters have been used to reduce nonlinear distortion. Prominent among them can be pointed out as the Functional Link Artificial Neural Networks (FLANN) [5], Recursive FLANN (RFLANN) [9], Generalized FLANN (GFLANN) [10], [11], Bilinear FLANN (BLANN) [12], Even Mirror Fourier (EMF) [13], [14], Volterra [4], [15], [16], etc. To compensate well for nonlinear distortion in ANC systems, LIP filters need to be chosen with a large enough order and/or memory length. This leads to a remarkable increase in computational complexity.

The strategy of updating the filter weights of the set-membership algorithm is to update only when the magnitude of the priori error exceeds a predefined threshold [17]. Based on this technique, data with no information or insignificant amount of information will be ignored, resulting in reduced update costs for redundant data. The set membership algorithm has been developed for many applications to reduce computational complexity [18], [19], [20], [21], [22], [23], [24]. In work [18], a nonlinear adaptive filter based on the

The associate editor coordinating the review of this manuscript and approving it for publication was Prakasam Periasamy.

kernel set membership method was introduced for the purpose of reducing complexity and increasing tracking ability. Many improvements to the set membership NLMS algorithm for the acoustic echo and sparse channel estimation problem have been proposed [19], [20], [21], [22]. To reduce power consumption in wireless sensor networks, solutions based on SM filtering strategy have also received the attention of many researchers [23], [24]. Much literature [25], [26], [27] dealing with identification systems in the set-membership framework exhibits robust performance in the impulsive noise environment. In particular, the filtered- x set-membership affine projection (Fx-SM-AP) algorithms have also been applied in the linear active noise control system to reduce the computation for the Fx-AP algorithm [28], [29].

In order to improve the efficiency of nonlinear ANC systems, in this study, we propose the filtered-x set membership with time-varying error bound (Fx-SM-VEB) and robust Fx-SM-VEB (RFx-SM-VEB) algorithms. The proposed Fx-SM-VEB algorithm updates only a small part of the filter weights, thus it can significantly reduce the computational complexity. Furthermore, due to using a time-varying error bound (VEB), the proposed algorithm is more suitable for nonlinear ANC systems, resulting in performance improvement compared to using the predetermined threshold. Besides, by calculating the threshold parameter through an impulsive-free estimation, the time-varying error bound can be easily applied to ANC systems containing impulsive noise. In practice, impulsive noise which often occurs with low probability, but large amplitude is a challenging problem for the stability of adaptive algorithms [15]. The proposed RFx-SM-VEB algorithm can against the impulse noise because the error bound is robust to outliers and achieves a good convergence performance resulting from a time-varying error bound. Many simulation studies have been conducted to evaluate the performance of the proposed algorithms in different scenarios of ANC systems.

The remainder of the paper is structured as follows. Section II provides a summary of the ANC system based on the LIP nonlinear expansion function. Section III proposes the set membership filter-based algorithms for the nonlinear ANC system. Sections IV and V give the analysis and the computational complexity of the algorithm, respectively. Sections VI discusses simulation results. Section VII concludes our work.

## II. BRIEF OF NONLINEAR ANC SYSTEM

Many linear-in-the-parameters (LIP) nonlinear filters (such as Volterra, FLANN, EMF, etc.) have been used to replace neural networks in ANC systems because of their simple structure and low complexity. Many researchers [4], [5], [9], [10], [11], [12], [13], [14], [15], [16] have also shown the effectiveness of the LIP filters for reducing the nonlinear distortion that exists in the primary path, secondary path, or reference noise. Fig. 1 illustrates a nonlinear ANC system based on LIP controller using Fx-LMS algorithm [1], [2], where $\boldsymbol{X}(n)$ is the reference noise source; $y(n)$ is the output

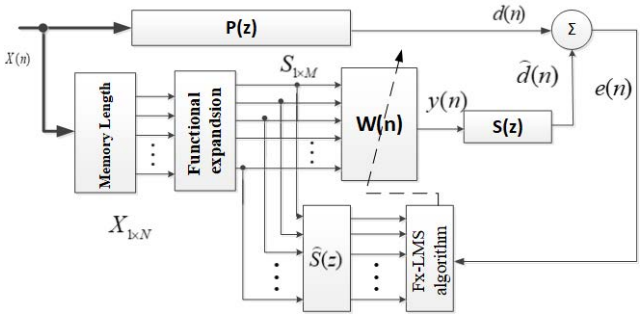


**FIGURE 1.** Nonlinear ANC system based on LIP controller using Fx-LMS algorithm.

of the controller; $P(z)$ is the transfer function of the primary path from the reference source to the error sensor; $S(z)$ is the transfer function of the secondary path from the controller output to the error sensor; $\hat{S}(z)$ is an estimate of the secondary path; $d(n)$ is the main path noise measured at the noise cancellation point; $\hat{d}(n)$ is the estimate of $d(n)$; and $\boldsymbol{W}(n)$ is the filter coefficient of the controller.

Suppose the external input signal vector $\boldsymbol{X}(n)$ at time instant $n$ is

$$\boldsymbol{X}(n) = [x(n), x(n-1), \ldots, x(n-N+1)]^T, \tag{1}$$

where $N$ is the memory length.

The external input signal vector $\boldsymbol{X}(n)$ can be expanded to the vector $\boldsymbol{S}(n) = [[\varphi_1\{\boldsymbol{X}(n)\}]^T, [\varphi_2\{\boldsymbol{X}(n)\}]^T, \ldots, [\varphi_k\{\boldsymbol{X}(n)\}]^T]^T$ based on subclasses of LIP nonlinear filters such as FLANN, Volterra, EMF, where $\varphi_k\{\boldsymbol{X}(n)\}$ is a set of linearly independent functions $k^{th}$ of $\boldsymbol{X}(n)$. In other words, the nonlinear state is generated based on these linearly independent functions. Hence, we can write the output of the LIP nonlinear controller as

$$y(n) = \boldsymbol{W}^T(n)\boldsymbol{S}(n), \tag{2}$$

where $\boldsymbol{W}(n)$ denotes the filter coefficient vector.

The trigonometric expansion function based on FLANN has been described in works [5], which provides a compact representation for the nonlinear filter. Its linearly independent functions are expressed as follows:

$$\varphi_1\{\boldsymbol{X}(n)\} = [x(n), x(n-1), \cdots, x(n-N+1)], \tag{3}$$

$$\varphi_2\{\boldsymbol{X}(n)\} = [sin(\pi x(n)), sin(\pi x(n-1)), \cdots, sin(\pi x(n-N+1))], \tag{4}$$

$$\varphi_3\{\boldsymbol{X}(n)\} = [cos(\pi x(n)), cos(\pi x(n-1)), \cdots, cos(\pi x(n-N+1))], \tag{5}$$

$$\vdots$$

$$\varphi_{2P}\{\boldsymbol{X}(n)\} = [sin(2P\pi x(n)), sin(2P\pi x(n-1)), \cdots, sin(2P\pi x(n-N+1))], \tag{6}$$

$$\varphi_{2P+1}\{\boldsymbol{X}(n)\} = [cos((2P+1)\pi x(n)), cos((2P+1)\pi x(n-1)), \cdots, cos((2P+1)\pi x(n-N+1))], \tag{7}$$

where $P$ is the order of the expansion function FLANN, and $M = N(2P+1)$ is the memory length of the expansion function.

For expansion function using a truncated Volterra series, linearly independent functions can be represented as [4], $\varphi_k\{\boldsymbol{X}(n)\} = x(n-n_{k,1})x(n-n_{k,2})\ldots x(n-n_{k,P_k})$, where $n_{k,1}, n_{k,2}, \ldots, n_{k,P}$ denote an arbitrary integer $(0, 1, 2, \ldots)$ representing delay (note that the maximum value of the delay is $N$, which is the memory size of the input vector $\boldsymbol{X}(n)$), and $P$ is the order of the nonlinear expansion function. As shown in [4], the memory length of the expansion function based on the truncated Volterra series is defined as $M = C^N_{(N+P)} = \frac{(N+P)!}{N!P!}$.

## III. THE ALGORITHM BASED ON SET MEMBERSHIP FILTERING (SMF) FOR NONLINEAR ANC SYSTEM

Many works in [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], and [29] have pointed out the effectiveness of adaptive algorithms based on the SMF technique in reducing computational complexity and the effect of impulse noise. However, only a few works [28], [29] focus on solving for the ANC framework, and they also only apply to linear systems. To overcome the problem of nonlinearity in ANC systems, LIP controllers (Volterra, FLANN…) have been used. But this remarkably increases the computational complexity. In this section, we propose filtered-x SM with a time-varying error bound (Fx-SM-VEB) algorithm for the nonlinear ANC system to reduce the computational burden. In addition, we introduce a robust error bound that uses an impulsive-free estimation, to develop a robust Fx-SM-VEB (RFx-SM-VEB) algorithm for nonlinear ANC systems containing impulsive noise.

### A. PROPOSED FX-SM-VEB ALGORITHM

Based on the least–perturbation property, the constrained cost function of the algorithm is represented as follows:

$$\min_{\boldsymbol{W}(n+1)} \|\boldsymbol{W}(n+1)-\boldsymbol{W}(n)\|^2 \quad \text{Subject to } d(n)-\hat{d}(n)=\xi, \tag{8}$$

where $\xi$ denotes the pre-specified error bound and must be chosen appropriately.

The constraint problem can be solved by using the Lagrange multipliers method, and an unconstrained cost function is given by

$$\mathcal{L} = \|\boldsymbol{W}(n+1)-\boldsymbol{W}(n)\|^2 + \lambda(d(n)-\hat{d}(n)-\xi) = 0, \tag{9}$$

where $\lambda$ is a Lagrange multiplier. Taking the gradient with respect to $\boldsymbol{W}(n+1)$, then setting it equals to zero, we obtain

$$\begin{aligned}\nabla_{\boldsymbol{W}(n+1)} &= \mathcal{L}(\boldsymbol{W}(n+1),\lambda)\\ &= 2(\boldsymbol{W}(n+1)-\boldsymbol{W}(n)) - \lambda\frac{\partial\hat{d}(n)}{\partial\boldsymbol{W}(n+1)} = 0.\end{aligned} \tag{10}$$

To achieve a structure for both nonlinear and linear secondary paths, the virtual secondary path filter [8] is used by

$$\begin{aligned}\tilde{\boldsymbol{H}}(n) &= [h(n,0), h(n,1), \cdots, h(n,K)]^T\\ &= [\frac{\partial\hat{d}(n)}{\partial y(n)}, \frac{\partial\hat{d}(n)}{\partial y(n-1)}, \cdots, \frac{\partial\hat{d}(n)}{\partial y(n-K)}]^T,\end{aligned} \tag{11}$$

where $K$ is the memory size of the secondary path. Note that the term $\frac{\partial\hat{d}(n)}{\partial\boldsymbol{W}(n+1)}$ in (10) can be written as

$$\frac{\partial\hat{d}(n)}{\partial\boldsymbol{W}(n+1)} = \sum_{k=0}^{K}\frac{\partial\hat{d}(n)}{\partial y(n-k)}\frac{\partial y(n-k)}{\partial\boldsymbol{W}(n+1)}. \tag{12}$$

Assuming that the step size is small (i.e., the weight taps are slowly varying), applying the approximation principle, we get

$$\frac{\partial y(n-k)}{\partial\boldsymbol{W}(n+1)} \approx \boldsymbol{S}(n-k), \tag{13}$$

where $\boldsymbol{S}(n)$ is the expanded version of the external input signal $\boldsymbol{X}(n)$ based on the LIP nonlinear functions (FLANN, Volterra,…).

Substituting (13), (12) into (10), and combining with (11), we have

$$\boldsymbol{W}(n+1) = \boldsymbol{W}(n) + \frac{\lambda}{2}\sum_{k=0}^{K}h(n,k)\boldsymbol{S}(n-k). \tag{14}$$

It is worth noting that the term $\sum_{k=0}^{K}h(n,k)\boldsymbol{S}(n-k)$ is the result of filtering the extended signal $\boldsymbol{S}(n)$ through the virtual secondary path filter. Thus, we can define the filtered signal of $\boldsymbol{S}(n)$ as

$$\boldsymbol{S}_f(n) = \boldsymbol{S}(n) * \tilde{\boldsymbol{H}}(n) = \sum_{k=0}^{K}h(n,k)\boldsymbol{S}(n-k), \tag{15}$$

where the asterisk * indicates convolution. Combining (15) and (14), we get

$$\boldsymbol{W}(n+1) = \boldsymbol{W}(n) + \frac{\lambda}{2}\boldsymbol{S}_f(n). \tag{16}$$

Taking the gradient with respect to $\lambda$, then setting it equals to zero, we obtain

$$\nabla_{\lambda} = \mathcal{L}(\boldsymbol{W}(n+1),\lambda) = d(n)-\hat{d}(n)-\xi = 0. \tag{17}$$

To compute (17), we can rewrite the estimate of $d(n)$ as

$$\hat{d}(n) = \tilde{\boldsymbol{H}}(n) * [\boldsymbol{W}(n+1)^T\boldsymbol{S}(n)]. \tag{18}$$

Assuming the step size is small, $\boldsymbol{W}(n+1)$ can be considered as very slowly varying. Thus, (18) can be expressed as

$$\hat{d}(n) \approx \boldsymbol{W}(n+1)^T[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)]. \tag{19}$$

Substituting (15) and (16) into (19), we have

$$\begin{aligned}&\hat{d}(n)\\ &\approx \{\boldsymbol{W}(n) + \frac{\lambda}{2}[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)]\}^T[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)]\\ &= \boldsymbol{W}(n)^T[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)] + \frac{\lambda}{2}[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)]^T[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)]\end{aligned}$$

$$= \boldsymbol{W}(n)^T[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)] + \frac{\lambda}{2}\boldsymbol{S}_f(n)^T\boldsymbol{S}_f(n)$$
$$\approx \tilde{\boldsymbol{H}}(n) * [\boldsymbol{W}(n)^T\boldsymbol{S}(n)] + \frac{\lambda}{2}\boldsymbol{S}_f(n)^T\boldsymbol{S}_f(n). \quad (20)$$

Substituting (20) in to (17), we get

$$d(n) - \tilde{\boldsymbol{H}}(n) * [\boldsymbol{W}(n)^T\boldsymbol{S}(n)] - \frac{\lambda}{2}\boldsymbol{S}_f(n)^T\boldsymbol{S}_f(n) - \xi = 0. \quad (21)$$

Note that

$$e(n) = d(n) - \tilde{\boldsymbol{H}}(n) * [\boldsymbol{W}(n)^T\boldsymbol{S}(n)] \quad (22)$$

and therefore, we have

$$\frac{\lambda}{2} = \frac{e(n) - \xi}{\boldsymbol{S}_f(n)^T\boldsymbol{S}_f(n)}. \quad (23)$$

Combining (16) and (23), we get the equation to update the weights as

$$\boldsymbol{W}(n+1) = \boldsymbol{W}(n) + \psi(n)\frac{e(n)}{\boldsymbol{S}_f(n)^T\boldsymbol{S}_f(n)}\boldsymbol{S}_f(n), \quad (24)$$

where

$$\psi(n) = \begin{cases} (1 - \frac{\xi}{|e(n)|}) & \text{if } |e(n)| > \xi, \\ 0 & \text{otherwise} \end{cases} \quad (25)$$

is a time-varying step-size. At the time instant $n$, if $|e(n)|$ is larger than a pre-specified error bound $\xi$, the equation (24) needs updating the filtering weights. Otherwise, there is no need to update. In other words, the algorithm used a data-selection update mechanism to reduce the computational complexity. We hereafter call this algorithm the filtered-x set membership (Fx-SM) algorithm.

The algorithm achieves good convergence performance and low computational complexity when a pre-specified error bound $\xi$ is chosen appropriately. However, this is difficult to achieve since we lack understanding of the environment of the nonlinearity that exist in the primary path and/or secondary path, the reference input signal. In order to tackle this disadvantage, a time-varying error bound (VEB) has been developed, which can automatically adjust the error bound in each iteration during the update process.

Referenced by works [25], [30], a time-varying error bound can be represented as follows:

$$\xi(n) = \begin{cases} |e(n)| - \rho\theta(n) & \text{if } |e(n)| > \theta(n), \\ \xi & \text{otherwise}, \end{cases} \quad (26)$$

where $\xi = \sqrt{\tau\delta_v^2}$ is a pre-specified error bound, $\delta_v^2$ is the variance of observation noise $v(n)$, $\rho$ is usually choose $0 < \rho \ll 1$ to achieve low steady-state misalignments, $\theta(n) = \Psi\delta(n)$ is a threshold parameter, with $\Psi$ being a scalar depending on the nonlinear model of the system; $\delta(n)$ calculated through a residual error estimate as follows:

$$\delta^2(n+1) = \omega\delta^2(n) + (1-\omega)e^2(n), \quad (27)$$

where $0 < \omega \leqslant 1$. Therefore, when $|e(n)| > \theta(n)$ then the error bound $\xi(n)$ changes according to residual error, otherwise it is equal to a pre-specified error bound. To increase convergence, it is common to set the initial $\delta^2(0)$ to large (i.e., $|e(n)|$ will be less than $\theta(n)$ during transient states), and thus the algorithm will work with a pre-specified error bound $\xi(n) = \xi$. Fig. 2 illustrates the nonlinear ANC system based on the proposed algorithm.

**Algorithm 1** Fx-SM-VEB Algorithm

**Parameters:** $\rho$; $\tau$; $\omega$; $\Psi$
**Initialization:** $\boldsymbol{W}(0) = 0$; $\delta^2(0) = 1$

**for** $n = 1$ to $N$ **do**
  $y(n) = \boldsymbol{W}^T(n)\boldsymbol{S}(n)$
  $\hat{d}(n) = y(n)*\tilde{\boldsymbol{H}}(n)$
  $e(n) = d(n) - \hat{d}(n)$
  $\boldsymbol{S}_f(n) = \boldsymbol{S}(n) * \tilde{\boldsymbol{H}}(n)$
  $\delta^2(n+1) = \omega\delta^2(n)+(1-\omega)e^2(n)$
  $\theta(n) = \Psi\delta(n)$
  **if** $|e(n)| > \theta(n)$ **then**
    $\xi(n) = |e(n)| - \rho\theta(n)$
  **else**
    $\xi(n) = \sqrt{\tau\delta_v^2}$
  **end if**
  **if** $|e(n)| > \xi(n))$ **then**
    $\boldsymbol{W}(n+1) = \boldsymbol{W}(n)+\frac{1}{(\delta+\|\boldsymbol{S}_f(n)\|^2)}(1-\frac{\xi(n)}{|e(n)|})\boldsymbol{S}_f(n)e(n)$
  **else**
    $\boldsymbol{W}(n+1) = \boldsymbol{W}(n)$
  **end if**
**end for**

### B. ROBURST FX-SM-VEB (RFX-SM-VEB) ALGORITHM

A serious challenge is that the adaptive Fx-SM-VEB algorithm becomes unstable when impulsive noise is present in the reference input signal. To address this, we extend the residual noise estimate $\delta^2(n)$ in (27) to a robust estimate for impulse noise as

$$\delta^2(n+1) = \omega\delta^2(n) + (1-\omega)min(\boldsymbol{\phi}(n)), \quad (28)$$

with $\boldsymbol{\phi}(n) = [e^2(n), e^2(n-1), \ldots, e^2(n-\varpi+1)]^T$, and $\varpi$ is length of the estimation window.

It is easy to see that the estimate $\delta^2(n)$ is robust for outliers, (i.e., when the impulsive noise-corrupted error signals, then $\theta(n) = \Psi\delta(n)$ would not increase). Hence, if the ANC system contains impulse noise, it is certain that $|e(n)| > \theta(n)$, and from (26) we use the time-varying error bound $\xi(n) = |e(n)| - \rho\theta(n)$. However, due to the impulse noise is very large, $\rho\theta(n) \ll |e(n)|$, and thus we can deduce the error bound $\xi(n) \approx |e(n)|$. In this case, the step-size in (25) equals to zero, resulting in the RFx-SM-VEB algorithm stopping the update process. This also means that the RFx-SM-VEB algorithm will not be unstable in the presence of the impulse noise.

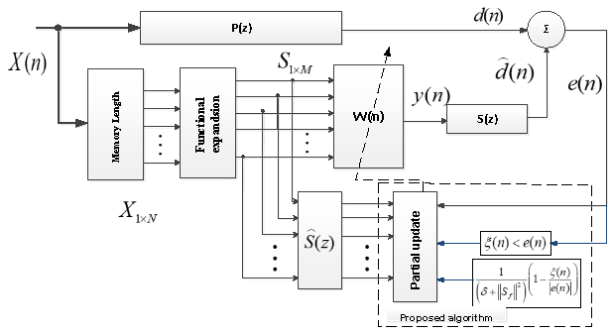


**FIGURE 2.** Nonlinear ANC system based on proposed algorithm.

## IV. ANALYSIS OF THE ALGORITHM

### A. THE STEADY-STATE BEHAVIOR ANALYSIS

In this section, analysis of the algorithm based on the SMF technique for the ANC system have been presented.

From (22) we can represent the residual error in the following form:

$$e(n) = d(n) - \boldsymbol{W}^T(n)[\tilde{\boldsymbol{H}}(n) * \boldsymbol{S}(n)] = d(n) - \boldsymbol{W}^T(n)\boldsymbol{S}_f(n), \tag{29}$$

where $\boldsymbol{S}_f(n)$ is the filtered version of the extended signal $\boldsymbol{S}(n)$ via the secondary path.

We can define the estimation error vector as $\triangle\boldsymbol{W}(n) = \boldsymbol{W}_0 - \boldsymbol{W}(n)$, where $\boldsymbol{W}_0$ is the weight vector that needs to be estimated, then the primary signal $d(n)$ at the noise cancellation point can be represented as

$$d(n) = \boldsymbol{W}_0^T\boldsymbol{S}_f(n) + \upsilon(n), \tag{30}$$

where $\upsilon(n)$ denotes noise measured at the noise cancellation point, which is assumed to be a zero-mean Gaussian process and have variance $\delta_\upsilon^2$.

Substituting (30) into (29), we get

$$e(n) = \upsilon(n) + \triangle\boldsymbol{W}^T(n)\boldsymbol{S}_f(n). \tag{31}$$

Thus, we can represent the output MSE of $e(n)$ as follows:

$$\begin{aligned}
&\zeta(n)\\
&= E[\|e(n)\|^2]\\
&= E\{[\upsilon(n) + \triangle\boldsymbol{W}^T(n)\boldsymbol{S}_f(n)]^T[\upsilon(n) + \triangle\boldsymbol{W}^T(n)\boldsymbol{S}_f(n)]\}\\
&= E[\|\upsilon(n)\|^2] + E\{[\triangle\boldsymbol{W}^T(n)\boldsymbol{S}_f(n)]^T[\triangle\boldsymbol{W}^T(n)\boldsymbol{S}_f(n)]\}\\
&= \delta_\upsilon^2 + E[\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}^T(n)\boldsymbol{S}_f(n)]\\
&= \delta_\upsilon^2 + trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}^T(n)]\}.
\end{aligned} \tag{32}$$

From (32), the excess MSE can be deduced as

$$\zeta_{ex}(n) = trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}^T(n)]\}. \tag{33}$$

On the other hand, the weight update equation of the SMF-based system can be rewritten by

$$\boldsymbol{W}(n+1) = \boldsymbol{W}(n) + \frac{1}{N\delta_s^2}\left(1 - \frac{\xi(n)}{|e_0(n)|}\right)e(n)\boldsymbol{S}_f(n), \tag{34}$$

**Algorithm 2** RFx-SM-VEB Algorithm

**Parameters:** $\rho$; $\tau$; $\omega$; $\Psi$; $\varpi$;
**Initialization:** $\boldsymbol{W}(0) = 0$; $\delta^2(0) = 1$

**for** $n = 1$ to $N$ **do**
  $y(n) = \boldsymbol{W}^T(n)\boldsymbol{S}(n)$
  $\hat{d}(n) = y(n)*\tilde{\boldsymbol{H}}(n)$
  $e(n) = d(n) - \hat{d}(n)$
  $\boldsymbol{S}_f(n) = \boldsymbol{S}(n) * \tilde{\boldsymbol{H}}(n)$
  $\boldsymbol{\phi}(n) = [e^2(n), e^2(n-1), \ldots, e^2(n-\varpi+1)]^T$
  $\delta^2(n+1) = \omega\delta^2(n) + (1-\omega)min(\boldsymbol{\phi}(n))$
  $\theta(n) = \Psi\delta(n)$
  **if** $|e(n)| > \theta(n)$ **then**
    $\xi(n) = |e(n)| - \rho\theta(n)$
  **else**
    $\xi(n) = \sqrt{\tau\delta_v^2}$
  **end if**
  **if** $|e(n)| > \xi(n))$ **then**
    $\boldsymbol{W}(n+1) = \boldsymbol{W}(n) + \frac{1}{(\delta+\|\boldsymbol{S}_f(n)\|^2)}(1 - \frac{\xi(n)}{|e(n)|})\boldsymbol{S}_f(n)e(n)$
  **else**
    $\boldsymbol{W}(n+1) = \boldsymbol{W}(n)$
  **end if**
**end for**

with

$$|e_0(n)| = \begin{cases} |e(n)| & \text{if } |e(n)| > \xi(n) \\ \xi(n) & \text{otherwise,} \end{cases} \tag{35}$$

and $\delta_s^2$ is the variance of the filtered input signal through the secondary path $\boldsymbol{S}_f(n)$.

Therefore, the updated equation of the estimation error $\triangle\boldsymbol{W}(n)$ is given by

$$\begin{aligned}
&\triangle\boldsymbol{W}(n+1)\\
&= \triangle\boldsymbol{W}(n) - \frac{1}{N\delta_s^2}\left(1 - \frac{\xi(n)}{|e_0(n)|}\right)e(n)\boldsymbol{S}_f(n)\\
&= \triangle\boldsymbol{W}(n) - \frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n) + \frac{1}{N\delta_s^2}\frac{\xi(n)}{|e_0(n)|}e(n)\boldsymbol{S}_f(n).
\end{aligned} \tag{36}$$

Substituting (36) into (33), the output excess MSE can be expressed by

$$\begin{aligned}
&\zeta_{ex}(n+1)\\
&= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n+1)\triangle\boldsymbol{W}^T(n+1)]\}\\
&= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)(\boldsymbol{U}+\boldsymbol{V})(\boldsymbol{U}+\boldsymbol{V})^T]\}\\
&= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)(\boldsymbol{U}\boldsymbol{U}^T + \boldsymbol{U}\boldsymbol{V}^T\\
&\quad + \boldsymbol{V}\boldsymbol{U}^T + \boldsymbol{V}\boldsymbol{V}^T)]\},
\end{aligned} \tag{37}$$

with $\boldsymbol{U} = \triangle\boldsymbol{W}(n) - \frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n)$, and $\boldsymbol{V} = \frac{1}{N\delta_s^2}\frac{\xi(n)}{|e_0(n)|}e(n)\boldsymbol{S}_f(n)$. By using calculation given in Appendix, we derive the output excess MSE in (37) as

$$\zeta_{ex}(n+1) = \delta_\upsilon^2 + 2\xi(n)E\left(\frac{1}{|e_0(n)|}\right)\zeta_{ex}(n)$$

$$-2\xi(n)E\left(\frac{e^2(n)}{|e_0(n)|}\right)+\xi^2(n)E\left(\frac{e^2(n)}{|e_0(n)|^2}\right). \tag{38}$$

Let $P_u$ be the probability of updating in each iteration, $E(|)$ the conditional expected value, we can write

$$\begin{aligned}E\left(\frac{1}{|e_0(n)|}\right)&=E\left(\frac{1}{|e(n)|}\Big||e(n)|>\xi(n)\right)P_u\\&\quad+\frac{1}{\xi(n)}(1-P_u)\\&=AP_u+\frac{1}{\xi(n)}(1-P_u),\end{aligned} \tag{39}$$

$$\begin{aligned}E\left(\frac{e^2(n)}{|e_0(n)|}\right)&=E(e(n)\big||e(n)|>\xi(n))P_u\\&\quad+\frac{1}{\xi(n)}E(e^2(n)\big||e(n)|\leqslant\xi(n))(1-P_u)\\&=BP_u+\frac{1}{\xi(n)}C(1-P_u),\end{aligned} \tag{40}$$

$$\begin{aligned}E\left(\frac{e^2(n)}{|e_0(n)|^2}\right)&=P_u+\frac{1}{\xi^2(n)}E(e^2(n)\Big||e(n)|\leqslant\xi(n))(1-P_u)\\&=P_u+\frac{1}{\xi^2(n)}C(1-P_u),\end{aligned} \tag{41}$$

where $A=E(\frac{1}{|e(n)|}\big||e(n)|>\xi(n))$, $B=E(e(n)\big||e(n)|>\xi(n))$, and $C=E(e^2(n)\big||e(n)|\leqslant\xi(n))$. Substituting (39-41) in (38) we have

$$\begin{aligned}\zeta_{ex}(n+1)&=\delta_v^2+2\xi(n)[AP_u+\frac{1}{\xi(n)}(1-P_u)]\zeta_{ex}(n)\\&\quad-2\xi(n)[BP_u+\frac{1}{\xi(n)}C(1-P_u)]+\xi^2(n)[P_u\\&\quad+\frac{1}{\xi^2(n)}C(1-P_u)].\end{aligned} \tag{42}$$

When the algorithm reaches the steady state (i.e., $\xi_{ex}(n+1)\rightarrow\xi_{ex}(n)$, the output excess MSE becomes (43), as shown at the bottom of the next page.

### B. THE CONVERGENCE AND STABILITY CONDITIONS ANALYSIS

To establish the convergent behavior of the proposed algorithms, we conduct the behavior analysis of the weight deviation vector ($\triangle\boldsymbol{W}(n)=\boldsymbol{W}_0$-$\boldsymbol{W}(n)$). The weighted deviation vector update equation for algorithms (24) and (25) can be expressed as,

$$\triangle\boldsymbol{W}(n+1)=\triangle\boldsymbol{W}(n)-\psi(n)\frac{e(n)\boldsymbol{S}_f(n)}{\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)}, \tag{44}$$

Substituting (31) into (44), the above expression can be rewritten as,

$$\triangle\boldsymbol{W}(n+1)=\left(\boldsymbol{I}-\psi(n)\frac{\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)}{\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)}\right)\triangle\boldsymbol{W}(n), \tag{45}$$

where $\boldsymbol{I}$ is the 1 by $M$ identity vector.

The expected coefficient vector error is given by,

$$\begin{aligned}&E[\triangle\boldsymbol{W}(n+1)]\\&=\left[\boldsymbol{I}-\psi(n)E\frac{\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)}{\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)}\right]E[\triangle\boldsymbol{W}(n)]\\&\cong\left[\boldsymbol{I}-\psi(n)\frac{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]}{E[\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)]}\right]E[\triangle\boldsymbol{W}(n)]\\&=\left[\boldsymbol{I}-\psi(n)\frac{\boldsymbol{R}}{Tr[\boldsymbol{R}]}\right]E[\triangle\boldsymbol{W}(n)],\end{aligned} \tag{46}$$

where $\boldsymbol{R}$ is the correlation matrix of the signal $\boldsymbol{S}_f(n)$. Equation (46) can be expressed as,

$$E[\triangle\boldsymbol{W}(n+1)]=\left[\boldsymbol{I}-\psi(n)\frac{\boldsymbol{R}}{Tr[\boldsymbol{R}]}\right]^{(n+1)}E[\triangle\boldsymbol{W}(0)]. \tag{47}$$

Based on the method of analyzing the convergence behavior of the NLMS algorithm [31], we deduce the stability condition of the proposed algorithm as,

$$0<\psi(n)<2. \tag{48}$$

On the other hand, from (25) we have $0<\psi(n)<1$. Therefore, it can be concluded that the proposed algorithms are always stable.

## V. ANALYSIS OF COMPUTATIONAL COMPLEXITY

Table 1 shows the comparison of computational complexity in each iteration in terms of multiplication and addition for Fx-LMS, Fx-SM, and proposed Fx-SM-VEB algorithms. For the proposed Fx-SM-VEB algorithm, the basic operations are analyzed specifically as follows:

- Computing the filter output in the equation (2) requires $M$ multiplication $M-1$ addition, where $M$ is the memory length of the extended input signal. For the Volterra expansion function $M=C_{N+P}^N=\frac{(N+P)!}{N!P!}$, for the FLANN expansion function $M=N(2P+1)$, where $P$ is the order of the expansion function, $N$ is the memory length of the external input signal.
- Computing the term $\|\boldsymbol{S}_f(n)\|_2^2$ in the update equation (24) requires one multiplication and two additions. (Refer to [32], we can decompose $\|\boldsymbol{S}_f(n)\|_2^2$ into $\|\boldsymbol{S}_f(n-1)\|_2^2+\boldsymbol{S}_f^2(n)-\boldsymbol{S}_f^2(n-N)$. Notice that the terms $\|\boldsymbol{S}_f(n-1)\|_2^2$ and $\boldsymbol{S}_f^2(n-N)$ have been computed from earlier iterations, and therefore we only need to compute for term $\boldsymbol{S}_f^2(n)$).
- Computing the time-varying error bound $\xi(n)$ in equations (26,27) requires 4 multiplications, 2 additions, and 1 comparison.
- Computing the weight update $\boldsymbol{W}(n)$ in the equation (24) requires $P_uM+1$ multiplication and $P_u(M-1)$ addition, and 1 comparison, where $P_u$=UR/100 is the probability of updating the filter weight in each iteration, and UR is the update rate which is defined by the percentage of update numbers of the equation (24) per the total iterations.

**TABLE 1. Computational complexity in each iteration of the algorithms.**

| Algorithms | Multiplication | Addition |
|---|---|---|
| Fx-LMS | $2M + KL + 1$ | $2M + K(L-1) - 2$ |
| Fx-SM | $M + PuM + KL + 1$ | $M + Pu(M-1) + K(L-1)$ |
| Fx-SM-VEB | $M + PuM + KL + 6$ | $M + Pu(M-1) + K(L-1) + 3$ |

- Computing the filtering of the extended signal $\boldsymbol{S}(n)$ via the virtual secondary path (15) requires $KL$ multiplication and $K(L-1)$ addition, where $K$ is the memory length of $S(z)$ and $L$ is the number of channels of extended signal. Note that the extended signal $\boldsymbol{S}(n)$ can be analyzed in a multichannel fashion to exploit the shift structure in each channel. Instead of having to compute for all the operations of the extended signal filtering, we only need to compute the first term in each channel and then use the shift property to derive the remaining terms.

## VI. SIMULATION

In this section, we apply the proposed algorithms for two nonlinear controllers commonly used in ANC systems, namely FLANN and Volterra, to evaluate the effectiveness in terms of reducing computational complexity and impulsive noise. First, we illustrate the comparison between the conventional Fx-LMS algorithm, Fx-SM algorithm, and the proposed Fx-SM-VEB algorithm for nonlinear controllers in the Gaussian noise environment. Then, we compare the proposed RFx-SM-VEB algorithm with previous robust algorithms (such as Fx-LogLMS [33], RFx-LMS [34], and Fx-LogLMP [35]) in an impulsive noise environment.

To compare the performance in the Gaussian noise environment, we use the normalized mean-square error (NMSE) defined by

$$NMSE = 10\log_{10}\{\frac{E(e^2(n))}{\delta_{dn}^2}\}, \tag{49}$$

where $\delta_{dn}^2$ is the variance of the primary path noise and $e^2(n)$ is the residual noise power.

To compare the performance in an impulsive noise environment, we use the averaged noise reduction (ANR) defined by

$$ANR(n) = 20\log_{10}\{\frac{A_e(n))}{A_d(n))}\}, \tag{50}$$

where $A_d(n) = \Omega A_d(n-1) + (1-\Omega)|d(n)|$ and $A_e(n) = \Omega A_e(n-1) + (1-\Omega)|e(n)|$. Note that $\Omega$ is very close to 1, $A_d(0) = 0$, and $A_e(0) = 0$.

In all experiments, the controller parameters have chosen as follows: (*i*) For Volterra, memory size $N = 10$ and order of $P = 2$; (*ii*) For FLANN, memory size $N = 10$ and order of $P = 3$; (*iii*) Signal-to-noise ratio $SNR = 30dB$; and (*iv*) Learning curves were plotted after 100 independent runs to better discern the behavior of the NMSE or ANR.

### A. PERFORMANCE COMPARISON IN THE GAUSSIAN NOISE ENVIRONMENT

*Experiment 1:* In this experiment, we assume the scenario of nonlinearity in the ANC system as referenced in [8]. Here, the nonlinear models of the primary path and the secondary path are expressed, respectively, by

$$\begin{aligned} d(n) = {} & x(n) + 0.8x(n-1) + 0.3x(n-2) + 0.4x(n-3) \\ & - 0.8x(n)x(n-1) + 0.9x(n)x(n-2) \\ & + 0.7x(n)x(n-3), \end{aligned} \tag{51}$$

and

$$\begin{aligned} \hat{d}(n) = {} & y(n) + 0.35y(n-1) + 0.09y(n-2) \\ & - 0.5y(n)y(n-1) + 0.4y(n)y(n-2). \end{aligned} \tag{52}$$

The reference input signal is a white Gaussian noise process.

#### 1) + CHOICE OF PARAMETERS

To determine the suitable parameters for the proposed algorithm, we conducted many experiments with different values for each parameter. The values of the parameters are chosen based on a trade-off between the NMSE performance and computational complexity (calculated through the update rate UR). In this section, we only illustrate the experiments for choosing the $\Psi$ and $\tau$ parameters for the Fx-SM-VEB algorithm using the Volterra controller.

To determine the parameter $\Psi$, we first keep the $\tau$ fixed, then change the $\Psi$ in turn to get the best compromise NMSE and UR values (and vice versa for parameter $\tau$). Fig. 3a illustrates the NMSE performance and the UR with different $\Psi$ values. Fig. 3b illustrates the NMSE performance and the UR with different $\tau$ values. From Fig. 3a and 3b, it is easy to see that the parameters $\Psi = 1.98$ and $\tau = 11$ achieves a good compromise between the NMSE performance and the UR update rate.

#### 2) + COMPARING PERFORMANCE

The parameters of the proposed Fx-SM-VEB algorithm are set: $\delta^2(n) = 16$, $\rho = 0.1$, $\omega = 0.99$, $\tau = 100$, $\Psi = 1.38$ for FLANN; and $\delta^2(n) = 1$, $\rho = 0.1$, $\omega = 0.99$, $\tau = 11$,

$$\begin{aligned} \zeta_{ex}(n) &= \frac{\delta_v^2 - 2\xi(n)[BP_u + \frac{1}{\xi(n)}C(1-P_u)] + \xi^2(n)[P_u + \frac{1}{\xi^2(n)}C(1-P_u)]}{1 - 2\xi(n)[AP_u + \frac{1}{\xi(n)}(1-P_u)]} \\ &= \frac{\delta_v^2 - 2\xi(n)BP_u - C(1-P_u) + \xi^2(n)P_u}{2P_u - 2\xi(n)AP_u - 1}. \end{aligned} \tag{43}$$

**TABLE 2. Summarize the update rate, computational complexity per iteration of the algorithms.**

| Controller | Fx-LMS | | | Fx-SM | | | Proposed Fx-SM-VEB | | |
|---|---|---|---|---|---|---|---|---|---|
| | Update rate | Complexity | | Update rate | Complexity | | Update rate | Complexity | |
| | | Multiplication | Addition | | Multiplication | Addition | | Multiplication | Addition |
| FLANN | 100% | 162 | 156 | 56% | 131 | 126 | 15% | 107 | 101 |
| Volterra | 100% | 166 | 160 | 24% | 116 | 112 | 15% | 115 | 109 |

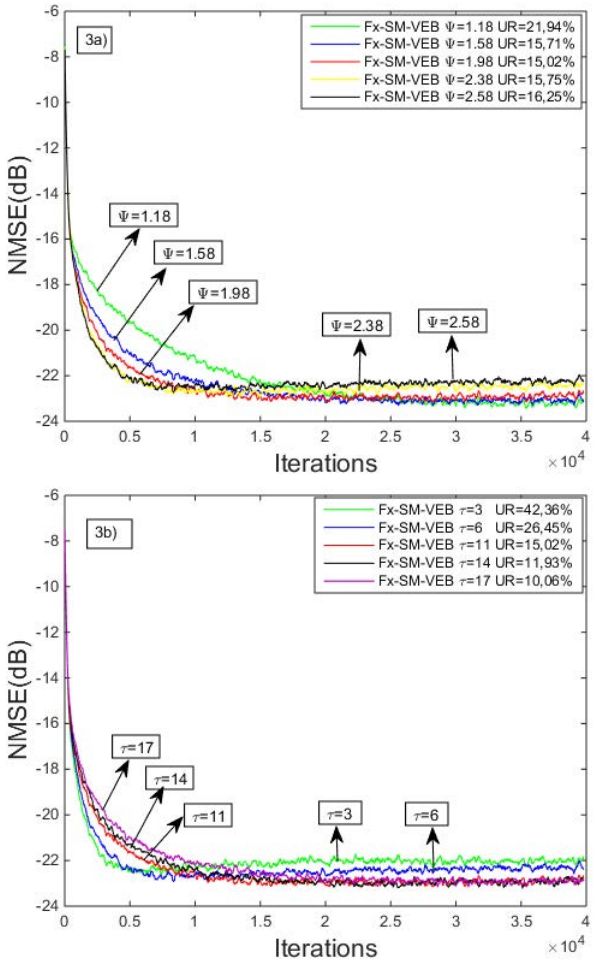


**FIGURE 3. NMSE performance versus the parameter Ψ (3a), and The NMSE performance versus the parameter $\tau$ (3b).**

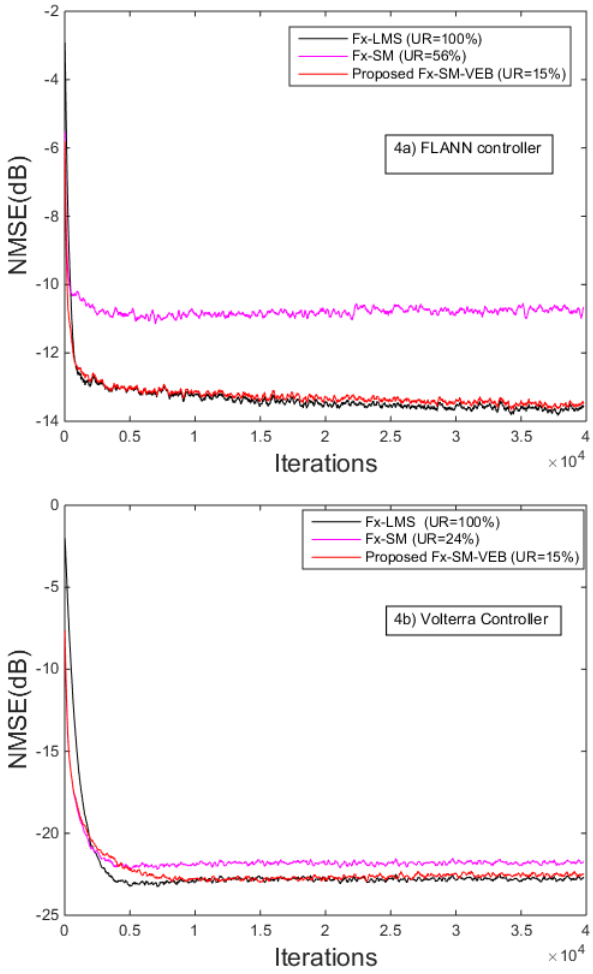


**FIGURE 4. Performance comparison of FLANN (4a), and Voltera (4b) controllers using Fx-LMS, Fx-SM and proposed Fx-SM-REB algorithms in Experiment 1.**

$\Psi = 1.98$ for Volterra. The parameters of Fx-SM are set: $\xi = \sqrt{3\delta_v^2}$ for FLANN; and $\xi = \sqrt{8\delta_v^2}$ for Volterra. Step-size for Fx-NLMS algorithm is set for FLANN by $\mu_1 = 0.003$ and $\mu_2 = 0.003$, and for Volterra by $\mu_1 = 0.04$ and $\mu_2 = 0.06$.

Fig. 4 exhibits the noise cancellation performance of the ANC systems based on the FLANN and Volterra, using the Fx-LMS, Fx-SM, and proposed Fx-SM-VEB algorithms. Table 2 summarizes the update rate, and the computation complexity in each iteration of the corresponding algorithms.

From Table. 2 and Fig. 4, it is easy to see that the proposed Fx-SM-VEB algorithm reduces the computational complexity significantly compared to the Fx-LMS algorithm while maintaining the equivalent noise cancellation performance. The Fx-SM algorithm with a pre-specified error bound is reduced performance compared to the Fx-LMS and proposed Fx-SM-VEB algorithms. This is probably due to the presence of nonlinearity in the primary path and secondary path, making the Fx-SM algorithm difficult to achieve a suitable pre-specified error bound.

*Experiment 2:* In this experiment, we keep the primary path model as in Experiment 1, but the secondary path is approximated by the Hammerstein [10] nonlinear model as follows:

$$u(n) = tanh\{y(n)\}$$
$$\hat{d}(n) = u(n) + 0.2u(n-1) + 0.05u(n-2). \quad (53)$$

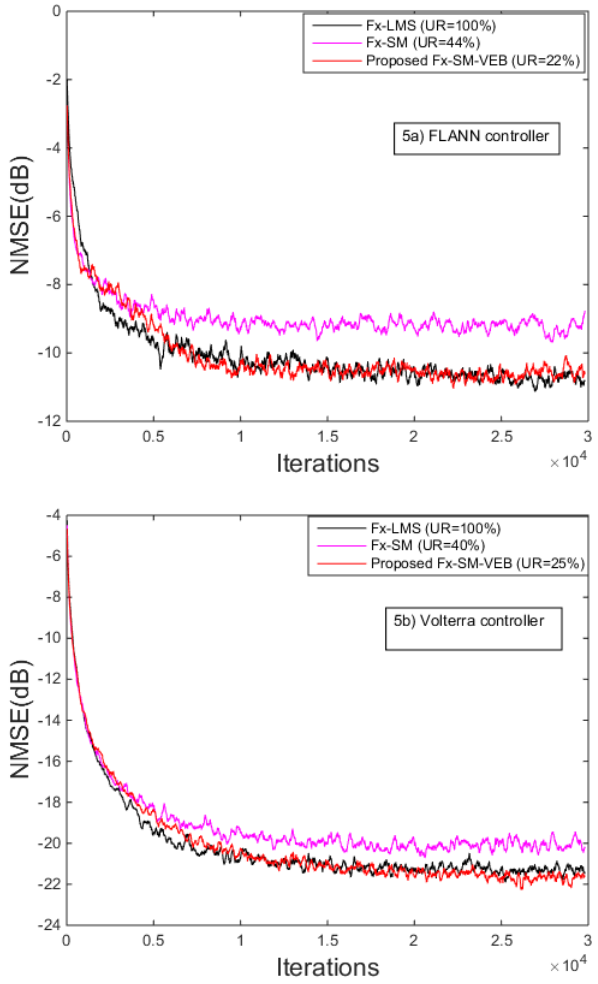


**FIGURE 5.** Performance comparison of FLANN (5a), and Voltera (5b) controllers using Fx-LMS, Fx-SM and proposed Fx-SM-REB algorithms in Experiment 2.

The reference noise source is the color noise generated by filtering the white Gaussian noise $v(n)$ through the ARMA (Autoregressive–moving-average) model as

$$x(n) = 0.04x(n-1) - 0.034x(n-2) + 0.0396x(n-3) - 0.07565x(n-4) - 0.1v(n) - 0.01v(n-1) - 0.137v(n-2) + 0.0353v(n-3) + 0.0698v(n-4). \quad (54)$$

Parameters of the proposed Fx-SM-VEB algorithm are chosen: $\delta^2(0) = 3$, $\rho = 0.1$, $\omega = 0.999$, $\tau = 100$, $\Psi = 1.38$ for FLANN; and $\delta^2(0) = 0.5$, $\rho = 0.1$, $\omega = 0.999$, $\tau = 8$, $\Psi = 1.68$ for Volterra. The parameters of Fx-SM are set: $\xi = \sqrt{10\delta_\upsilon^2}$ for FLANN; and $\xi = \sqrt{5\delta_\upsilon^2}$ for Volterra. For Fx-LMS, the step-size is selected: $\mu_1 = 0.006$ and $\mu_2 = 0.006$ for FLANN; $\mu_1 = 0.025$ and $\mu_2 = 0.025$ for Volterra.

Fig. 5 illustrates the performance of the FLANN and Volterra controllers based on the Fx-LMS, Fx-SM, and proposed Fx-SM-VEB algorithms. From this Figure, it is clear that the proposed Fx-SM-VEB algorithm achieves steady-state performance equivalent to the conventional

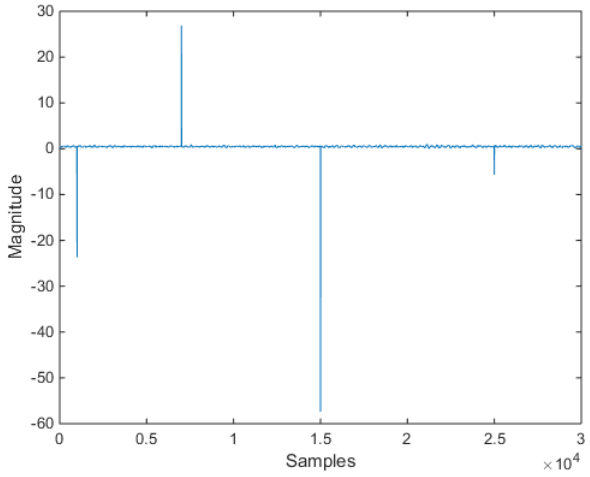


**FIGURE 6.** Reference input signal corrupted by impulsive noise $\mho(n)$, with $\Upsilon = 10000$ and $\Re = 0.8$.

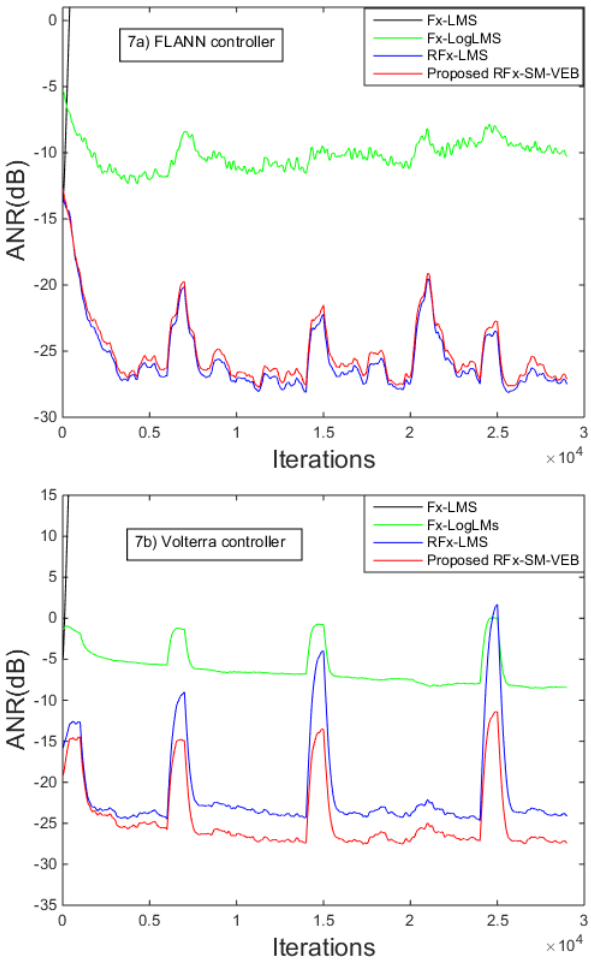


**FIGURE 7.** Depicts the ANR performance of the algorithms on the FLANN (7a), and Volterra (7b) controllers.

Fx-LMS algorithm but with lower computational complexity, thanks to the data-selective update strategy and time-varying error bound. Specifically, FLANN and Volterra-based ANC

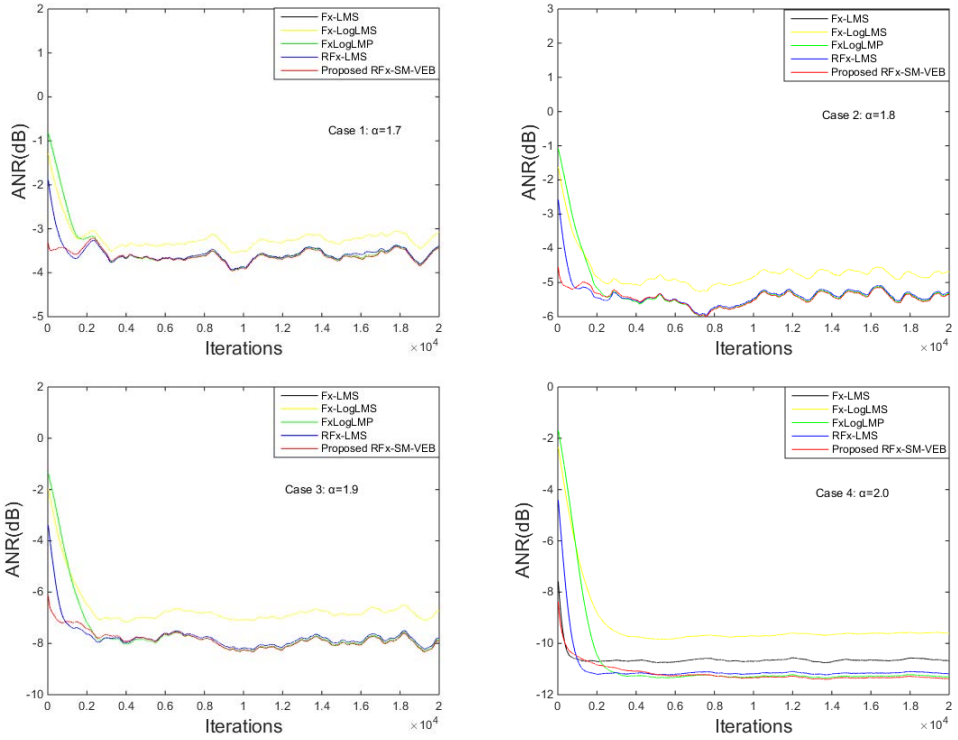


**FIGURE 8.** Comparison of ANR in FLANN based-ANC system with a non-minimum phase secondary path for $S\alpha S$ primary noise: case 1: $\alpha = 1.7$, case 2: $\alpha = 1.8$, case 3: $\alpha = 1.9$ and case 4: $\alpha = 2.0$.

systems using the proposed Fx-SM-VEB algorithm can reduce the number of operations to update the weights by 78% and 75%, respectively. The Fx-SM algorithm using pre-specified error bound suffers from attenuation in nonlinear environments, in particular, the noise cancellation performance is attenuated by 2.5 dB for the FLANN controller, and 2 dB for the Volterra controller.

### B. PERFORMANCE COMPARISON IN THE IMPULSIVE NOISE ENVIRONMENT

*Experiment 3:* In this experiment, we use the reference input as an actual noise generated by the fan. This noise source is recorded at a frequency of 44100 Hz and sampled at 16 bits per sample. The length of the reference signal is equal to 30 000 samples. In order to evaluate the performance of the proposed RFx-SM-VEB algorithm, we add impulsive noise to the reference input signal at iterations 1000, 7000, 15000, and 25000. The impulsive noise is modeled by $\mho(n) = \sigma(n)\Gamma(n)$ [27], where $\sigma(n)$ is a Bernoulli random sequence, independent and identically distributed (i.i.d) and $\sigma(n)$ has an instantaneous value at time $n$ being either zero or one (occurrence probability ($P(\sigma(n) = 1) = \Re$)); and $\Gamma(n)$ is an i.i.d zero-mean Gaussian sequence, with variance $\delta_\Gamma^2 = \Upsilon\delta_s^2$, here $\Upsilon \gg 1$, $\delta_s^2$ is the variance of the reference input signal. Fig. 6 illustrates the reference input signal corrupted by impulsive noise $\mho(n)$, with $\Upsilon = 10000$ and $\Re = 0.8$.

Fig. 7 depicts the ANR learning curves of the algorithms on the FLANN (7a), and Volterra (7b) controllers. It is found that the proposed RFx-SM-VEB algorithm is stable and achieves better convergence performance than the Fx-LMS, and Fx-LogLMS algorithms for both controllers. The Fx-LMS algorithm suffer from diverges, while the RFx-LMS is stable but achieves worse performance than proposed RFx-SM-VEB when applied to the Volterra controller.

*Experiment 4:* Here we use the input reference noise as an impulse noise [36] modeled by a standard symmetric $\alpha$-stable distribution as

$$\Theta(t) = \exp(-|t|^\alpha), \tag{55}$$

where parameter $\alpha$ denotes a characteristics exponent, $0 < \alpha < 2$. When $\alpha$ is close to 2, we can consider this distribution as a Gaussian function. For a small value of $\alpha$, this distribution can be viewed as a function of strong impulse noise. The nonlinearity of the primary path noise at the canceling point is modeled by [15],

$$d(n) = v(n-2) + 0.08v^2(n-2) - 0.04v^3(n-1), \tag{56}$$

where $v(n) = x(n) * p(n)$, with $p(n)$ is the impulse response of the function $P(n) = z^{-3} - 0.3z^{-4} + 0.2z^{-5}$. The secondary path is assumed to be a minimum phase transfer function $S(z) = z^{-2} + 1.5z^{-3} - z^{-4}$.

Four scenarios corresponding to the impulsive noise reference input have been demonstrated in this experiment: scenario 1: $\alpha = 1.7$; scenario 2: $\alpha = 1.8$; scenario 3: $\alpha = 1.9$; scenario 4: $\alpha = 2.0$.

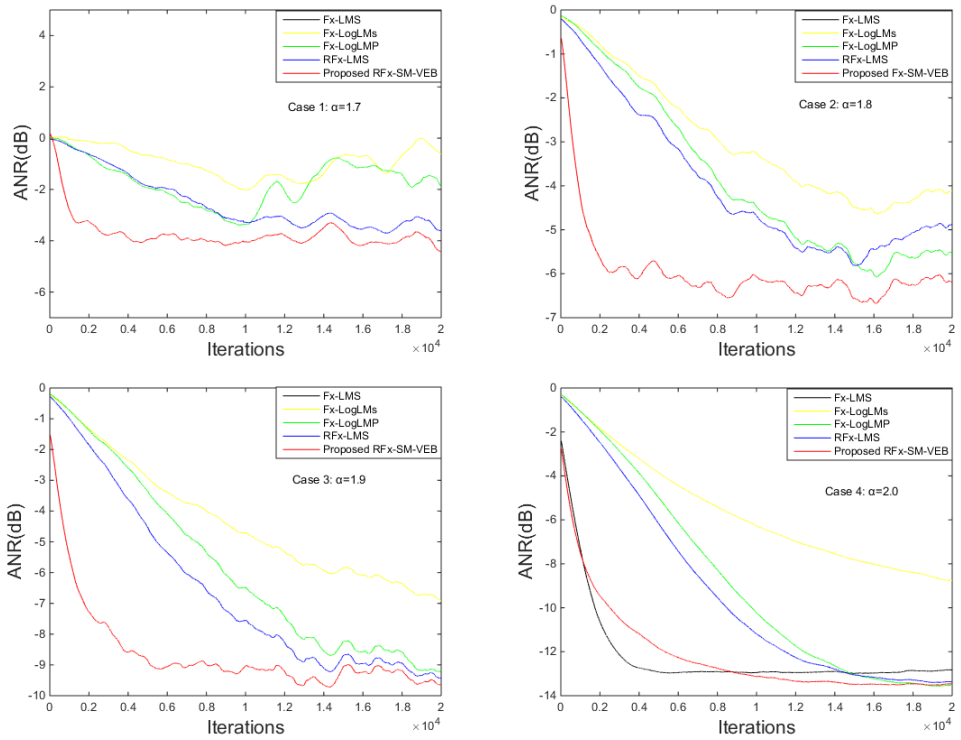


**FIGURE 9.** Comparison of ANR in Volterra based-ANC system with a non-minimum phase secondary path for $S\alpha S$ primary noise: case 1: $\alpha = 1.7$, case 2: $\alpha = 1.8$, case 3: $\alpha = 1.9$ and case 4: $\alpha = 2.0$.

Fig. 8 and Fig. 9 depict the learning curve of the Fx-LMS, Fx-LogLMS, Fx-LogLMP, RFx-LMS, and proposed RFx-SM-VEB algorithms for the Volterra and FLANN based-ANC systems, respectively. As we can see, the Fx-LMS algorithm diverges in the cases $\alpha = 1.7$, $\alpha = 1.8$, and $\alpha = 1.9$ which is obvious since this algorithm is only for assuming the reference noise is a Gaussian process. For the FLANN based-ANC system, the proposed RFx-SM-VEB algorithm is equivalent to RFx-LMS and Fx-LogLMP algorithms, and outperforms the Fx-LogLMS algorithm. Notice that the Fx-LogLMP algorithm needs to know the a priori information of the reference noise. For the Volterra-based ANC system, the proposed RFx-SM-VEB algorithm outperforms all the above algorithms. It is easy to see that this superiority is more obvious in the case of strong impulsive noise.

## VII. CONCLUSION

In this paper, we proposed two Fx-SM-VEB and RFx-SM-VEB algorithms for nonlinear ANC systems. The Fx-SM-VEB algorithm greatly reduces the computational complexity and avoids the difficult problem of choosing a predetermined threshold thanks to the data-selection update strategy and the time-varying error bound. Our study also shows that the algorithm based on the SMF technique using the time-varying bound outperforms that of using a pre-specified bound, especially in a nonlinear environment. The proposed RFx-SM-VEB algorithm can combat the impulse noise that occurs in a nonlinear ANC system, which is implemented by introducing an impulsive-free estimation into the time-varying error bound to ensure the stability of the algorithm. Analysis of the computational complexity and the steady-state output MSE has also been derived. Comparison results between the Fx-LMS, Fx-SM, and proposed Fx-SM-VEB algorithms in the Gaussian environment; as well as between the Fx-LogLMS, Fx-LogLMP, RFx-LMS, and proposed RFx-SM-VEB algorithms in the impulsive noise environment have demonstrated the efficiency of the proposed algorithms for nonlinear ANC system. In addition, we believe that the proposed algorithms will be highly efficient when applied to the multichannel ANC system. A study on developing these algorithms for the multichannel ANC system will be conducted in the near future.

## APPENDIX

From (37), we can deduce that

$$\begin{aligned}
&\zeta_{ex}(n+1)\\
&\quad = trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)(\boldsymbol{U}\boldsymbol{U}^T + \boldsymbol{U}\boldsymbol{V}^T + \boldsymbol{V}\boldsymbol{U}^T + \boldsymbol{V}\boldsymbol{V}^T)]\}\\
&\quad = trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{U}\boldsymbol{U}^T]\}\\
&\qquad + trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{U}\boldsymbol{V}^T]\}\\
&\qquad + trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{V}\boldsymbol{U}^T]\}\\
&\qquad + trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{V}\boldsymbol{V}^T]\}\\
&\quad = \beta_1 + \beta_2 + \beta_3 + \beta_4. \qquad (57)
\end{aligned}$$

For calculating $\beta_1$, we have:

$$\begin{aligned}\beta_1 &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{U}\boldsymbol{U}^T]\}\\ &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)[\triangle\boldsymbol{W}(n) - \frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n)]\\ &\quad\times[\triangle\boldsymbol{W}(n) - \frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n)]^T]\}\\ &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}(n)^T]\}\\ &\quad - 2trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n)\triangle\boldsymbol{W}(n)^T]\}\\ &\quad + \frac{1}{N^2\delta_s^4}trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)e^2(n)\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \beta_{11} + \beta_{12} + \beta_{13}, \end{aligned} \tag{58}$$

$$\beta_{11} = trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}(n)^T]\} = \zeta_{ex}(n), \tag{59}$$

$$\begin{aligned}\beta_{12} &= 2trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n)\triangle\boldsymbol{W}(n)^T]\}\\ &= 2trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\frac{1}{N\delta_s^2}\boldsymbol{S}_f(n)[\upsilon(n)\\ &\quad + \boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)]\triangle\boldsymbol{W}(n)^T]\}\\ &= 2\frac{1}{N\delta_s^2}N\delta_s^2 trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}(n)^T]\}\\ &= 2\zeta_{ex}(n), \end{aligned} \tag{60}$$

$$\begin{aligned}\beta_{13} &= \frac{1}{N^2\delta_s^4}trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)e^2(n)\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \frac{1}{N^2\delta_s^4}trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)[\upsilon(n) + \boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)]^T\\ &\quad\times[\upsilon(n) + \boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)]\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\} = \delta_\upsilon^2 + \frac{1}{N^2\delta_s^4}\\ &\quad\times trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)^T\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\\ &\quad\times\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\} = \delta_\upsilon^2 + \frac{1}{N^2\delta_s^4}trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\\ &\quad\times\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}(n)^T\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \delta_\upsilon^2 + trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}(n)^T]\}\\ &= \delta_\upsilon^2 + \zeta_{ex}(n). \end{aligned} \tag{61}$$

For calculating $\beta_2$, we have:

$$\begin{aligned}\beta_2 &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{U}\boldsymbol{V}^T]\}\\ &= trace\{E[[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)][\triangle\boldsymbol{W}(n) - \frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n)]\\ &\quad\times[\frac{1}{N\delta_s^2}\frac{\xi(n)}{|e_0(n)|}e(n)\boldsymbol{S}_f(n)]^T]\} = \frac{\xi(n)}{N\delta_s^2}E\left(\frac{1}{|e_0(n)|}\right)\\ &\quad\times trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)e(n)\boldsymbol{S}_f^T(n)]\}\\ &\quad - \frac{\xi(n)}{N^2\delta_s^4}E\left(\frac{e^2(n)}{|e_0(n)|}\right)trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \beta_{21} - \beta_{22}, \end{aligned} \tag{62}$$

$$\begin{aligned}\beta_{21} &= \frac{\xi(n)}{N\delta_s^2}E\left(\frac{1}{|e_0(n)|}\right)\\ &\quad\times trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)e(n)\boldsymbol{S}_f^T(n)]\}\\ &= \frac{\xi(n)}{N\delta_s^2}E\left(\frac{1}{|e_0(n)|}\right)trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)[\upsilon(n)\\ &\quad + \triangle\boldsymbol{W}(n)^T\boldsymbol{S}_f(n)]\boldsymbol{S}_f^T(n)]\} = \frac{\xi(n)}{N\delta_s^2}E\left(\frac{1}{|e_0(n)|}\right)\\ &\quad\times trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)\triangle\boldsymbol{W}(n)^T\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \xi(n)E\left(\frac{1}{|e_0(n)|}\right)\zeta_{ex}(n), \end{aligned} \tag{63}$$

$$\begin{aligned}\beta_{22} &= \frac{\xi(n)}{N^2\delta_s^4}E\left(\frac{e^2(n)}{|e_0(n)|}\right)\\ &\quad\times trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \xi(n)E\left(\frac{e^2(n)}{|e_0(n)|}\right). \end{aligned} \tag{64}$$

For calculating $\beta_3$, we have:

$$\begin{aligned}\beta_3 &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{V}\boldsymbol{U}^T]\}\\ &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)[\frac{1}{N\delta_s^2}\frac{\xi(n)}{|e_0(n)|}e(n)\boldsymbol{S}_f(n)]\\ &\quad\times[\triangle\boldsymbol{W}(n) - \frac{1}{N\delta_s^2}e(n)\boldsymbol{S}_f(n)]^T]\}\\ &= \frac{\xi(n)}{N\delta_s^2}E\left(\frac{1}{|e_0(n)|}\right)\\ &\quad\times trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\triangle\boldsymbol{W}(n)e(n)\boldsymbol{S}_f^T(n)]\}\\ &\quad - \frac{\xi(n)}{N^2\delta_s^4}E\left(\frac{e^2(n)}{|e_0(n)|}\right)trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \beta_2. \end{aligned} \tag{65}$$

For calculating $\beta_4$, we have:

$$\begin{aligned}\beta_4 &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{V}\boldsymbol{V}^T]\}\\ &= trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)[\frac{1}{N\delta_s^2}\frac{\xi(n)}{|e_0(n)|}e(n)\boldsymbol{S}_f(n)]\\ &\quad\times[\frac{1}{N\delta_s^2}\frac{\xi(n)}{|e_0(n)|}e(n)\boldsymbol{S}_f(n)]^T]\}\\ &= \frac{\xi^2(n)}{N^2\delta_s^4}E\left(\frac{e^2(n)}{|e_0(n)|}\right)trace\{E[\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)\boldsymbol{S}_f(n)\boldsymbol{S}_f^T(n)]\}\\ &= \xi^2(n)E\left(\frac{e^2(n)}{|e_0(n)|}\right). \end{aligned} \tag{66}$$

## REFERENCES

[1] S. M. Kuo and D. R. Morgan, "Active noise control: A tutorial review," *Proc. IEEE*, vol. 87, no. 6, pp. 943–973, Jun. 1999.
[2] N. V. George and G. Panda, "Advances in active noise control: A survey, with emphasis on recent nonlinear techniques," *Signal Process.*, vol. 93, pp. 363–377, Feb. 2013.
[3] L. Lu, K.-L. Yin, R. C. de Lamare, Z. Zheng, Y. Yu, X. Yang, and B. Chen, "A survey on active noise control in the past decade—Part II: Nonlinear systems," *Signal Process.*, vol. 181, Apr. 2021, Art. no. 107929.
[4] L. Tan and J. Jiang, "Adaptive Volterra filters for active control of nonlinear noise processes," *IEEE Trans. Signal Process.*, vol. 49, no. 8, pp. 1667–1676, Aug. 2001.
[5] D. P. Das and G. Panda, "Active mitigation of nonlinear noise processes using a novel filtered-s LMS algorithm," *IEEE Trans. Speech Audio Process.*, vol. 12, no. 3, pp. 313–322, May 2004.
[6] P. Strauch and B. Mulgrew, "Active control of nonlinear noise processes in a linear duct," *IEEE Trans. Signal Process.*, vol. SP-46, no. 9, pp. 2404–2412, Sep. 1998.

[7] X. Guo, Y. Li, J. Jiang, C. Dong, S. Du, and L. Tan, "Adaptive function expansion 3-D diagonal-structure bilinear filter for active noise control of saturation nonlinearity," *IEEE Access*, vol. 6, pp. 65139–65150, 2018.
[8] D. Zhou and V. DeBrunner, "Efficient adaptive nonlinear filters for nonlinear active noise control," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 3, pp. 669–681, Mar. 2007.
[9] G. L. Sicuranza and A. Carini, "On the BIBO stability condition of adaptive recursive FLANN filters with application to nonlinear active noise control," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 234–245, Jan. 2012.
[10] G. L. Sicuranza and A. Carini, "A generalized FLANN filter for nonlinear active noise control," *IEEE Trans. Audio, Speech, Language Process.*, vol. 19, no. 8, pp. 2412–2417, Nov. 2011.
[11] D. C. Le, J. Zhang, D. Li, and S. Zhang, "A generalized exponential functional link artificial neural networks filter with channel-reduced diagonal structure for nonlinear active noise control," *Appl. Acoust.*, vol. 139, pp. 174–181, Oct. 2018.
[12] D. C. Le, J. Zhang, and Y. Pang, "A bilinear functional link artificial neural network filter for nonlinear active noise control and its stability condition," *Appl. Acoust.*, vol. 132, pp. 19–25, Mar. 2018.
[13] X. Guo, J. Jiang, L. Tan, and S. Du, "Improved adaptive recursive even mirror Fourier nonlinear filter for nonlinear active noise control," *Appl. Acoust.*, vol. 146, pp. 310–319, Mar. 2019.
[14] V. Patel and N. V. George, "Partial update even mirror Fourier non-linear filters for active noise control," in *Proc. 23rd Eur. Signal Process. Conf. (EUSIPCO)*, Nice, France, Aug. 2015, pp. 295–299.
[15] L. Lu and H. Zhao, "Adaptive Volterra filter with continuous *lp*-norm using a logarithmic cost for nonlinear active noise control," *J. Sound Vib.*, vol. 364, pp. 14–29, Mar. 2016.
[16] G. L. Sicuranza and A. Carini, "Filtered-X affine projection algorithm for multichannel active noise control using second-order Volterra filters," *IEEE Signal Process. Lett.*, vol. 11, no. 11, pp. 853–857, Nov. 2004.
[17] 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 Process. Lett.*, vol. 5, no. 5, pp. 111–114, May 1998.
[18] K. Chen, S. Werner, A. Kuh, and Y.-F. Huang, "Nonlinear adaptive filtering with kernel set-membership approach," *IEEE Trans. Signal Process.*, vol. 68, pp. 1515–1528, 2020.
[19] I. Hassani, M. Arezki, and A. Benallal, "A novel set membership fast NLMS algorithm for acoustic echo cancellation," *Appl. Acoust.*, vol. 163, Jun. 2020, Art. no. 107210.
[20] Y. Li, Y. Wang, and T. Jiang, "Sparse-aware set-membership NLMS algorithms and their application for sparse channel estimation and echo cancelation," *AEU-Int. J. Electron. Commun.*, vol. 70, no. 7, pp. 895–902, 2016.
[21] Y. Li, Z. Jiang, O. M. Osman, X. Han, and J. Yin, "Mixed norm constrained sparse APA algorithm for satellite and network echo channel estimation," *IEEE Access*, vol. 6, pp. 65901–65908, 2018.
[22] Y. Li, Y. Wang, and L. Sun, "A flexible sparse set-membership NLMS algorithm for multi-path and acoustic echo channel estimations," *Appl. Acoust.*, vol. 148, pp. 390–398, May 2019.
[23] T. Wang, R. C. de Lamare, and P. D. Mitchell, "Low-complexity set-membership channel estimation for cooperative wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 60, no. 6, pp. 2594–2607, Jul. 2011.
[24] R. C. de Lamare and P. S. R. Diniz, "Set-membership adaptive algorithms based on time-varying error bounds for CDMA interference suppression," *IEEE Trans. Veh. Technol.*, vol. 58, no. 2, pp. 644–654, Feb. 2009.
[25] S. Zhang and J. Zhang, "Set-membership NLMS algorithm with robust error bound," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 61, no. 7, pp. 536–540, Jul. 2014.
[26] Z. Zheng, Z. Liu, H. Zhao, Y. Yu, and L. Lu, "Robust set-membership normalized subband adaptive filtering algorithms and their application to acoustic echo cancellation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 8, pp. 2098–2111, Aug. 2017.
[27] S. Zhang, J. Zhang, and Y. Pang, "Pipelined set-membership approach to adaptive Volterra filtering," *Signal Process.*, vol. 129, pp. 195–203, Dec. 2016.
[28] A. Carini and G. L. Sicuranza, "Analysis of a multichannel filtered-x set-membership affine projection algorithm," in *Proc. IEEE Int. Conf. Acoust. Speed Signal Process.*, vol. 3. Toulouse, France, May 2006, pp. 1–4.
[29] F. Albu, "Set-membership sparsity-aware proportionate normalized least mean square algorithms for active noise control," in *Proc. IEEE East-West Design Test Symp. (EWDTS)*, Varna, Bulgaria, Sep. 2020, pp. 1–4.
[30] M. Z. A. Bhotto and A. Antoniou, "Robust quasi-Newton adaptive filtering algorithms," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 58, no. 8, pp. 537–541, Aug. 2011.
[31] D. P. Sergio, *Adaptive Filtering: Algorithms and Practical Implementation*. Nowell, MA, USA: Kluwer, 2002.
[32] P. S. R. Diniz and S. Werner, "Set-membership binormalized data-reusing LMS algorithms," *IEEE Trans. Signal Process.*, vol. 51, no. 1, pp. 124–134, Jan. 2007.
[33] L. Wu, H. He, and X. Qiu, "An active impulsive noise control algorithm with logarithmic transformation," *IEEE Trans. Audio, Speech, Language Process.*, vol. 19, no. 4, pp. 1041–1044, May 2011.
[34] N. V. George and G. Panda, "A robust filtered-s LMS algorithm for nonlinear active noise control," *Appl. Acoust.*, vol. 73, no. 8, pp. 836–841, 2012.
[35] R. Leahy, Z. Zhou, and Y.-C. Hsu, "Adaptive filtering of stable processes for active attenuation of impulsive noise," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, vol. 5. Detroit, MI, USA, 1995, pp. 2983–2986.
[36] F. Jabeen, A. Mirza, A. Zeb, M. Imran, F. Afzal, and A. Maqbool, "FxRLS algorithms based active control of impulsive noise with online secondary path modeling," *IEEE Access*, vol. 9, pp. 117471–117485, 2021.

**DINH CONG LE** received the B.S. degree from Hanoi National University, Vietnam, in 2001, the M.S. degree from Le Quy Don Technical University, Hanoi, Vietnam, in 2011, and the Ph.D. degree in signal and information processing from Southwest Jiaotong University, Chengdu, China, in 2019. He is currently a Lecturer with the School of Engineering and Technology, Vinh University, Vietnam. His current research interests include adaptive signal processing, active noise control, and control systems.

**HOANG HUU VIET** received the B.S. degree in mathematics from Vinh University, Nghe An, Vietnam, in 1994, the B.S. and M.S. degrees in computer science from the Hanoi University of Technology, Hanoi, Vietnam, in 1998 and 2002, respectively, and the Ph.D. degree in computer engineering from Kyung Hee University, Republic of Korea, in August 2013. He is currently an Associate Professor with the School of Engineering and Technology, Vinh University. His current research interests include artificial intelligence, reinforcement learning, and adaptive signal processing.