Al Omour et al. EURASIP Journal on Advances in Signal Processing (2016) 2016:39
DOI 10.1186/s13634-016-0337-z

EURASIP Journal on Advances in Signal Processing

RESEARCH Open Access

Filtered-X Least Mean Fourth (FXLMF) and Leaky FXLMF adaptive algorithms

Ali M. Al Omour, Abdelmalek Zidouri, Naveed Iqbal and Azzedine Zerguine*

Abstract

Adaptive filtering algorithms promise an improvement of the active noise control (ANC) problem encountered in many scenarios. Just to name a few, the Filtered-X Least Mean Square (FXLMS) algorithm, the Leaky FXLMS (LFXLMS) algorithm, and other modified LMS-based algorithms have been developed and utilized to combat the ANC problem. All of these algorithms enjoy great performance when the signal-to-noise ratio (SNR) is high. On the other hand, when the SNR is low, which is a known trend in ANC scenarios, the performance of these algorithms is not attractive. The performance of the Least Mean Fourth (LMF) algorithm has never been tested on any ANC scenario under low or high SNR. Therefore, in this work, reflecting the development in the LMS family on the LMF, we are proposing two new adaptive filtering algorithms, which are the Filtered-X Least Mean Fourth (FXLMF) algorithm and the Leakage-based variant (LFXLMF) of the FXLMF algorithm. The main target of this work is to derive the FXLMF and LFXLMF adaptive algorithms, study their convergence behaviors, examine their tracking and transient conduct, and analyze their performance for different noise environments. Moreover, a convex combination filter utilizing the proposed algorithm and algorithm robustness test is carried out. Finally, several simulation results are obtained to validate the theoretical findings and show the effectiveness of the proposed algorithms over other adaptive algorithms.

1 Introduction

Adaptive filtering algorithms are by now omnipresent in a variety of applications, such as plant modeling, adaptive equalization, and system identification, to name a few [1–8]. Add to that, noise control and noise cancelation are important issues whose effects adaptive filtering algorithms strive to mitigate. Active noise control (ANC) techniques use adaptive filtering algorithms to cancel the effect of acoustic noise, by playing anti-noise signal estimated from the noise source itself.

The Least Mean Square (LMS) algorithm suffers from problems, such as a degradation in the algorithm efficiency, due to the presence of a filter in the auxiliary or error path, as in the case of the ANC technique, as well as slow convergence, instability of the algorithm, increased residual noise power, and lower convergence rate. These constraints urged researchers to enhance the performance of the conventional LMS algorithm [9–11].

The Filtered-X LMS (FXLMS) algorithm is considered as the cornerstone for ANC applications [12–15]. In this algorithm, an identical copy of the secondary path, mainly used to solve the instability problem and to eliminate the noise from the primary signal, is used to filter the input before the adaptive algorithm uses it in order to adjust the coefficient vector of the adaptive filter, as depicted in Fig. 1 [9]. More details about the different parameters in Fig. 1 will be given in the next section.

In the last decade, intensive research was carried out for the purpose of enhancing the performance of the FXLMS algorithm. In [14], a new stochastic analysis for the FXLMS algorithm was introduced, using an analytical model not based on the independence theory [16], to derive the first moment of the adaptive weight filter. The main assumption of this work was to ignore the correlation between the data vector and the weights and compare the correlation between data vectors, preserving past and present data vector correlations. This model was validated for both white and colored primary input signals and shows stability even when using large step sizes.

The FXLMS algorithm is preferred because of its inherent stability and simplicity, but sometimes, the adaptive filter suffers from high noise levels caused by low-frequency resonances, which may cause nonlinear distortion due to overloading of the secondary source.

* Correspondence: azzedine@kfupm.edu.sa
Electrical Engineering Department, KFUPM, Dhahran, Saudi Arabia

This problem was solved by adding output power constraints to the cost function, as was proposed in the Leaky FXLMS (LFXLMS) algorithm [17]. Moreover, the LFXLMS reduces the numerical error in the finite precision implementation and limits the output power of the secondary source to avoid nonlinear distortion. The LFXLMS increases the algorithm's stability, especially when a large source strength is used.

Another modification of the FXLMS algorithm is the Modified FXLMS (MFXLMS) algorithm [15]. Since the FXLMS exhibits poor convergence performance, the MFXLMS proposes a better convergence and reduces the computational load.

The LMS may suffer from divergence due to the insufficient spectral excitation, like a sinusoid signal without noise, which consequently may cause overflow for the weight vector during the updating process. This divergence problem can be resolved by proposing a leak term during the update process of the weight vector. Of course, this will result in lesser performance; however, the leakage factor is controlled, which is necessary to balance for the lost performance. In addition, this will add complexity, but more robustness of the adaptive filter is achieved, as was done in the case of the Leaky LMS (LLMS) algorithm.

In [17], a stochastic analysis for the LFXLMS algorithms was proposed without resorting to the independence theory. Furthermore, to strengthen their work, the authors assumed an inexact estimation for the secondary path, which is the case in most practical implementations for the adaptive filter.

Due to very low input signal, the Leaky LMS algorithm proposed in [18] aims to reduce the stalling effect, where the gradient estimate is too small to adjust the coefficients of the algorithm. Moreover, the leakage term stabilized the LMS algorithm successfully. Also, the LLMS solved the problem of bursting in short-distance telephones when they added the adaptive echo canceller [19].

A very important extension of the LMS algorithm is the Least Mean Fourth (LMF) algorithm [20], where the cost function for LMF algorithm, defined in terms of the error signal ($e(n)$) is given by

$$J_{\mathrm{LMF}}(n) = E\left[e^4(n)\right]. \tag{1}$$

The LMF weights converge proportionally to the LMS weights. The performance of the LMF algorithm has never been tested on any ANC scenario under low or high signal-to-noise ratio (SNR). In this work, we propose two new algorithms, the Filtered-X LMF (FXLMF) and Leaky FXLMF (LFXLMF) algorithms. We analyze the convergence behaviors and examine the performance of both of them. This is carried out under different statistical input signals and noise for the mean and mean square error of the adaptive filter weights, depending on secondary path modeling error using an energy conservation relation framework. These two algorithms are expected to have a high effectiveness on the ANC issue at an extra computational complexity. Monte Carlo simulations used to assess the analytical assumptions, as well as the accuracy of the proposed model, are verified and assessed.

This paper is organized as follows: Section 1 provides an introduction and a literature review. In Section 2, analytical derivations for the FXLMF and LFXLMF algorithms are presented. Section 3 proposes the convex combination of the FXLMF algorithm with the FXLMS algorithm. Simulation results are presented in Section 4, and finally, the conclusions and future work are presented in Section 5.

2 Analysis

Figure 1 illustrates the block diagram of an ANC and illustrates the location of the secondary path $\boldsymbol{S}$ and its estimated secondary path $\hat{\boldsymbol{S}}$. The secondary path is a transfer function which can be represented by a group of a digital-to-analog (D/A) converter, a power amplifier, a

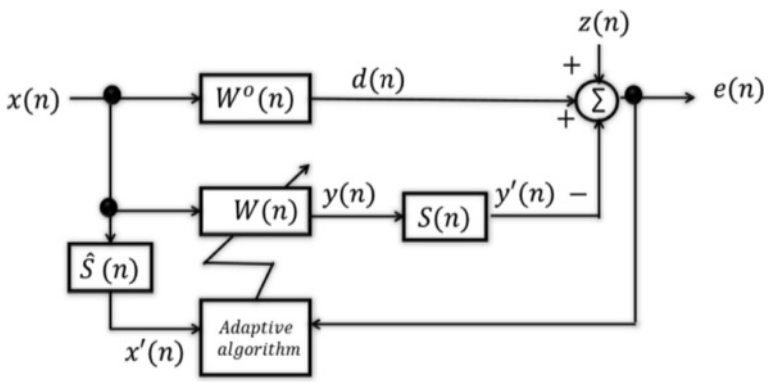

Fig. 1 Block diagram for ANC system

canceling loudspeaker, an error microphone, and an (A/D) converter.

The realization of the secondary path is usually obtained using a system identification technique. For this work, the assumption used considers an inexact estimation for the secondary path which may cause errors on the number of coefficients or on their values as was done in [21]. Consequently, the values of the secondary path will be as $\hat{\boldsymbol{S}} \neq \boldsymbol{S}$, and the filter coefficients' number $\hat{M} \neq M$. Table 1 describes the different parameters used in Fig. 1.

Referring to Fig. 1, the error signal is given by

$$e(n) = d(n) - y'(n) + z(n), \tag{2}$$

where $d(n)$ is the desired response and $y(n)$ is the output of the adaptive filter given by

$$y(n) = \boldsymbol{x}^T(n)\boldsymbol{w}(n) = \boldsymbol{w}^T(n)\boldsymbol{x}(n), \tag{3}$$

$y'(n)$ is the output of the secondary path

$$\begin{aligned} y'(n) &= \sum_{i=0}^{M-1} s_i y(n-i) \\ &= \sum_{i=0}^{M-1} s_i \boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i), \end{aligned} \tag{4}$$

and $z(n)$ is the active noise. Finally, the filtered input signal is given as

$$x'(n) = \sum_{i=0}^{\hat{M}-1} \hat{s}_i \boldsymbol{x}(n-i). \tag{5}$$

For the case of an exact approximation for the secondary path, that is $\boldsymbol{S} = \hat{\boldsymbol{S}}$, the input signal, $\boldsymbol{x}(n)$, will be filtered by $\boldsymbol{S}$.

2.1 Development of FXLMF algorithm

Using the block diagram in Fig. 1, the cost function for the FXLMF algorithm is given by the following relation:

$$J_{\mathrm{FXLMF}}(n) = E\left[e^4(n)\right], \tag{6}$$

Table 1 Parameters and their descriptions used in Fig. 1

Adaptive filter weights	$\boldsymbol{w}(n) = [w_0(n)\ w_1(n) \ldots w_{N-1}(n)]^T$
Stationary input signal	$\boldsymbol{x}(n) = [x(n)\ x(n-1) \ldots x(n-N+1)]^T$
Secondary path	$\boldsymbol{S} = [s_0\ s_1 \ldots s_{M-1}]^T$
Estimate of the secondary path	$\hat{\boldsymbol{S}} = [\hat{s}_0\ \hat{s}_1 \ldots \hat{s}_{M-1}]^T$
Primary (desired) signal	$d(n)$
Stationary noise process	$z(n)$
Number of tap weight coefficients	N
Number of the secondary path coefficients	M

where the error signal, $e(n)$, is given below as the difference between the output signal from the secondary path and the primary signal, that is,

$$e(n) = d(n) - \sum_{i=0}^{M-1} s_i \boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i) + z(n). \tag{7}$$

During the course of derivations, we will resort to the same assumptions, used in the literature [17–23], to simplify our algorithms. These assumptions are as follows:

2.1.1 Assumption A1

$\boldsymbol{x}(n)$ is the input signal, a zero mean wide-stationary Gaussian process with variance ${\sigma_x}^2$, and $\boldsymbol{R_{i,j}} = E[\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)] > 0$ is a positive definite autocorrelation matrix of the input signal.

2.1.2 Assumption A2

$z(n)$ is the measurement noise, an independent and identically distributed (i.i.d) random variable with zero mean and variance $\sigma_z^2 = E[z^2(n)]$, and there is no correlation between the input signal and the measurement noise. In other words, the sequence $z(n)$ is independent of $\boldsymbol{x}(n)$ and $\boldsymbol{w}(n)$. The measurement is assumed to have an even probability density function.

Assuming that the vector $\boldsymbol{w}$ is fixed, then the cost function looks like the following:

$$\begin{aligned} J_{\mathrm{FXLMF}} = {} & \left\{\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}\sum_{l=0}^{M-1} s_i s_j s_k s_l\, E[\boldsymbol{x}^T(n-l)\boldsymbol{x}(n-k)\,\boldsymbol{x}^T(n-j)\boldsymbol{x}(n-i)]\right)\right\}\|\boldsymbol{w}\|^4 \\ & -4\left\{\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1} s_i s_j s_k\, E[d(n)\boldsymbol{x}(n-k)\,\boldsymbol{x}^T(n-j)\boldsymbol{x}(n-i)]\right)\right\}\|\boldsymbol{w}\|^3 \\ & +6\left\{\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1} s_i s_j\, E\left[d^2(n)\,\boldsymbol{x}(n-j)\,\boldsymbol{x}^T(n-i)\right]\right)\right. \\ & \left.+\sigma_z{}^2\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1} s_i s_j\, E[\boldsymbol{x}(n-j)\,\boldsymbol{x}^T(n-i)]\right)\right\}\|\boldsymbol{w}\|^2 \\ & -4\left\{\left(\sum_{i=0}^{M-1} s_i\, E\left[d^3(n)\,\boldsymbol{x}^T(n-i)\right]\right)\right. \\ & \left.-\sigma_z{}^2\left(\sum_{i=0}^{M-1} s_i\, E[d(n)\boldsymbol{x}^T(n-i)]\right)\right\}\|\boldsymbol{w}\| \\ & +\left\{\left(E\left[d^4(n)\right] + \sigma_z{}^2 - 4\,E\left[z^3(n)\right]E\left[d(n)\right] + 6\,\sigma_d{}^2\sigma_z{}^2\right)\right\}. \end{aligned} \tag{8}$$

To obtain the optimal weight vector for the cost function, we take the derivative of Eq. (8) with respect to $\boldsymbol{w}$ and set it to zero. Discarding the noise, $z(n)$, the derivative of Eq. (8) will be

$$\frac{\partial J_{\mathrm{FXLMF}}(n)}{\partial \boldsymbol{w}(n)} = \|\boldsymbol{w}\|^3 - 3\left(\tilde{\boldsymbol{R}}_{s^4}{}^{-1}\tilde{\boldsymbol{P}}_{d,s^3}\right)\|\boldsymbol{w}\|^2 + 3\tilde{\boldsymbol{R}}_{s^4}{}^{-1}\left(\tilde{\boldsymbol{P}}_{d^2,s^2}\right) \times \|\boldsymbol{w}\| - \tilde{\boldsymbol{R}}_{s^4}{}^{-1}\left(\tilde{\boldsymbol{P}}_{d^3,s}\right), \quad (9)$$

where

$$\tilde{\boldsymbol{R}}_{s^2} = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1} s_i s_j\, E\left[\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\right] = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1} s_i s_j\, \boldsymbol{R}_{i,j},$$

$\boldsymbol{R}_{i,j} = E\ [\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)]$ is the input autocorrelation matrix,

$$\tilde{\boldsymbol{P}}_{d,s} = \sum_{j=0}^{M-1} s_j\, E\ [d(n)\boldsymbol{x}(n-j)] = \sum_{j=0}^{M-1} s_j P_{d,j},$$

$\boldsymbol{P}_{d,j} = E\ [d(n)\boldsymbol{x}(n-j)]$ is the cross-correlation between the input and the primary signals,

$$\tilde{\boldsymbol{R}}_{s^4} = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}\sum_{l=0}^{M-1} s_i s_j s_k s_l\, E[\boldsymbol{x}^T(n-l)\boldsymbol{x}(n-k)\ \boldsymbol{x}^T(n-j)\boldsymbol{x}(n-i)] = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}\sum_{l=0}^{M-1} s_i s_j s_k s_l\, \boldsymbol{R}_{i,j,k,l},$$

$$\tilde{\boldsymbol{P}}_{d,s^3} = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1} s_i s_j s_k\, E[d(n)\boldsymbol{x}(n-k)\ \boldsymbol{x}^T(n-j)\boldsymbol{x}(n-i)] = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1} s_i s_j s_k\, \boldsymbol{P}_{d,i,j,k},$$

$$\tilde{\boldsymbol{P}}_{d^2,s^2} = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1} s_i s_j\, E\left[d^2(n)\ \boldsymbol{x}(n-j)\ \boldsymbol{x}^{\boldsymbol{T}}(n-i)\right] = \sum_{i=0}^{M-1}\sum_{j=0}^{M-1} s_i s_j \boldsymbol{P}_{d^2,i,j},$$

and

$$\tilde{\boldsymbol{P}}_{d^3,s} = \sum_{i=0}^{M-1} s_i\, E\left[d^3(n)\ \boldsymbol{x}^T(n-i)\right] = \sum_{i=0}^{M-1} s_i\ \boldsymbol{P}_{d^3,i}.$$

Equation (9) has three solutions, and the optimal solution is given by

$$\boldsymbol{w_o} = \tilde{\boldsymbol{R}}_{s^2}{}^{-1}\tilde{\boldsymbol{P}}_{d,s}. \quad (10)$$

2.2 Mean behavior for the FXLMF algorithm

The FXLMF algorithm is governed by the following recursion:

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) - \frac{\mu}{4}\ \frac{\partial J_{\mathrm{FXLMF}}(n)}{\partial \boldsymbol{w}(n)}, \quad (11)$$

where the instantaneous gradient can be approximated as

$$\frac{\partial \hat{j}_{\mathrm{FXLMF}}(n)}{\partial \boldsymbol{w}(n)} \approx -4\ e^3(n)\sum_{i=1}^{\hat{M}-1}\hat{s}_i\boldsymbol{x}^T(n-i) \quad (12)$$

due to the absence of the exact knowledge of the secondary path. Substituting Eqs. (2)–(5) and (11) in (10), the adaptive weight vector update is given by

$$\begin{aligned}\boldsymbol{w}(n+1) = \boldsymbol{w}\ (n) &+ \mu\sum_{i=0}^{\hat{M}-1}\hat{s}_i\ d^3(n)\boldsymbol{x}(n-i)\\ &-3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1} s_i\hat{s}_j\ d^2(n)\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i)\right)\\ &+3\ \mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{\hat{M}-1} s_i s_j\hat{s}_k\ d(n)\boldsymbol{x}(n-k)\boldsymbol{x}^T(n-j)\ \boldsymbol{x}(n-i)\boldsymbol{w}(n-j)\ \boldsymbol{w}^T(n-i)\right)\\ &-6\ \mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1} s_i\hat{s}_j\ z(n)\ d(n)\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i)\right)\\ &-\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}\sum_{l=0}^{\hat{M}-1} s_i s_j s_k\hat{s}_l\ \boldsymbol{x}(n-l)\boldsymbol{x}^T(n-k)\ \boldsymbol{x}(n-j)\right.\\ &\qquad\left.\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-k)\ \boldsymbol{w}^T(n-j)\boldsymbol{w}(n-i)\right)\\ &+3\ \mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{\hat{M}-1} s_i s_j\hat{s}_k\ z(n)\boldsymbol{x}(n-k)\boldsymbol{x}^T(n-j)\ \boldsymbol{x}(n-i)\boldsymbol{w}(n-j)\ \boldsymbol{w}^T(n-i)\right)\\ &-3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1} s_i\hat{s}_j\ z^2(n)\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i)\right)\\ &+\mu\left(\sum_{i=0}^{\hat{M}-1}\hat{s}_i\ z^3(n)\boldsymbol{x}(n-i)\right)+3\,\mu\left(\sum_{i=0}^{\hat{M}-1}\hat{s}_i\ d(n)z^2(n)\boldsymbol{x}(n-i)\right)\\ &+3\,\mu\left(\sum_{i=0}^{\hat{M}-1}\hat{s}_i\ d^2(n)z(n)\boldsymbol{x}(n-i)\right).\end{aligned} \quad (13)$$

To find the expectations for the terms on the right-hand side of Eq. (13), we resort to the following assumptions [1, 21]:

2.2.1 Assumption A3

Independence theory (IT) states that the taps of the input vector $x(n-i)$, $i = 0,1,2,\ldots$ are statistically dependent so that $E[\boldsymbol{x}(n-i)\boldsymbol{x}^T(n-j)] = E[\boldsymbol{x}(n-i)\boldsymbol{x}^T(n-j)\boldsymbol{x}(n-k)] = E[\boldsymbol{x}(n-l)\boldsymbol{x}^T(n-k)\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)] = 0$, for any $i \neq j$, $i \neq j \neq k$, and $i \neq j \neq k \neq l$, respectively.

2.2.2 Assumption A4

Take into consideration the correlation between $\boldsymbol{x}(n-i)$, $\boldsymbol{x}(n-j)$, $\boldsymbol{x}(n-k)$, and $\boldsymbol{x}(n-l)\forall\ i, j, k, l$ and ignore the correlation between $\boldsymbol{w}(n-v)$ and $\boldsymbol{x}(n-i)$ or $\boldsymbol{x}(n-j)$ or $\boldsymbol{x}(n-k)\forall\ i, j, k$.

Using assumption *A3*, the mean weight update recursion for the FXLMF algorithm will look like the following:

$$\begin{aligned}E[\boldsymbol{w}(n+1)] = E\left[\boldsymbol{w}\,(n)\right] + \mu\sum_{i=0}^{\hat{M}-1}\hat{s}_i\,\boldsymbol{P}_{d^3,i} - 3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1}s_i\hat{s}_j\;\boldsymbol{P}_{d^2,i,j}E[\boldsymbol{w}\,(n-i)]\right)\\ +3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{\hat{M}-1}s_i s_j\hat{s}_k\,\boldsymbol{P}_{d,i,j,k}E[\boldsymbol{w}(n-j)\boldsymbol{w}^T(n-i)]\right)\\ -\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}\sum_{l=0}^{\hat{M}-1}s_i s_j s_k\hat{s}_l\,\boldsymbol{R}_{\boldsymbol{i,j,k,l}}E[\boldsymbol{w}(n-k)\boldsymbol{w}^T(n-j)\boldsymbol{w}(n-i)]\right)\\ -3\,\mu\,{\sigma_z}^2\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1}s_i\hat{s}_j\,\boldsymbol{R}_{\boldsymbol{i,j}}E[\boldsymbol{w}\,(n-i)]\right) + 3\,\mu\,{\sigma_z}^2\left(\sum_{i=0}^{\hat{M}-1}\hat{s}_i\,\boldsymbol{P}_{d,i}\right).\end{aligned} \tag{14}$$

Consequently, after taking into account the independence theory, Eq. (14) looks like the following:

$$\begin{aligned}E[\boldsymbol{w}(n+1)] = E[\boldsymbol{w}(n)] + \mu\,\hat{s}_0\,E\left[d^3(n)\boldsymbol{x}^{\boldsymbol{T}}(n)\right]\\ -3\,\mu\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_i\hat{s}_i\,\boldsymbol{P}_{d^2,i,j}E[\boldsymbol{w}\,(n-i)]\right)\\ -3\,\mu{\sigma_z}^2\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_i\hat{s}_i\boldsymbol{R}_{\boldsymbol{i,j}}E[\boldsymbol{w}\,(n-i)]\right)\\ +3\,\mu\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_i s_i\hat{s}_i\,\boldsymbol{P}_{d,i,j,k}E[\boldsymbol{w}(n-i)\boldsymbol{w}^T(n-i)]\right)\\ +3\,\mu\,{\sigma_z}^2\hat{s}_0\,E[d(n)\boldsymbol{x}(n)]\\ -\mu\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_i s_i s_i\hat{s}_i\boldsymbol{R}_{\boldsymbol{i,j,k,l}}\,E[\boldsymbol{w}(n-i)\boldsymbol{w}^T(n-i)\boldsymbol{w}(n-i)]\right).\end{aligned} \tag{15}$$

Since in a practical situation, an exact modeling for the secondary path cannot be achieved, which may lead to incorrect number of tap weights, such as $\hat{M} < M$ or may have the same number of taps but they do not have the same $\hat{S} \neq S$. Here, we consider the case of overestimation for the secondary path, as was the case for Eq. (12). Moreover, to study the steady-state condition, we assume that the optimal solution of tap weights is governed by $\lim_{n\to\infty} E[\boldsymbol{w}(n+1)] = \lim_{n\to\infty} E[\boldsymbol{w}(n)] = \boldsymbol{w}_\infty$; as a result,

$$\boldsymbol{w}_\infty \approx \boldsymbol{w}_o = {\tilde{\boldsymbol{R}}_{s^2}}^{-1}\tilde{\boldsymbol{P}}_{d,s} \tag{16}$$

2.3 Second moment analysis for FXLMF algorithm

Using Eq. (7), the mean square error (MSE) for the FXLMF algorithm is obtained:

$$\begin{aligned}\mathrm{MSE}_{\mathrm{FXLMF}}(n) &= E\left[e^2(n)\right]\\ &= E\left[d^2(n)\right] - 2\sum_{i=0}^{M-1}s_i\,E[d(n)\boldsymbol{x}(n-i)]E[\boldsymbol{w}(n-i)]\\ &+\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}s_i s_j\,E[\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)]E[\boldsymbol{w}(n-i)\boldsymbol{w}^T(n-i)]\\ &+E[z^2(n)] \quad = {\sigma_d}^2 - 2\sum_{i=0}^{M-1}s_i\,\boldsymbol{P}_{d,i}\,E[\boldsymbol{w}(n-i)]\\ &+\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}s_i s_j\,\boldsymbol{R}_{\boldsymbol{i,j}}\,E[\boldsymbol{w}\,(n-i)\boldsymbol{w}^T\,(n-i)] + {\sigma_z}^2.\end{aligned} \tag{17}$$

Next, to find the minimum mean square error (MMSE), we need to substitute the optimal solution of the FXLMF algorithm (16) in (17); moreover, the optimal error is given by

$$e_o(n) = d(n) - \sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}_o \tag{18}$$

Relying on the orthogonality principle [1, 2], the input signal will be orthogonal to the error, and noting that ${\sigma_d}^2 = \tilde{\boldsymbol{P}}_{d,s}{}^{*}\,{\tilde{\boldsymbol{R}}_{s^2}}^{-1}\tilde{\boldsymbol{P}}_{d,s}$, where σ_d^2 is the power of the desired response and $\tilde{\boldsymbol{P}}_{\boldsymbol{d},\boldsymbol{s}}$ and $\tilde{\boldsymbol{R}}_{s^2}$ have been already defined in Section 2.1, the $\mathrm{MMSE}_{\mathrm{FXLMF}}$ can be expressed as follows:

$$\mathrm{MMSE}_{\mathrm{FXLMF}} = {\sigma_z}^2 \tag{19}$$

2.4 FXLMF algorithm stability

Choosing the right value of step size ensures that the algorithm will converge. The FXLMF algorithm weight update equation is given by

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu\,e^3(n)\,x'(n). \tag{20}$$

Then, the algorithm converges when $E[\boldsymbol{w}(n+1)] = E[\boldsymbol{w}(n)]$, that is, the expected value of the weight adjustment term will be zero:

$$\begin{aligned}&\mu\,E[e^3(n)\,x'(n)] = 0\\ \text{or}\\ &\mu\,E\left[\left(d(n) - \sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i) + z(n)\right)^3\left(\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\right)\right] = 0,\end{aligned} \tag{21}$$

since

$$\begin{aligned}e(n) &= d(n) - \sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i) + z(n)\\ &= \sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}_{\boldsymbol{o}} - \sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i) + z(n).\end{aligned} \tag{22}$$

The weight error vector is defined as

$$\boldsymbol{v}(n) = \boldsymbol{w}(n) - \boldsymbol{w_o}. \tag{23}$$

Hence,

$$e(n) = z(n) - \sum_{i=0}^{M-1} s_i \boldsymbol{x}^T(n-i)\boldsymbol{v}(n). \tag{24}$$

Using Eqs. (23) and (24) in Eq. (19), then

$$\boldsymbol{v}(n+1) = \boldsymbol{v}(n) + \mu\left(z(n) - \sum_{i=0}^{M-1} s_i \boldsymbol{x}^T(n-i)\boldsymbol{v}(n)\right)^3 \left(\sum_{i=0}^{M-1} s_i \boldsymbol{x}^T(n-i)\right). \tag{25}$$

The value of $\boldsymbol{v}(n)$ approaches zero when the algorithm converges, and therefore, higher order terms of $\mathbf{v}(n)$ can be ignored, and as a result, the weight error update equation can look like

$$\begin{aligned}\boldsymbol{v}(n+1) \cong & \boldsymbol{v}(n) \\ & + \mu\left(z^3(n) - 3\sum_{i=0}^{M-1} s_i z^2(n)\boldsymbol{x}^T(n-i)\boldsymbol{v}(n)\right) \\ & \times \left(\sum_{i=0}^{M-1} s_i \boldsymbol{x}^T(n-i)\right).\end{aligned} \tag{26}$$

Following assumptions *A1* and *A2* that the noise is independent of the input signal and independent of the weight error vector, the expected value of Eq. (26) results into

$$\begin{aligned}E[\boldsymbol{v}(n+1)] &= E[\boldsymbol{v}(n)] - \mu\left\{3{\sigma_z}^2 \sum_{i=0}^{M-1}\sum_{j=0}^{M-1} s_i E[\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)]E[\boldsymbol{v}(n)]\right\} \\ &= \left[I - \mu\left(3{\sigma_z}^2 \tilde{\boldsymbol{R}}_{s^2}\right)\right]E[\boldsymbol{v}(n)].\end{aligned} \tag{27}$$

Since the autocorrelation matrix $\boldsymbol{R}_{i,j} > 0$, the range of the step size for the FXLMF algorithm can be shown to be given by

$$0 < \mu < \frac{2}{3{\sigma_z}^2 \lambda_{\max}\left(\tilde{\boldsymbol{R}}_{s^2}\right)}, \tag{28}$$

where $\lambda_{\max}\left(\tilde{\boldsymbol{R}}_{s^2}\right)$ represents the maximum eigenvalue of $\tilde{\boldsymbol{R}}_{s^2}$.

2.5 Development of Leaky FXLMF (LFXLMF) algorithm

In this section, the leaky version of the FXLMF algorithm is developed using assumptions *A1–A4*. Using the block diagram in Fig. 1, the cost function for the LFXLMF algorithm will be as follows:

$$\boldsymbol{J}_{\mathrm{LFXLMF}}(n) = E\left[e^4(n)\right] + \gamma\, \boldsymbol{w}^T(n)\boldsymbol{w}(n), \tag{29}$$

where γ is the leakage factor $\gamma \geq 0$. In the case where $\gamma = 0$, then the cost function will be for the FXLMF algorithm, and the error signal, $e(n)$, is given by

$$e(n) = d(n) - \sum_{i=0}^{M-1} s_i \boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i) + z(n). \tag{30}$$

The derivative of the cost function with respect to $\boldsymbol{w}(n)$ will be as follows:

$$\begin{aligned}\frac{\partial J_{\mathrm{LFXLMF}}(n)}{\partial \boldsymbol{w}(n)} = & \|\boldsymbol{w}\|^3 - 3\left({\tilde{\boldsymbol{R}}_{s^4}}^{-1}\tilde{\boldsymbol{P}}_{d,s^3}\right)\|\boldsymbol{w}\|^2 \\ & + 3{\tilde{\boldsymbol{R}}_{s^4}}^{-1}\left(\tilde{\boldsymbol{P}}_{d^2,s^2} + \frac{\gamma}{2}I\right)\|\boldsymbol{w}\| - {\tilde{\boldsymbol{R}}_{s^4}}^{-1}\tilde{\boldsymbol{P}}_{d^3,s}.\end{aligned} \tag{31}$$

2.6 Mean behavior of the adaptive weight vector for LFXLMF algorithm

Using the same block diagram used for the FXLMF algorithm in Fig. 1, the weight update equation for the LFXLMF algorithm is given by

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) - \frac{\mu}{4}\frac{\partial J_{\mathrm{LFXLMF}}(n)}{\partial \boldsymbol{w}(n)} = \left(1 - \mu\frac{\gamma}{2}\right)\boldsymbol{w}(n) + \mu\, e^3(n)\, x'(n), \tag{32}$$

where the instantaneous gradient can be approximated as follows:

$$\frac{\partial \hat{\boldsymbol{j}}_{\mathrm{LFXLMF}}(n)}{\partial \boldsymbol{w}(n)} \approx -4\, e^3(n)\sum_{i=1}^{\hat{M}-1} \hat{s}_i \boldsymbol{x}^T(n-i) + 2\gamma \boldsymbol{w}(n). \tag{33}$$

Since we do not have the exact knowledge of the secondary path, we can substitute Eqs. (2)–(4) and (32) in Eq. (31) to get the adaptive weight vector update expression as follows:

$$\begin{aligned}\boldsymbol{w}(n+1) = & \left(1 - \mu\frac{\gamma}{2}\right)\boldsymbol{w}(n) + \mu\left(\sum_{i=0}^{\hat{M}-1} \hat{s}_i\, z^3(n)x(n-i)\right) \\ & + \mu\sum_{i=0}^{\hat{M}-1} \hat{s}_i\, d^3(n)\boldsymbol{x}(n-i) - 3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1} s_i\hat{s}_j\, d^2(n)\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i)\right) \\ & + 3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{\hat{M}-1} s_i s_j \hat{s}_k\, d(n)\boldsymbol{x}(n-k)\boldsymbol{x}^T(n-j)\,\boldsymbol{x}(n-i)\boldsymbol{w}(n-j)\,\boldsymbol{w}^T(n-i)\right) \\ & - 6\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1} s_i\hat{s}_j\, z(n)d(n)\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i)\right) \\ & + 3\,\mu\left(\sum_{i=0}^{\hat{M}-1} \hat{s}_i\, d^2(n)z(n)\boldsymbol{x}(n-i)\right) \\ & - \mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}\sum_{l=0}^{\hat{M}-1} s_i s_j s_k \hat{s}_l\, \boldsymbol{x}(n-l)\boldsymbol{x}^T(n-k)\,\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\right. \\ & \qquad \left.\boldsymbol{w}(n-k)\,\boldsymbol{w}^T(n-j)\boldsymbol{w}(n-i)\right) \\ & + 3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{\hat{M}-1} s_i s_j \hat{s}_k\, z(n)\boldsymbol{x}(n-k)\boldsymbol{x}^T(n-j)\,\boldsymbol{x}(n-i)\boldsymbol{w}(n-j)\,\boldsymbol{w}^T(n-i)\right) \\ & - 3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1} s_i\hat{s}_j\, z^2(n)\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)\boldsymbol{w}(n-i)\right) \\ & + 3\,\mu\left(\sum_{i=0}^{\hat{M}-1} \hat{s}_i\, d(n)z^2(n)\boldsymbol{x}(n-i)\right).\end{aligned} \tag{34}$$

Following assumptions *A1–A4*, the mean weight of the adaptive weight vector for LFXLMF algorithm is expressed as in the following:

$$\begin{aligned}E[\boldsymbol{w}(n+1)] = &\left(1-\mu\frac{\gamma}{2}\right)E\left[\boldsymbol{w}\left(n\right)\right]\\ &+\mu\sum_{i=0}^{\hat{M}-1}\hat{s}_i\,\boldsymbol{P}_{d^3,i}-3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1}s_i\hat{s}_j\;\boldsymbol{P}_{d^2,i,j}E[\boldsymbol{w}\left(n-i\right)]\right)\\ &+3\,\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{\hat{M}-1}s_is_j\hat{s}_k\,\boldsymbol{P}_{d,i,j,k}E[\boldsymbol{w}(n-j)\boldsymbol{w}^T(n-i)]\right)\\ &-\mu\left(\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}\sum_{l=0}^{\hat{M}-1}s_is_js_k\hat{s}_l\,\boldsymbol{R}_{\boldsymbol{i,j,k,l}}E[\boldsymbol{w}(n-k)\boldsymbol{w}^T(n-j)\boldsymbol{w}(n-i)]\right)\\ &-3\,\mu\,\sigma_z{}^2\left(\sum_{i=0}^{M-1}\sum_{j=0}^{\hat{M}-1}s_i\hat{s}_j\,\boldsymbol{R}_{\boldsymbol{i,j}}E[\boldsymbol{w}\left(n-i\right)]\right)+3\,\mu\,\sigma_z{}^2\left(\sum_{i=0}^{\hat{M}-1}\hat{s}_i\,\boldsymbol{P}_{d,i}\right).\end{aligned} \tag{35}$$

The mean weight of the adaptive weight vector for LFXLMF algorithm considering the independence theory looks like the following:

$$\begin{aligned}\mathrm{E}[\boldsymbol{w}(n+1)] = &\left(1-\mu\frac{\gamma}{2}\right)E[\boldsymbol{w}(n)]+\mu\,\hat{s}_0\,E\left[d^3(n)\boldsymbol{x}^{\boldsymbol{T}}(n)\right]\\ &-3\,\mu\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_i\hat{s}_i\,\boldsymbol{P}_{d^2,i,j}E[\boldsymbol{w}\left(n-i\right)]\right)\\ &+3\,\mu\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_is_i\hat{s}_i\,\boldsymbol{P}_{d,i,j,k}E[\boldsymbol{w}(n-i)\boldsymbol{w}^T(n-i)]\right)\\ &-\mu\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_is_is_i\hat{s}_i\boldsymbol{R}_{\boldsymbol{i,j,k,l}}\,E[\boldsymbol{w}(n-i)\boldsymbol{w}^T(n-i)\boldsymbol{w}(n-i)]\right)\\ &-3\,\mu\sigma_z{}^2\left(\sum_{i=0}^{\min\left(\hat{M},M\right)-1}s_i\hat{s}_i\,\boldsymbol{R}_{\boldsymbol{i,j}}E[\boldsymbol{w}\left(n-i\right)]\right)\\ &+3\,\mu\,\sigma_z{}^2(\hat{s}_0\,E[d(n)\boldsymbol{x}(n)]).\end{aligned} \tag{36}$$

2.7 Second moment analysis for LFXLMF

The performance analysis for the mean square error, $E[e^2(n)]$., of the LFXLMF algorithm is carried out, where the error is updated according to Eq. (7). Therefore, the MSE for the LFXLMF algorithm is obtained as follows:

$$\begin{aligned}\mathrm{MSE}_{\mathrm{LFXLMF}}(n) &= E\left[e^2(n)\right]\\ &= E\left[d^2(n)\right]-2\sum_{i=0}^{M-1}s_i\,E[d(n)\boldsymbol{x}(n-i)]E[\boldsymbol{w}(n-i)]\\ &+\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}s_is_j\,E[\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)]E[\boldsymbol{w}\left(n-i\right)\boldsymbol{w}^T\left(n-i\right)]+E[z^2(n)].\\ &= \sigma_d{}^2-2\sum_{i=0}^{M-1}s_i\,\boldsymbol{P}_{d,i}\,E[\boldsymbol{w}(n-i)]\\ &+\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}s_is_j\,\boldsymbol{R}_{\boldsymbol{i,j}}\,E[\boldsymbol{w}\left(n-i\right)\boldsymbol{w}^T\left(n-i\right)]+\sigma_z{}^2.\end{aligned} \tag{37}$$

Following the steps used in deriving Eq. (19), here, we reach the same results for the MMSE of the LFXLMF as given by

$$\mathrm{MMSE}_{\mathrm{LFXLMF}} = \sigma_z{}^2. \tag{38}$$

2.8 LFXLMF algorithm stability

In the ensuing, the effect of the leakage factor γ on the stability of the LFXLMF algorithm is discussed. As was done in [21], the value of γ is determined by the filter designer using trial and error methodology. For this work, the range of the leakage factor can be found with respect to the step size μ. To do that, first, we start with the LFXLMF algorithm weight update:

$$\begin{aligned}\boldsymbol{w}(n+1) &= \left(1-\mu\frac{\gamma}{2}\right)\boldsymbol{w}(n)-\frac{\mu}{4}\;\frac{\partial\boldsymbol{J}_{\mathrm{LFXLMF}}(n)}{\partial\boldsymbol{w}(n)}\\ &= \left(1-\mu\frac{\gamma}{2}\right)\boldsymbol{w}(n)+\mu\,e^3(n)\,x'(n).\end{aligned} \tag{39}$$

The algorithm converges when $E[\boldsymbol{w}(n+1)] = E[\boldsymbol{w}(n)]$. In other words, the weight adjustment term will be zero, that is,

$$\begin{aligned}&\mu\,E\left[\,e^3(n)\,x'(n)\right]=0\\ &E\left[\left(d(n)-\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}+z(n)\right)^3\left(\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\right)\right]=0.\end{aligned} \tag{40}$$

But, since

$$\begin{aligned}e(n) &= d(n)-\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}+z(n)\\ &= \sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}_{\boldsymbol{o}}-\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{w}+z(n),\end{aligned} \tag{41}$$

and assuming fixed $\boldsymbol{w}$, then we can define the weight error vector

$$\boldsymbol{v}(n) = \boldsymbol{w}(n)-\boldsymbol{w}_{\boldsymbol{o}}, \tag{42}$$

Hence, Eq. (41) looks like the following:

$$e(n) = z(n)-\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{v}(n). \tag{43}$$

Using Eqs. (42) and (43) in Eq. (39), one obtains

$$\begin{aligned}\boldsymbol{v}(n+1) = &\left(1-\mu\frac{\gamma}{2}\right)\boldsymbol{v}(n)\\ &+\mu\left(z(n)-\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\boldsymbol{v}(n)\right)^3\left(\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\right).\end{aligned} \tag{44}$$

The value of $\boldsymbol{v}(n)$ approaches zero when the algorithm converges so that we can ignore the high-order terms of $\boldsymbol{v}(n)$, and as a result, the weight error update equation can be written as

$$\boldsymbol{v}(n+1)\cong\left(1-\mu\frac{\gamma}{2}\right)\boldsymbol{v}(n) + \mu\left(z^3(n)-3\sum_{i=0}^{M-1}s_i z^2(n)\boldsymbol{x}^T(n-i)\boldsymbol{v}(n)\right) \times\left(\sum_{i=0}^{M-1}s_i\boldsymbol{x}^T(n-i)\right). \quad (45)$$

To find the mean weight error, we need to take the expectation of Eq. (45), and relying on the assumptions *A1–A4*, the noise is independent of the input signal as well as the weight error vector. Consequently, the mean of the weight error vector is given by

$$E[\boldsymbol{v}(n+1)] = \left(1-\mu\frac{\gamma}{2}\right)E[\boldsymbol{v}(n)] -\mu\left\{3\sigma_z{}^2\sum_{i=0}^{M-1}\sum_{j=0}^{M-1}s_i E[\boldsymbol{x}(n-j)\boldsymbol{x}^T(n-i)]E[\boldsymbol{v}(n)]\right\}$$
$$\mathrm{E}[\boldsymbol{v}(n+1)] = \left(\left(1-\mu\frac{\gamma}{2}\right)I-\mu(3\sigma_z{}^2\boldsymbol{R}_{s^2})\right)E[\boldsymbol{v}(n)]. \quad (46)$$

Assuming a positive definite autocorrelation matrix, $\boldsymbol{R}_{i,j} > 0$, the range of the leakage factor γ for LFXLMF algorithm is given by

$$\frac{3\sigma_z{}^2\lambda_{\max}(\tilde{\boldsymbol{R}}_{s^2})-1}{\frac{3}{2}\sigma_z{}^2\lambda_{\max}(\tilde{\boldsymbol{R}}_{s^2})} < \gamma < \frac{2}{\mu} \quad (47)$$

where $\lambda_{\max}(\tilde{\boldsymbol{R}}_{s^2})$ represents the maximal eigenvalue of $\tilde{\boldsymbol{R}}_{s^2}$. As can be seen from Eq. (48), the leakage factor has an effect on the step size.

3 Algorithms' convex combination

In this section, we examine the behavior of our algorithm through the convex combination approach, namely the convex combination with the FXLMF algorithm. The method of combining two algorithms is an interesting proposal. It aims to mix the output of each filter and highlights the best features of each individual algorithm. Then, it utilizes the features in the overall equivalent filter to improve the performance of the adaptive filter [24–29]. In this section, we will examine our proposed algorithms with members from the LMS and LMF families.

Figure 2 is the proposed block diagram for the convex combination of two filtered input signals, where the output of the overall combined filter can be given as in [25] by the following equation:

$$y(n) = \lambda(n)y'_1(n) + [1-\lambda(n)]y'_2(n) \quad (48)$$

where $y'_1(n)$ and $y'_2(n)$ are the output of the two filters and $\lambda(n)$ is the contribution or mixing parameter, where $0 \le \lambda(n) \le 1$. This parameter shows the percentage of involvement for each algorithm in the overall filter output. Therefore, the combined filter will extract the best features for each filter $\boldsymbol{w}_1(n)$ and $\boldsymbol{w}_2(n)$ individually. Assuming both filters $\boldsymbol{w}_1(n)$ and $\boldsymbol{w}_2(n)$ have the same size M, then the weight vector of the overall filter can be given as

$$\boldsymbol{w}(n) = \lambda(n)\boldsymbol{w}_1(n) + [1-\lambda(n)]\boldsymbol{w}_2(n). \quad (49)$$

Each filter is updated individually, depending on its own error $e_1(n)$ or $e_2(n)$, and the overall weight vector is updated according to the total error $e(n) = [d(n) - y(n) + z(n)]$ which adapts the mixing parameter $\lambda(n)$. Using the gradient descent method, we can minimize the fourth-order $e^4(n)$ and the second-order $e^2(n)$ errors for the overall filter. Based on that, we can use the convex combined filter over two scenarios.

In the first scenario, we will do the minimization for the quadratic error $e^2(n)$, where $\lambda(n)$ is the sigmoidal function given as

$$\lambda(n) = \frac{1}{1+e^{-a(n)}}, \quad (50)$$

and instead of doing the update equation with respect to $\lambda(n)$, we will define the update equation with respect to the changing value $a(n)$ as follows:

$$\begin{aligned} a(n+1) &= a(n)-\frac{\mu_{a^2}}{2}\frac{\partial e^2(n)}{\partial a(n)} \\ &= a(n)-\frac{\mu_{a^2}}{2}\frac{\partial e^2(n)}{\partial\lambda(n)}\frac{\partial\lambda(n)}{\partial a(n)} \\ &= a(n)+\mu_{a^2}e(n)[y'_1(n)-y'_2(n)]\lambda(n)[1-\lambda(n)]. \end{aligned} \quad (51)$$

The second scenario is to conduct the minimization for the fourth-order error of the overall filter; then, the updated equation with respect to $a(n)$ will be as the following:

$$\begin{aligned} a(n+1) &= a(n)-\frac{\mu_{a^4}}{4}\frac{\partial e^4(n)}{\partial a(n)} \\ &= a(n)-\frac{\mu_{a^4}}{4}\frac{\partial e^4(n)}{\partial\lambda(n)}\frac{\partial\lambda(n)}{\partial a(n)} \\ &= a(n)+\mu_{a^4}e^3(n)[y'_1(n)-y'_2(n)]\lambda(n)[1-\lambda(n)], \end{aligned} \quad (52)$$

where μ_{a^2} and μ_{a^4} are the step sizes for the overall filter, for the quadratic and fourth-order errors, respectively. In this work, we study the mean square performance for the convex-combined filter using the filtered input signal. Since the range of $\lambda(n)$ is between zero and one, we need to insure that the combined filter keeps adapting and does not stick with only one algorithm all the time. For this purpose, we have to reduce the interval of the mixing parameter by limiting the value of $a(n)$ inside $[1-a^+, a^+]$; then, the range of the mixing parameter will be between $1-\lambda^+ \le \lambda(n) \le \lambda^+$, as the following:

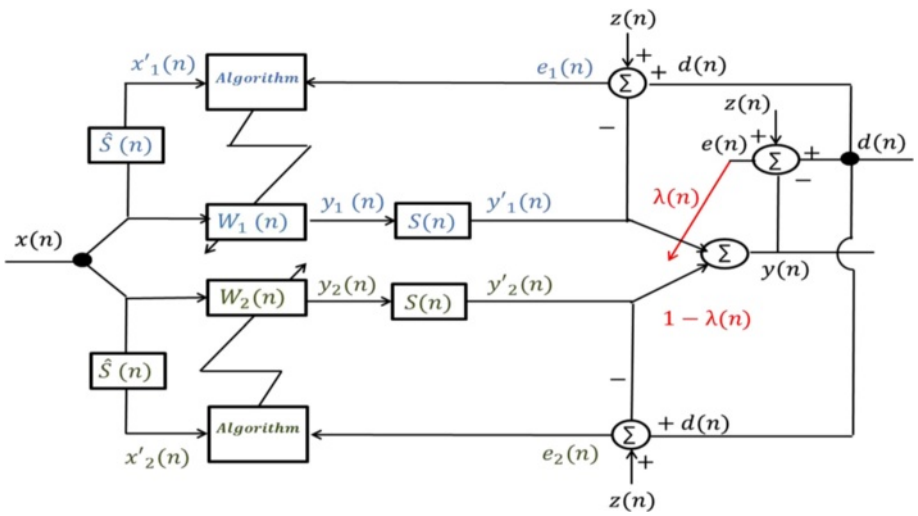

Fig. 2 Block diagram of adaptive convex combination for two filtered input signal algorithms

$$\lambda(n) = \begin{cases} 0.998, & a(n) > a^{+} \\ \lambda(n), & a^{+} \geq a(n) \geq -a^{+} \\ 0.002, & a(n) < -a^{+} \end{cases} \quad (53)$$

Simulations in Section 4 will investigate four cases, where the comparison will be done by using the FXLMF and FXLMS algorithms, as the two transversal filters are used in the convex combination, according to the second error order minimization.

4 Simulation results

Simulations in this section are divided into two parts. The first part examines the proposed algorithms in the mean square error and mean weight context. The simulation has been done for the FXLMF and LFXLMF algorithms under some conditions and environments. While in the second part, we test the concept of convex combinations over the FXLMF and FXLMS algorithms. Furthermore, comparisons with other algorithms are carried

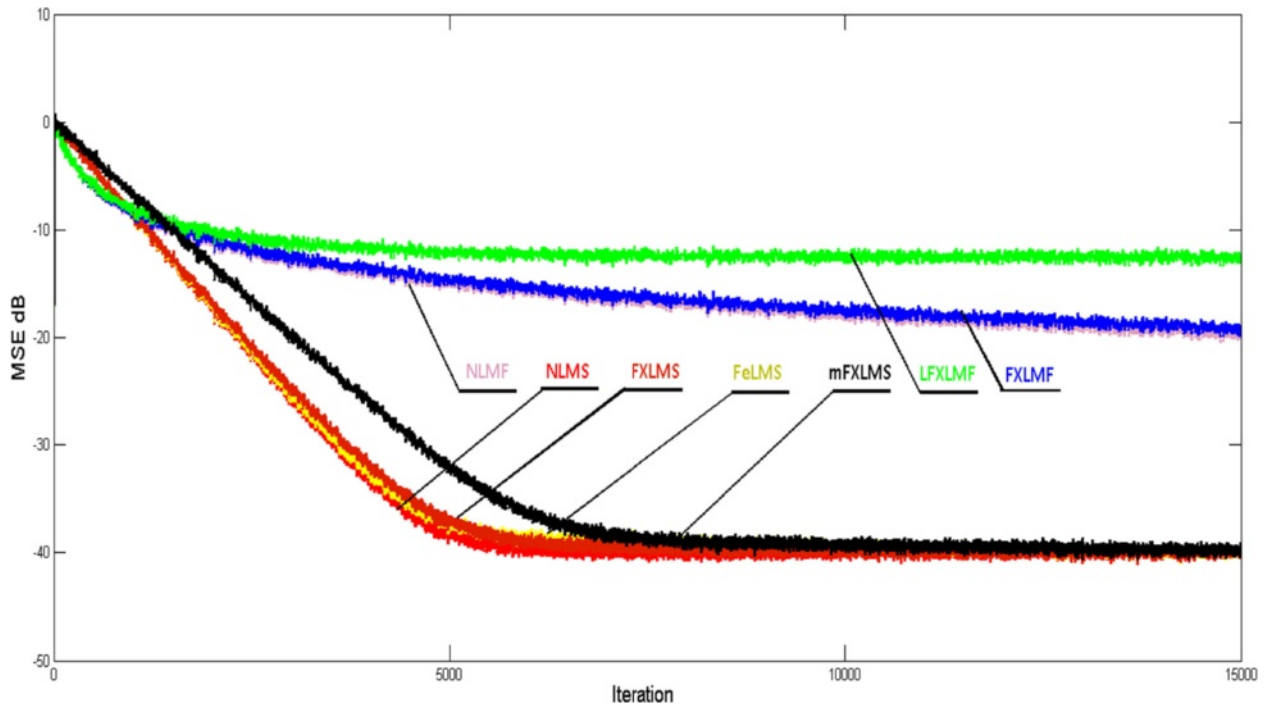

Fig. 3 Comparison over MSE for FXLMF and LFXLMF with other algorithms using fixed step size $\mu = 0.001$ and high SNR = 40 dB and Gaussian noise, leakage factor $\gamma = 0.05$

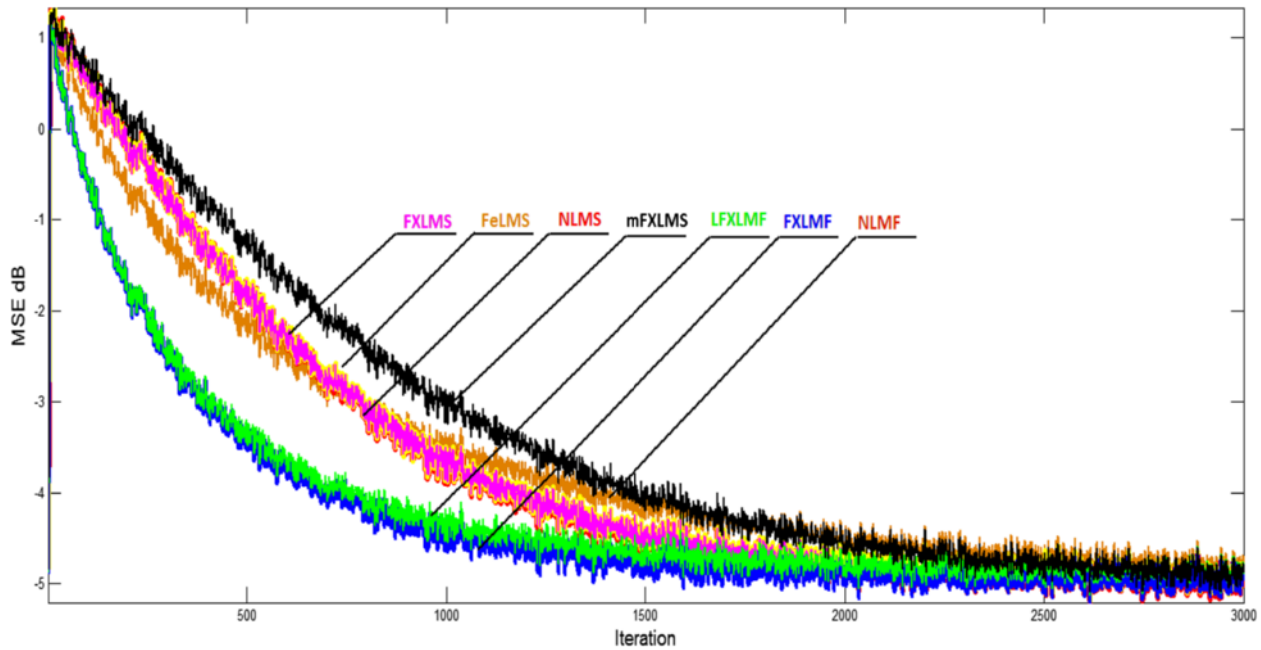

Fig. 4 Comparison over MSE for FXLMF and LFXLMF algorithms with other algorithms using fixed step size $\mu = 0.001$ and low SNR = 5 dB and Gaussian noise, leakage factor $\gamma = 0.05$

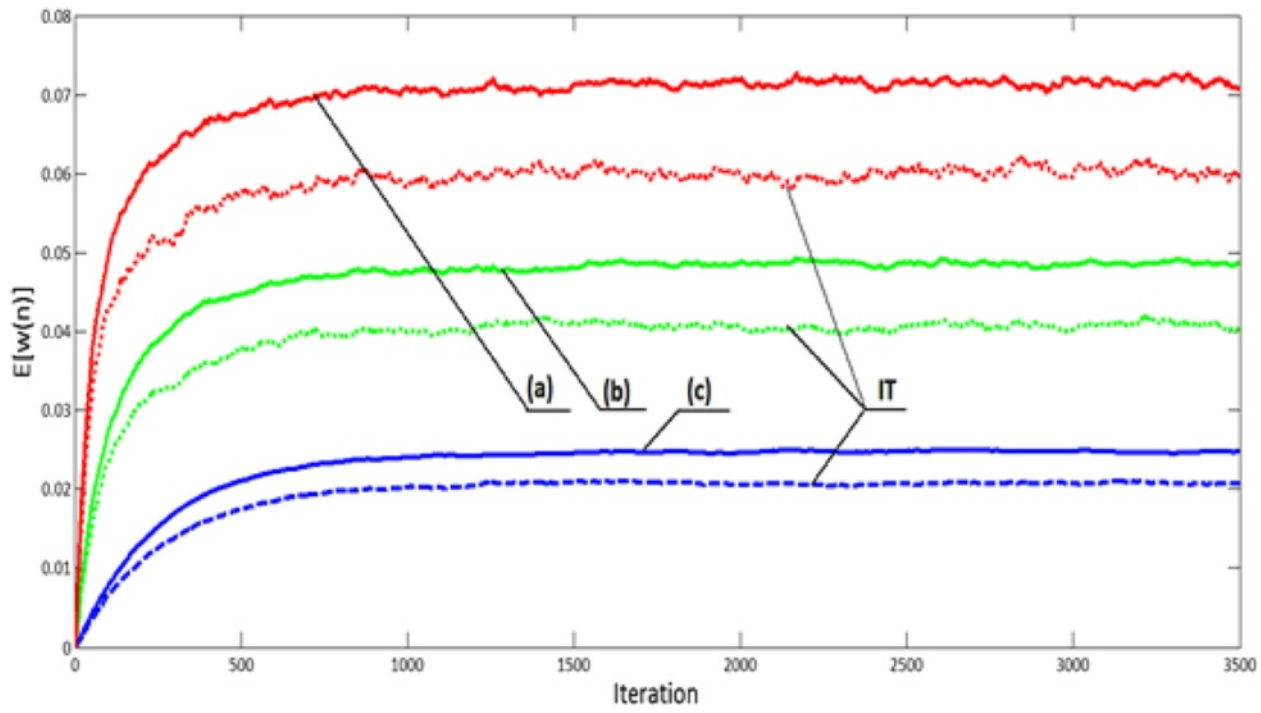

Fig. 5 Comparison over mean weight vector for FXLMF algorithms using different step sizes $\mu = [red = 0.001, green = 0.0005, blue = 0.0001]$ using Gaussian noise at low SNR = 5 dB. *Solid line*: proposed models (*a*), (*b*), and (*c*). *Dashed line*: IT model

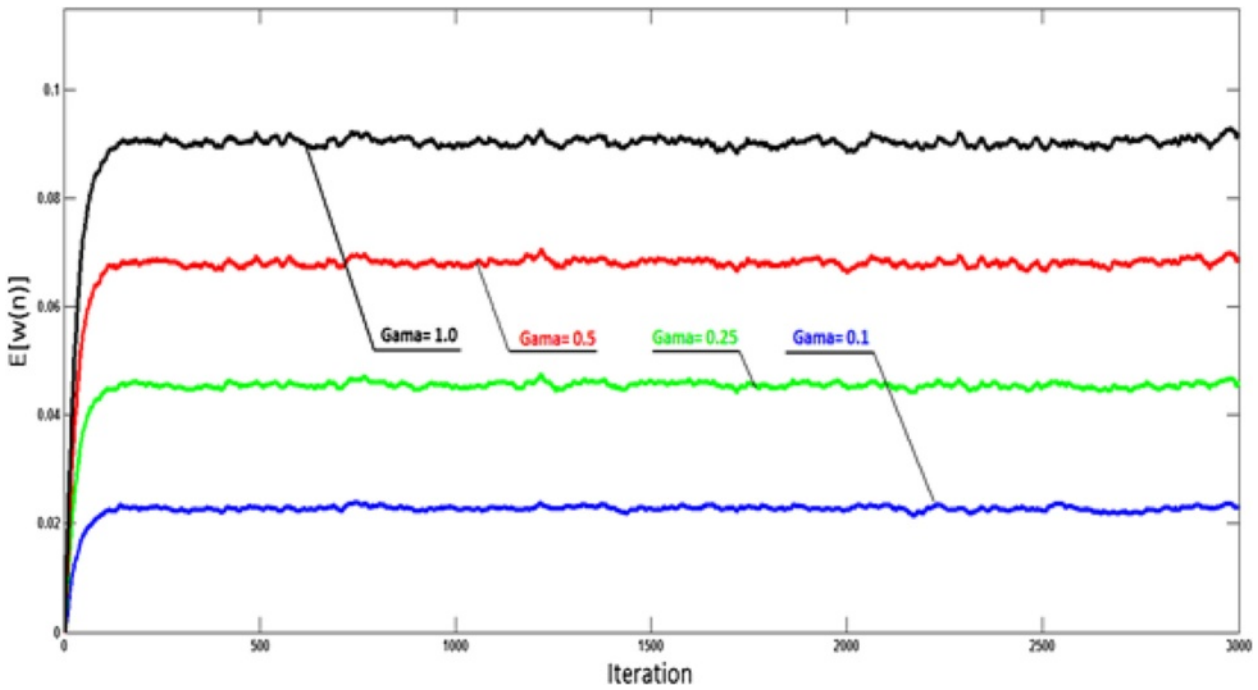

Fig. 6 Comparison over mean weight vector for LFXLMF algorithms using different leakage factors $\gamma = [0.1, 0.250, 0.50, 1]$ and fixed step size $\mu = 0.001$ using Gaussian noise at low SNR = 5 dB

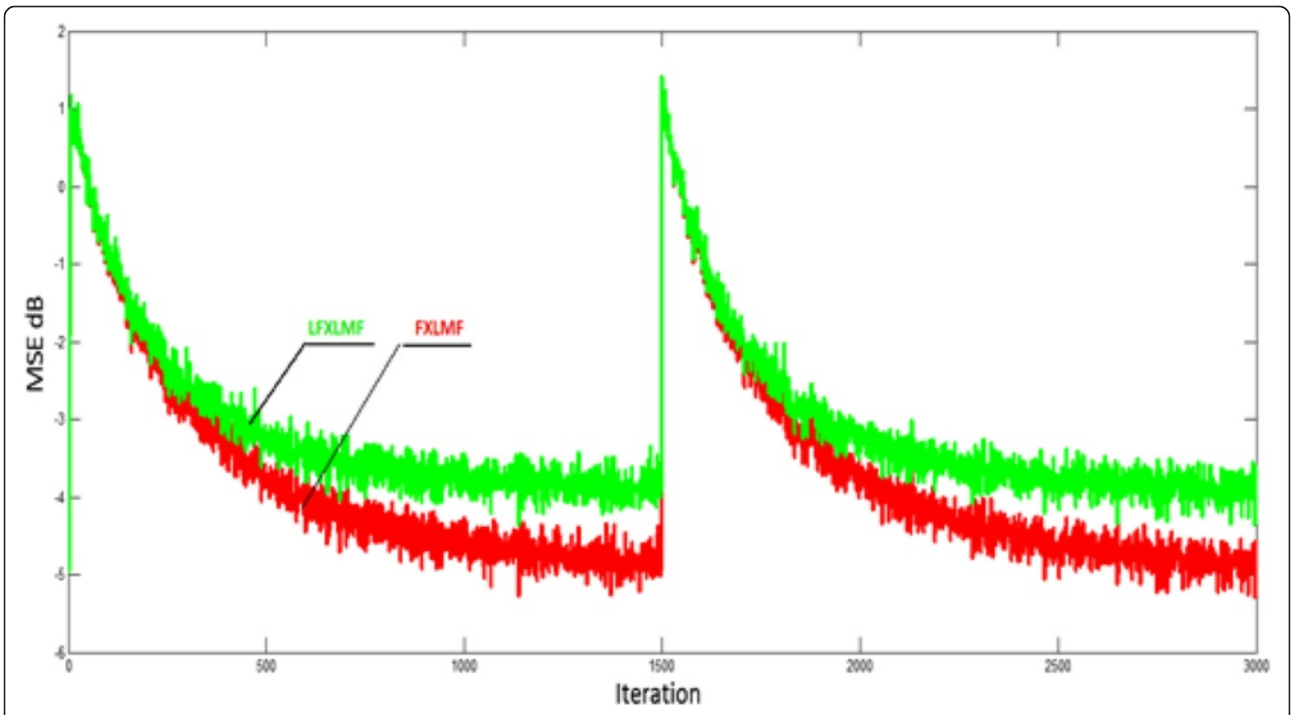

Fig. 7 MSE for the FXLMF and LFXLMF algorithm robustness using Gaussian noise at low SNR = 5 dB, fixed step size $\mu = 0.00125$, and leakage factor $\gamma = 0.50$

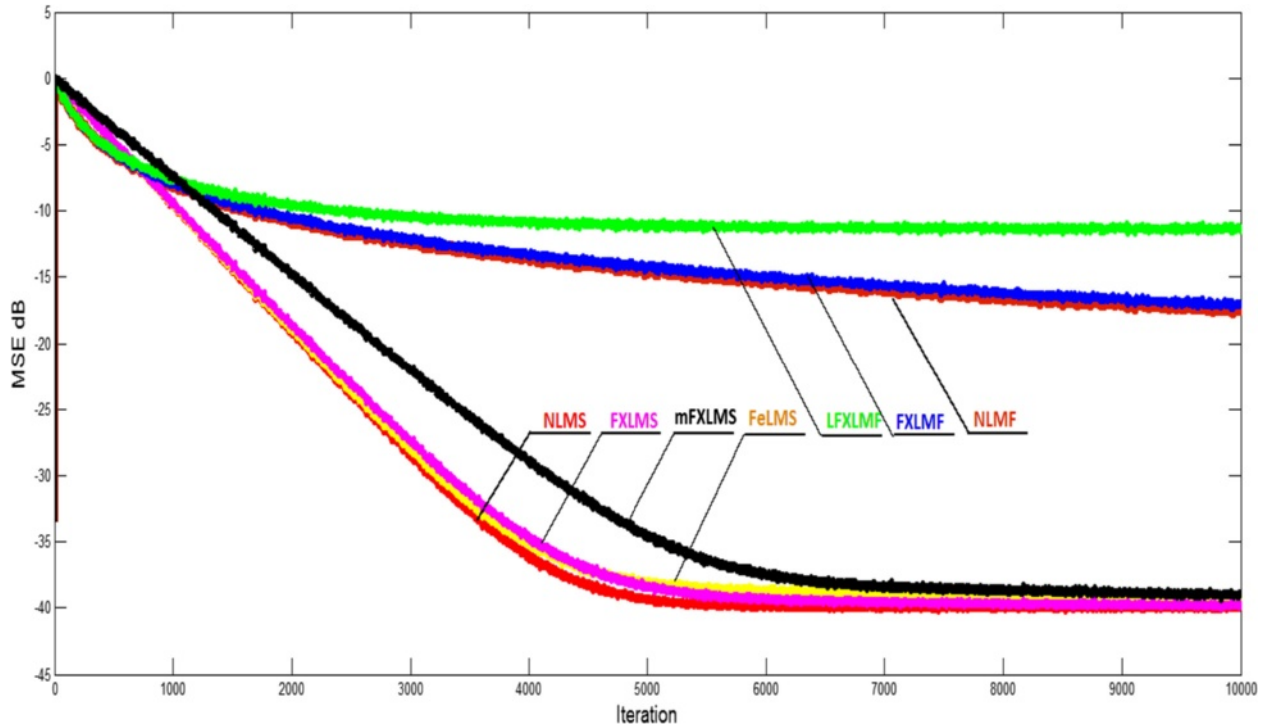

Fig. 8 Comparison over MSE for FXLMF and LFXLMF with other algorithms using a fixed step size $\mu = 0.001$ using uniform noise at high SNR = 40 dB and leakage factor $\gamma = 0.05$

out to show under which circumstances the new proposed algorithms outperform algorithms from the LMS family, in convergence. The plant vector used to filter the input signal is wp with nine taps where

$$wp = [\,0.0179 \quad 0.1005 \quad 0.2795 \quad 0.4896 \quad 0.5860 \quad 0.4896 \quad 0.2795 \quad 0.1005 \quad 0.0179].$$

In addition, for simplicity, we assume the secondary path and the estimated secondary path are equal $\boldsymbol{S} = \hat{\boldsymbol{S}} = [\,0.7756 \quad 0.5171 \quad -0.3620\,]$.

The result for all simulations are the average of 500 Monte Carlo simulations, the noise is white Gaussian for Figs. 3, 4, 5, 6, and 7 and uniform noise for Figs. 8, 9, 10, 11, and 12.

Next, Figs. 13, 14, 15, 16, 17, 18, and 19 are for the convex combination; we used both of the transversal filters to have the same adaptive algorithm but with different step sizes. Then, we did a comparison between the FXLMF and FXLMS algorithms at low and high SNR for white Gaussian noise. All previous simulations were done using the minimization for quadratic error equation.

Figure 3 shows a comparison of the mean square error MSE behavior for different algorithms from the LMS family (i.e., NLMS, FXLMS, FeLMS, MFXLMS), the NLMF, and our proposed ones. It can be shown that the FXLMF algorithm converges, and it will reach the white noise level after a large number of iterations. For the LFXLMF algorithm, it reaches the steady state level faster than the others and after almost 5000 iterations, but it converges to a higher white noise level at almost 12 dB. Using a larger step size μ may lead the algorithm to diverge.

Figure 4 shows a comparison of the mean square error MSE behavior for different algorithms with fixed step size but this time for low SNR with a value of 5 dB. We can clearly notice that the FXLMF and LFXLMF algorithms outperform other LMS family algorithms in speed of convergence, an advantage to our proposed algorithms with almost 500 iterations. FXLMF and LFXLMF almost have identical curves because we are using a small leakage factor γ.

Figure 5 shows the effect of changing the step size on the mean weight vector of the FXLMF algorithm; when we increase the values of the step size, the algorithm converges faster to the larger mean of the weight. Moreover, using assumption *A4* makes the algorithm converge to a higher mean weight level.

Figure 6 shows the effect of changing the leakage factor on the mean weight of the LFXLMF algorithm. We can see that increasing the value of the leakage factor will increase the mean weight of the LFXLMF algorithm and it does not affect the speed of convergence.

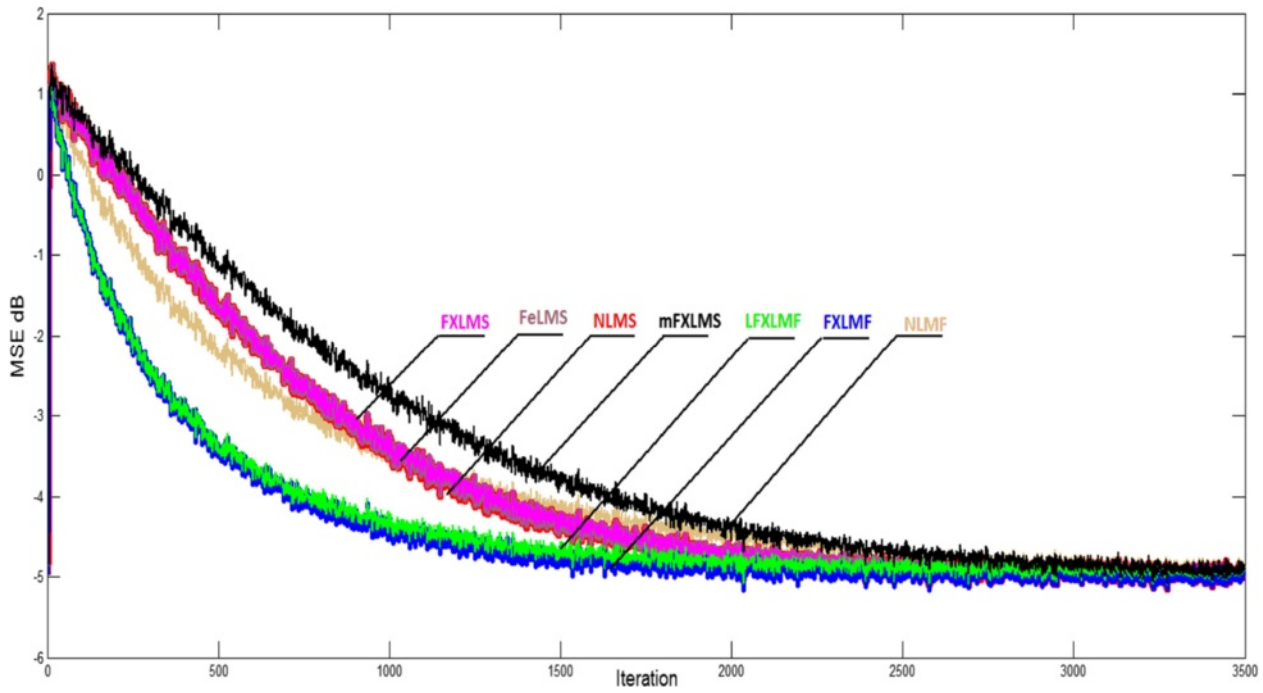

Fig. 9 Comparison over MSE for the FXLMF and LFXLMF algorithms with other algorithms using a fixed step size $\mu = 0.001$ and uniform noise at low SNR = 5 dB and leakage factor $\gamma = 0.05$

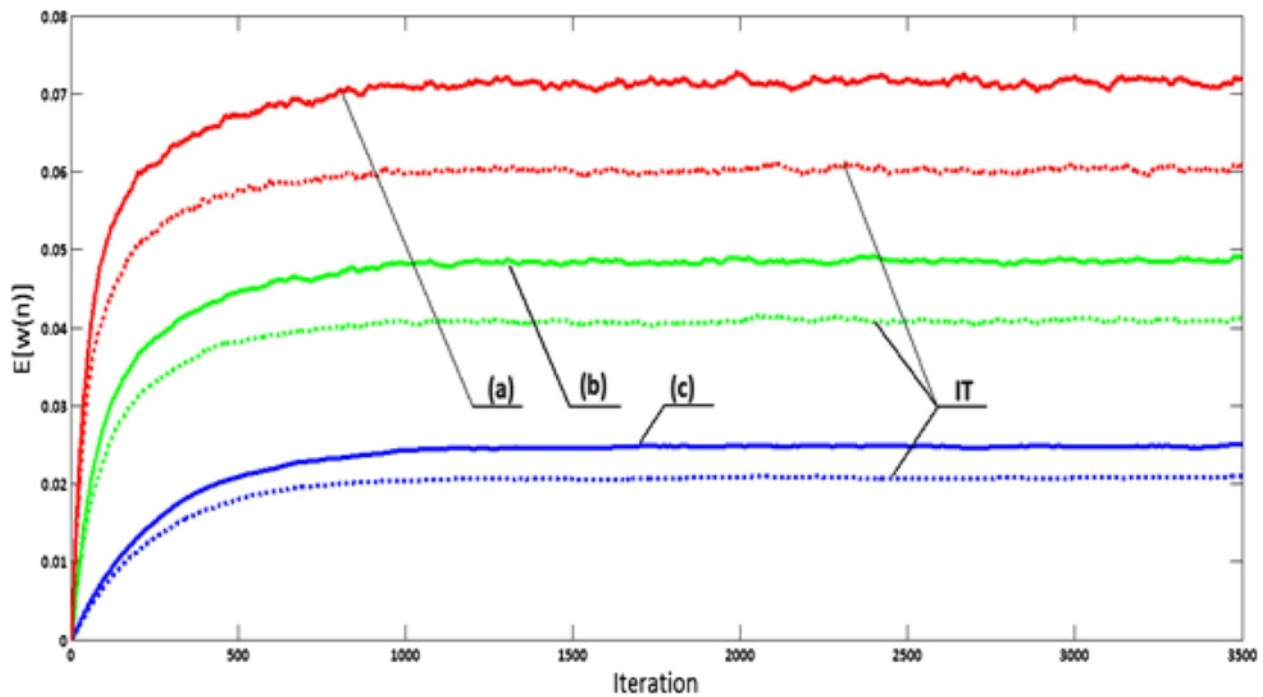

Fig. 10 Comparison over mean weight vector for FXLMF algorithms using different step sizes $\mu = [red = 0.001, green = 0.0005, blue = 0.0001]$ using uniform noise at low SNR = 5 dB. *Solid line*: proposed models (*a*), (*b*), and (*c*). *Dashed line*: IT model

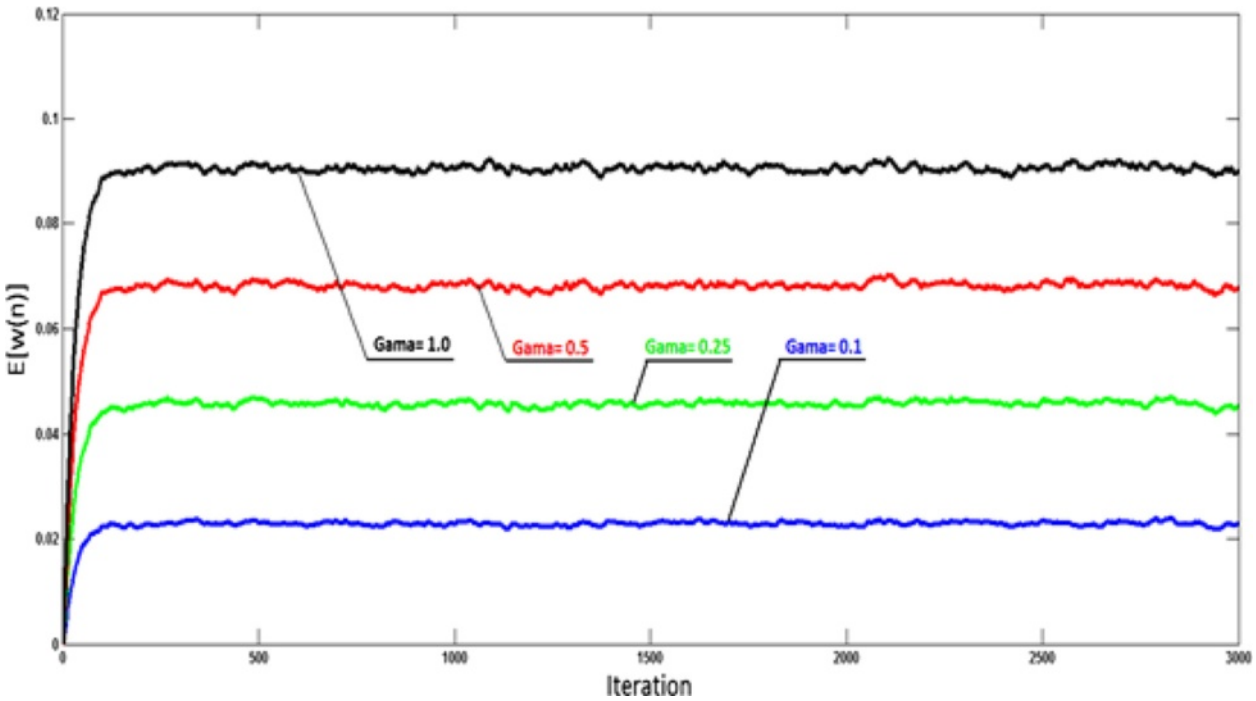

Fig. 11 Comparison over mean weight vector for LFXLMF algorithms using different leakage factors $\gamma = [0.1, 0.250, 0.50, 1]$ and fixed step size $\mu = 0.001$ using uniform noise at low SNR = 5 dB

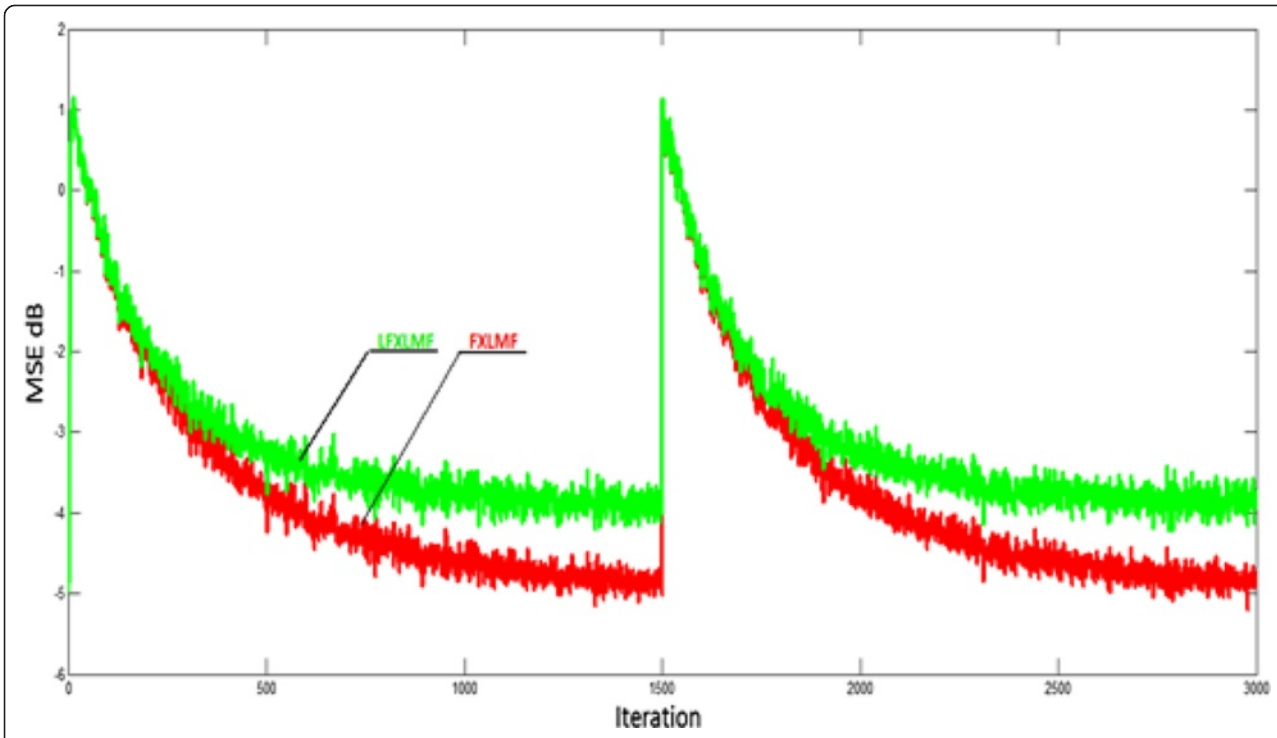

Fig. 12 MSE for the FXLMF and LFXLMF algorithm robustness using uniform noise at low SNR = 5 dB, fixed step size $\mu = 0.00125$, and leakage factor $\gamma = 0.50$

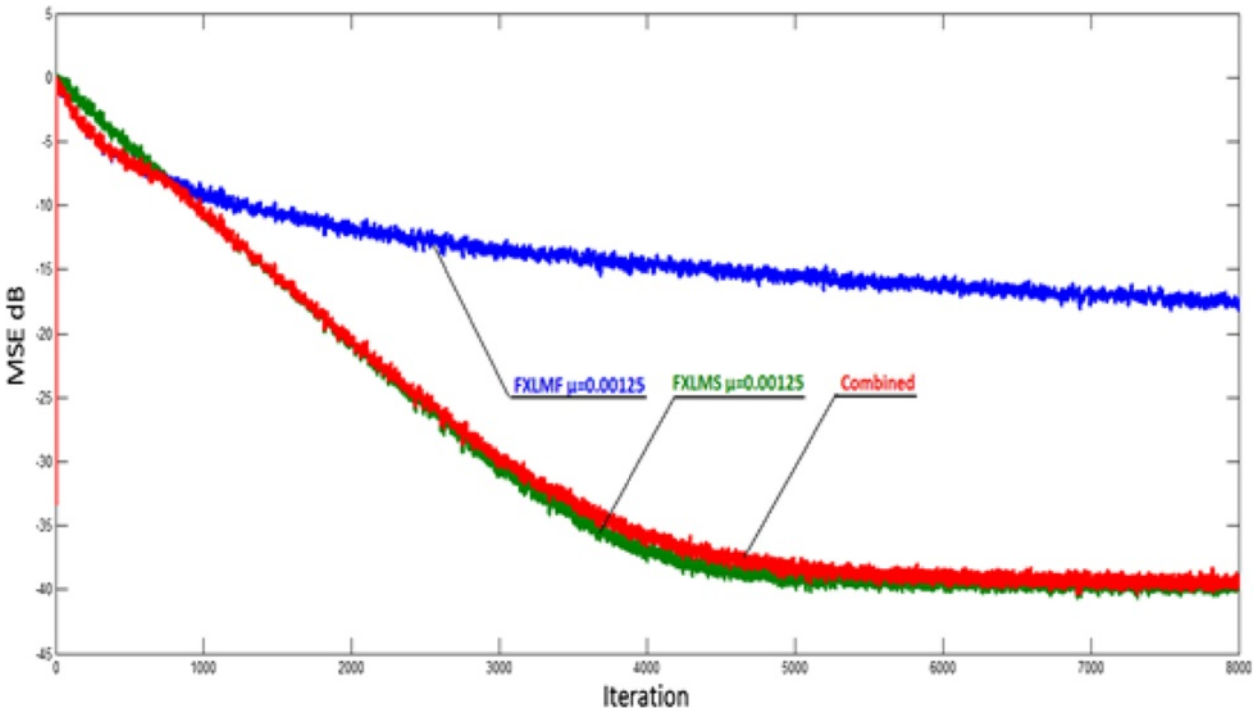

Fig. 13 MSE for combined FXLMF and FXLMS using Gaussian noise at high SNR = 40 dB and fixed step size $\mu = 0.00125$

Figure 7 shows the robustness of the proposed algorithms FXLMF and LFXLMF at low SNR and using Gaussian noise, when a sudden change occurred in the weight vector.

Figure 8 reports the performance of the algorithms when the uniform noise ids used instead of Gaussian, using the same conditions as we used before in Fig. 3. As we can see, we have almost the same result, since both the FXLMF and LFXLMF algorithms converge, where the first one keeps converging while the second one reaches the steady state faster.

Figure 9 is the same as Fig. 4 but using a fixed step size and uniform noise. In addition, the FXLMF and LFXLMF algorithms outperform the LMS family in convergence.

Figure 10 shows the effect of changing the step size on the mean weight vector of the FXLMF algorithm, and as

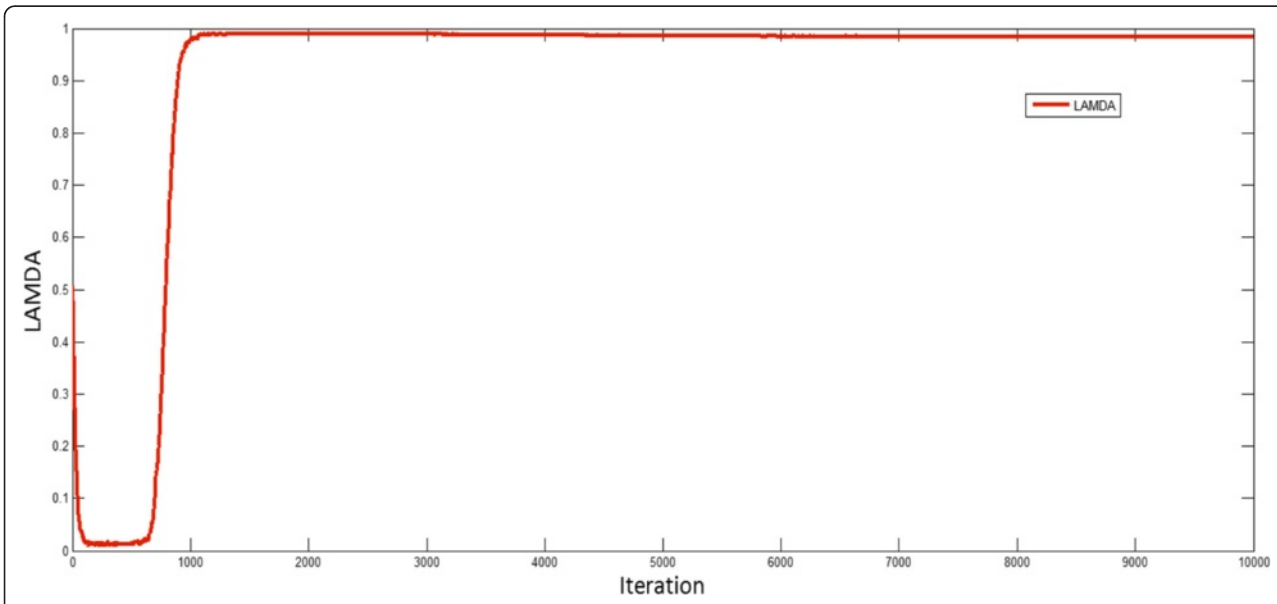

Fig. 14 Values of the mixing parameter $\lambda(n)$ for Fig. 13

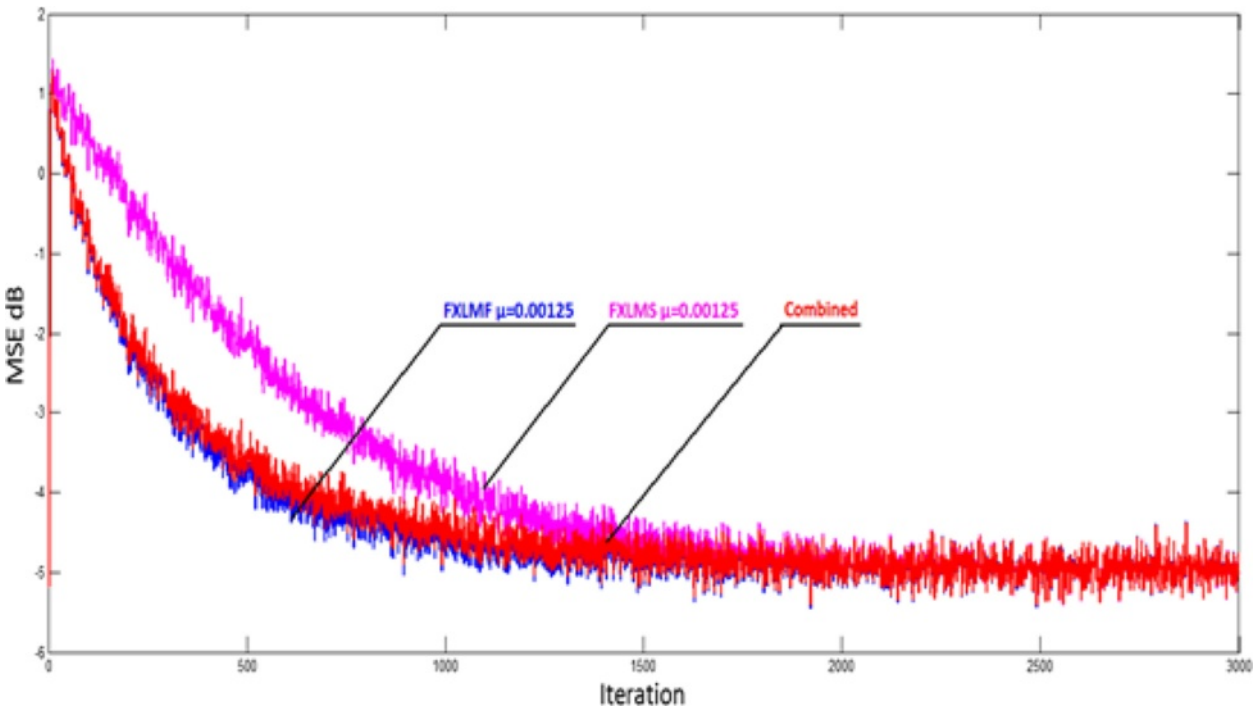

Fig. 15 MSE for combined FXLMF and FXLMS using Gaussian noise at low SNR = 5 dB and fixed step size $\mu = 0.00125$

depicted in Fig. 5, the algorithm converges faster as we increase the step size using uniform noise.

Figure 11 shows the effect of changing the leakage factor on the mean weight of the LFXLMF algorithm as shown in Fig. 6; increasing the value of the leakage factor will increase the mean weight of the LFXLMF algorithm.

Figure 12 shows the robustness of the proposed algorithms FXLMF and LFXLMF at low SNR and using uniform noise.

Figure 13 illustrates the behavior of the convex-combined filter of FXLMS and FXLMF algorithms; we can see at the beginning that the combined filter followed the FXLMF algorithm since it has a faster

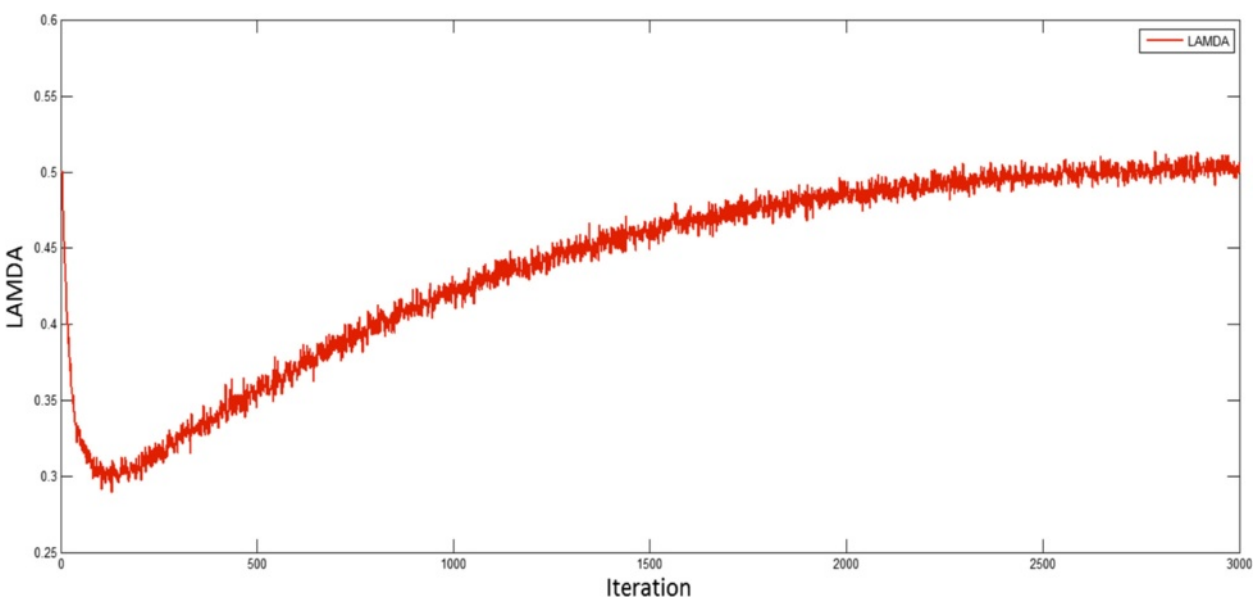

Fig. 16 Values of the mixing parameter for Fig. 15

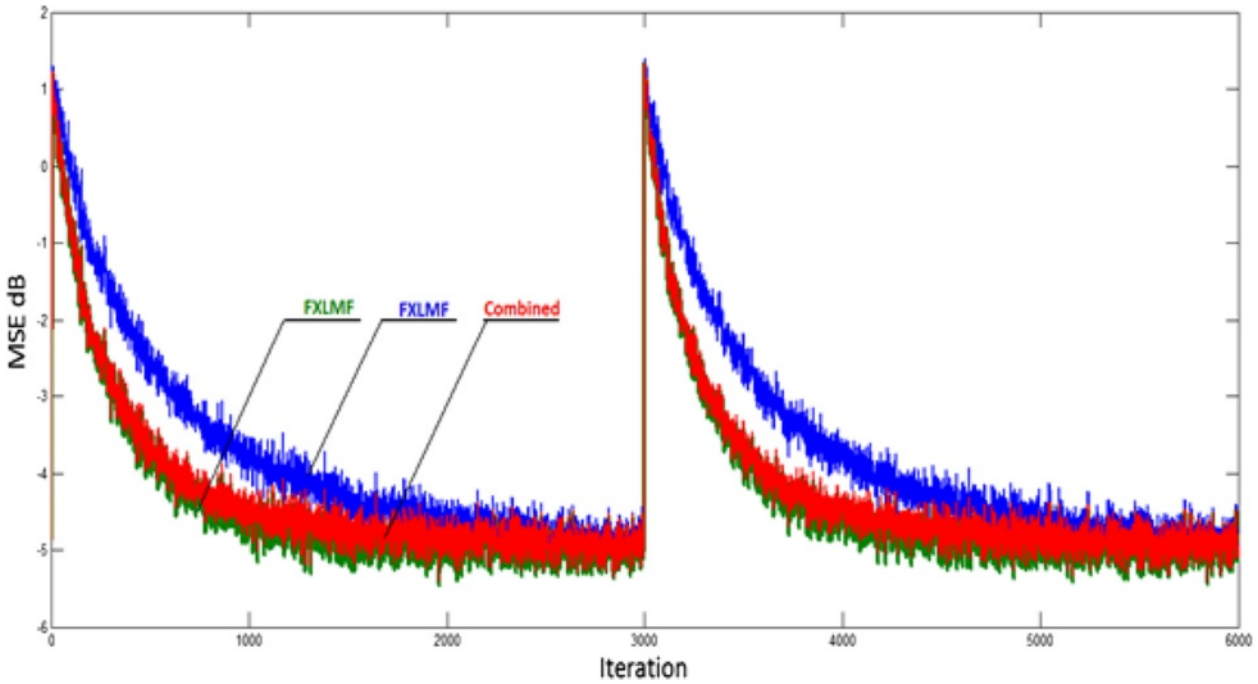

Fig. 17 MSE for combined FXLMF algorithm robustness using Gaussian noise at low SNR = 5 dB and fixed step sizes $\mu = 0.00125$ (*green*) and 0.000625 (*blue*)

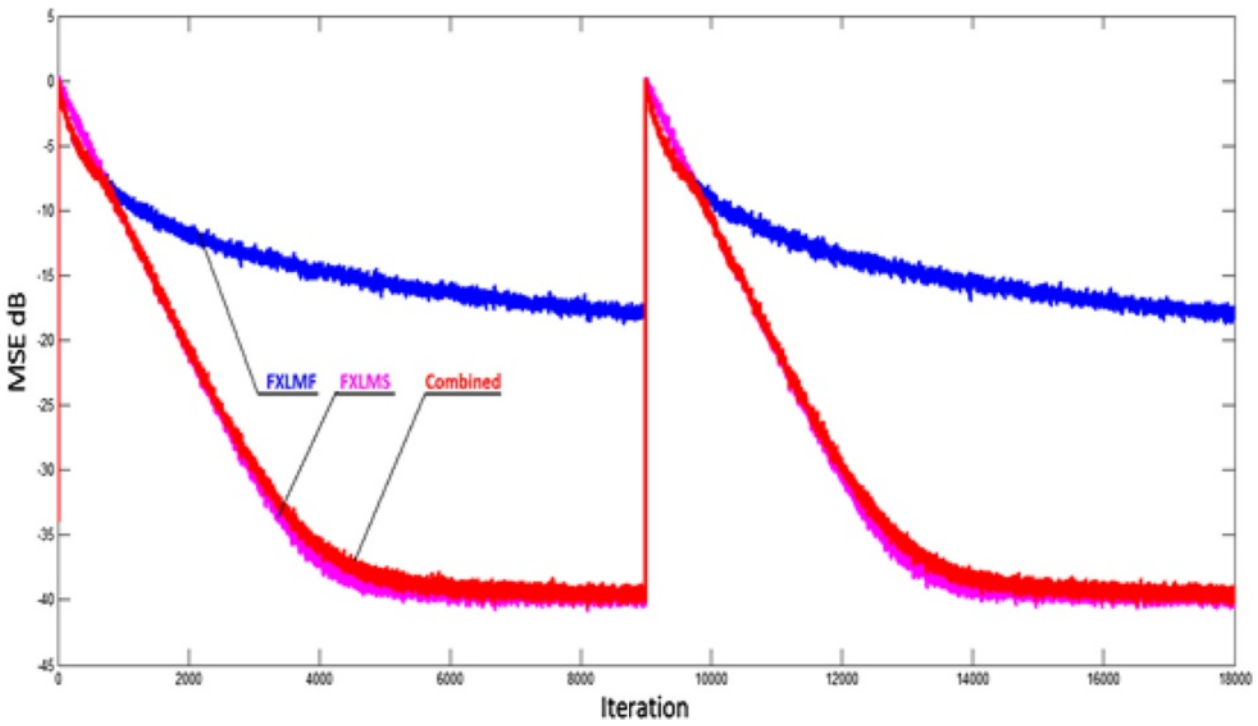

Fig. 18 MSE for the combined FXLMF and FXLMS algorithm robustness test using Gaussian noise at high SNR = 40 dB and fixed step size $\mu = 0.00125$

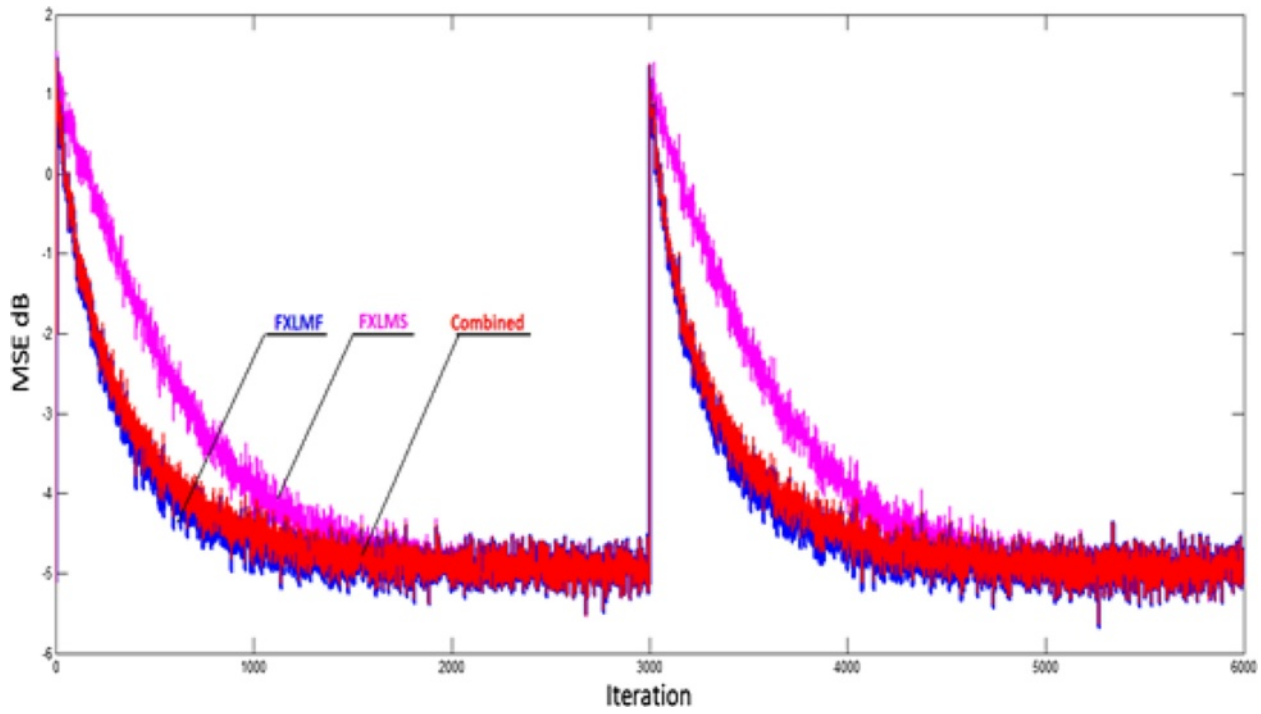

Fig. 19 MSE for combined FXLMF and FXLMS algorithm robustness test using Gaussian noise at low SNR = 5 dB and fixed step size $\mu = 0.00125$

speed of convergence. After that, the combined filter moved to the FXLMS algorithm, which showed better convergence at high SNR. Also, we can see from Fig. 14 the behavior of the mixing parameter $\lambda(n)$. We assume a 50 % mixing percentage as the initial case, then $\lambda(n)$ followed the FXLMF algorithm at the beginning where the FXLMF shows faster convergence, and after that, the mixing parameter switched to the other algorithm FXLMS where it has a better convergence. In Fig. 15, with the same environment as in Fig. 13 but with low SNR, the FXLMF algorithm outperforms the FXLMS algorithm and the combined filter followed the FXLMF algorithm at the beginning; then, when both algorithms have the same convergence, the

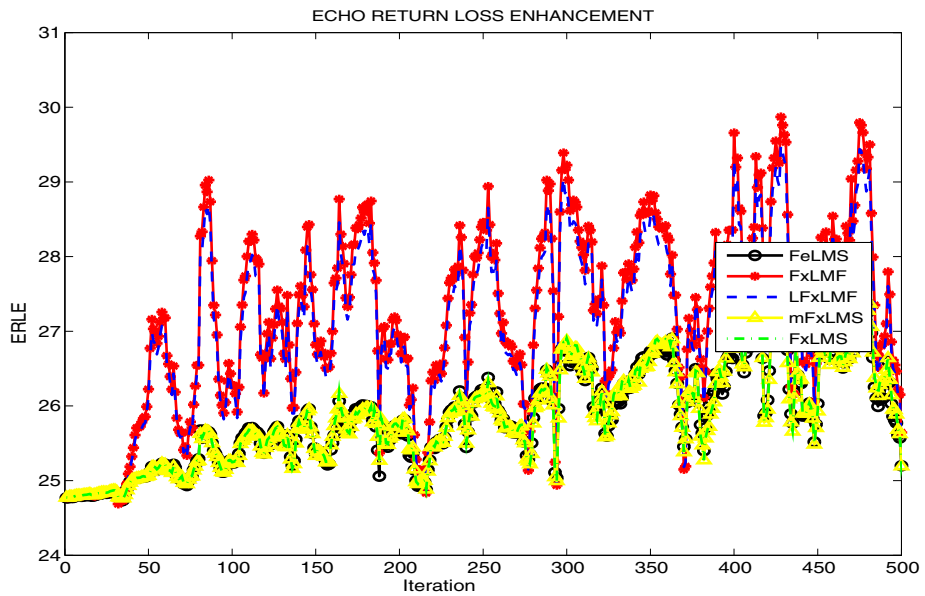

Fig. 20 ERLE performance of the FXLMF and LFXLMF algorithms

mixing parameter is $\lambda(n) = 50$ %. This is shown in Fig. 16.

Figure 17 shows the robustness of the convex-combined filter of FXLMF for two different step sizes at low SNR and using Gaussian noise. We can clearly see that the combined filter followed the one with a larger step size, which already shows better performance. Similarly, Fig. 18 shows the robustness of the convex-combined filter of the FXLMF and FXLMS algorithms at high SNR and using Gaussian noise. We can clearly see that the combined filter followed the FXLMF algorithm at the beginning and then switched to the FXLMS algorithm, which shows better performance at high SNR. Finally, Fig. 19 shows the robustness of the convex-combined filter of the FXLMF and FXLMS algorithms at low SNR and using Gaussian noise. We can clearly see that the combined filter followed the FXLMF algorithm all the time since it shows better performance than the FXLMS algorithm at low SNR.

Finally, the performance of the proposed algorithms is tested using the echo return loss enhancement (ERLE) metric. As can be depicted from Fig. 20, our proposed algorithms outperform the rest of the algorithms.

5 Conclusions

Two algorithms FXLMF and LFXLMF were proposed in this work; an analytical study and mathematical derivations for the mean weight adaptive vector and the mean square error for both algorithms have been obtained. Moreover, the step size and the leakage factor bound ranges were investigated.

From the literature, we received a good sense about proposing new algorithms to the LMF family, as was proposed before in the LMS. The FXLMF and LFXLMF algorithms successfully converge under a large range of SNR. Furthermore, we see the ability of both algorithms to converge under different environments of noise: Gaussian and uniform. However, the LMF family requires more computational complexity; our proposed algorithms were faster in convergence than members of the LMS family under some circumstances.

From the simulations, we saw that both algorithms converge well under relatively high SNR but they converge faster under low SNR. In addition, using a step size near the upper boundary will guarantee less time to converge; however, working close to the upper boundary of the step size ensures faster convergence, but we have to take the risk of algorithm divergence. Also, we see that a larger step size will increase the mean of the weight vector. Step size under the upper boundary is given in Eq. (28).

The leakage factor in the LFXLMF algorithm adds more stability to the algorithm, at the expense of reduced performance as was expected from the literature. The leakage factor boundaries were derived in Eq. (47).

The convex combination is an interesting proposal to get the best feature of two or more adaptive algorithms. We were able to successfully apply it using the FXLMF and FXLMS algorithms with different step sizes. In the other scenario, we applied the combination over the FXLMS and FXLMF algorithms and we noticed that the convex-combined filter, at every iteration, followed the best algorithm.

A robustness test was done for all the scenarios used, to ensure that the proposed algorithms are able to adapt in case of a sudden change in the tap weights of the filter, either in the transient or steady state stage.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

The authors would like to thank the anonymous reviewers for their feedback that had improved the quality of the paper. The authors acknowledge the support provided by the Deanship of Scientific Research at KFUPM.

Received: 30 June 2015 Accepted: 16 March 2016
Published online: 01 April 2016

References

1. S Haykin, *Adaptive filter theory*, 4th edn. (Prentice-Hall, Englewood Cliffs, NJ, 2002)
2. AH Sayed, *Adaptive filters* (Wiley, NJ, USA, 2008)
3. E Hänsler, G Schmidt, *Acoustic echo and noise control: a practical approach* (Wiley & Sons, New Jersey, 2004)
4. SM Kuo, DR Morgan, *Active noise control systems, algorithms and DSP implementation functions* (Wiley, New York, 1996)
5. B Widrow, D Shur, S Shaffer, On adaptive inverse control, in *Proc. 15th Asilomar Conf*, 1981, pp. 185–189
6. B. Widrow, S.D. Stearns, *Adaptive signal processing*, (Prentice-Hall, Upper-Saddle River, NJ, 1985)
7. WA Gardner, Learning characteristics of stochastic-gradient-descent algorithms: a general study, analysis and critique. Signal Processing **6**, 113–133 (1984)
8. B Widrow, ME Hoff, Adaptive switching circuits, in *Proc. Of WESCON Conv. Rec., part 4*, 1960, pp. 96–140
9. P Dreiseitel, E Hänsler, H Puder, Acoustic echo and noise control—a long lasting challenge, in *Proc EUSIPCO*, 1998, pp. 945–952
10. E Hänsler, GU Schmidt, Hands-free telephones—joint control of echo cancellation and postfiltering. Signal Processing **80**, 2295–2305 (2000)
11. C Breining, P Dreiscitel, E Hänsler, A Mader, B Nitsch, H Puder, T Schertler, G Schmidt, J Tilp, Acoustic echo control. An application of very-high-order adaptive filters. IEEE Signal Proc. Mag. **16**(4), 42–69 (1999)
12. E. Bjarnason, Analysis of the Filtered-X LMS algorithm. IEEE Trans. Speech Audio Process. **3**, 504–514 (1995)
13. OJ Tobias, JCM Bermudez, NJ Bershad, R Seara, Mean weight behavior of the Filtered-X LMS algorithm, in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process*, 1998, pp. 3545–3548
14. OJ Tobias, Stochastic analysis of the Filtered-X LMS algorithm, in *Ph.D dissertation* (Federal Univ, Santa Catarina, Brazil, 1999)
15. OJ Tobias, JCM Bermudez, NJ Bershad, Mean weight behavior of the Filtered-X LMS algorithm. IEEE Trans. Signal Process. **48**(4), 1061–1075 (2000)
16. JE Mazo, On the independence theory of equalizer convergence. Bell Syst. Tech. J. **58**, 963–993 (1979)
17. GJ Rey, RR Bitmead, CR Johnson, The dynamics of bursting in simple adaptive feedback systems with leakage. IEEE Trans. Circuits Syst. **38**, 475–488 (1991)
18. L Vicente, E Masgrau, Novel FxLMS convergence condition with deterministic reference. IEEE Trans. Signal Process. **54**, 3768–3774 (2006)

19. K Mayyas, T Aboulnasr, Leaky LMS algorithm: MSE analysis for Gaussian data. IEEE Trans. Signal Process. **45**(4), 927–934 (1997)
20. E. Walach, B. Widrow, The Least Mean Fourth (LMF) adaptive algorithm and its family. IEEE Trans. Inf. Theory. **IT-30**(2), (1984), pp. 275–283
21. O.J. Tobias, R. Seara, Leaky-FXLMS algorithm: stochastic analysis for Gaussian data and secondary path modeling Error. IEEE T. Speech Audi. P. **13**(6), 1217–1230 (2005)
22. RD Gitlin, HC Meadors Jr, SB Weinstein, The tap-leakage algorithm: an algorithm for the stable operation of a digitally implemented fractionally spaced equalizer. Bell Syst. Tech. J. **61**(8), 1817–1839 (1982)
23. O Khattak, A Zerguine, Leaky Least Mean Fourth adaptive algorithm. IET Signal Process. **7**(2), 134–145 (2013)
24. LA Azpicueta-Ruiz, M Zeller, AR Figueiras-Vidal, J Arenas-García, Least squares adaptation of affine combinations of multiple adaptive filters, in *Proc. of IEEE Intl. Symp. on Circuits and Systems, Paris, France*, 2010, pp. 2976–2979
25. JC Burgess, Active adaptive sound control in a duct: a computer simulation. J. Acoust. Soc. Am. **70**, 715–726 (1981)
26. SS Kozat, AC Singer, Multi-stage adaptive signal processing algorithms, in *Proc. 2000 IEEE Sensor Array Multichannel SignalWork- shop, Cambridge, MA*, 2000, pp. 380–384
27. AC Singer, M Feder, Universal linear prediction by model order weighting. IEEE Trans. Signal Process. **47**, 2685–2700 (1999)
28. M Niedz`wiecki, Multiple-model approach to finite memory adaptive filtering. IEEE Trans. Signal Process **40**, 470–473 (1992)
29. J. Arenas-Garcia, L.A. Azpicuet-Ruiz, M.T.M. Silva, V.H. Nascimento, A.H. Sayed, Combinations of adaptive filters: performance and convergence properties. IEEE Signal Proc. Mag. **33**, 120-140 (2016)

Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com