A Computationally Efficient Frequency-Domain Filtered-X LMS Algorithm for Virtual Microphone

Debi Prasad Das[1], Danielle J. Moreau[2], Ben Cazzolato[3]

[1]Process Engineering and Instrumentation Cell, CSIR-Institute of Minerals and Materials Technology, Bhubaneswar, 751013, India

[1,2,3]School of Mechanical Engineering, The University of Adelaide SA 5005, AUSTRALIA

[1]debi_das_debi@yahoo.com, dpdas@immt.res.in

[2]danielle.moreau@adelaide.edu.au

[3]benjamin.cazzolato@adelaide.edu.au

ABSTRACT

The computational complexity of the virtual FXLMS algorithm is higher than that of the conventional FXLMS algorithm. The additional complexity comes from computation of three secondary path transfer functions (as opposed to one) and a transfer function between the physical and the virtual microphones. The order of these transfer functions may be very high in practical situations where the acoustic damping is low. The high computational complexity of the virtual FXLMS algorithm imposes issues like high power consumption, making it difficult to implement the algorithm in battery operated ANC devices such as active headsets. In addition, the operating sampling frequency of the algorithm is limited and this in turn restricts its operation to relatively low frequency applications. In this paper, a new virtual FXLMS algorithm is derived by implementing all of the secondary path transfer functions in the frequency domain. The algorithm is simulated using measured transfer functions in a duct with low acoustic damping. Implementation schemes are proposed for the new frequency-domain virtual FXLMS algorithm, citing its advantages for use as an efficient real-time active noise control algorithm.

Index Terms: **Active noise control, adaptive filter, frequency-domain adaptive filter, computational complexity, virtual ANC.**

I. INTRODUCTION:

Active noise control (ANC) has been an important topic of research due to its widespread application to control acoustic noise [1]. The objective of the research presented in this paper is to develop a computationally efficient ANC algorithm for practical use. The filtered-X LMS (FXLMS) algorithm is the simplest and most well known adaptive ANC algorithm, which is typically used to actively control the noise at the position of the error microphone [1]. However, there has been demand to control the noise at a remote location where it is not feasible to place a microphone, referred to as the virtual location. This has been achieved by modifying the FXLMS algorithm in a number of ways [2-8]. The most popular virtual ANC algorithms are the virtual microphone arrangement [3], the remote microphone technique [4], the forward-difference prediction technique [5], the adaptive LMS virtual microphone technique [6], the Kalman filtering virtual sensing technique [7] and the stochastically optimal tonal diffuse field (SOTDF) virtual sensing method [8]. The remote microphone technique [4], seen to be the most effective virtual sensing algorithm, uses the offline identification of two secondary paths: one between the control signal and the physical microphone position and the other between the control signal and the virtual microphone position. In addition, there is also the need to estimate the primary path between the physical and the virtual microphones. The conventional FXLMS algorithm incorporated with the remote microphone technique therefore requires the additional computation of these three transfer functions.

It is well known that most real-time ANC algorithms run on a sample-by-sample basis, where all the computations required to generate the control signal and tuning of the controller take place within a single sample period. The added

computational complexity of incorporating a virtual sensing algorithm means additional computational time is required by the processor and hence the maximum sampling frequency of operation is reduced. This in turn restricts the efficacy of ANC to frequencies less than half of its sampling frequency. A number of researchers have shown interest in developing computationally efficient algorithms, otherwise known as fast ANC algorithms [10-18]. A class of fast ANC algorithms known as the block algorithm, which exploits frequency-domain convolution and correlation operations, has been shown to achieve a significant computational advantage over its conventional time-domain counterpart [19-21]. One disadvantage of the block implementation is however, the inherent delay equal to the size of the block length. As a result, there are a number of works on delayless subband adaptive filtering algorithms applied to ANC [22, 23].

In this paper, all the estimated transfer functions in virtual ANC are implemented in the frequency-domain and the main control loop runs in the time-domain which makes the ANC delayless [22, 23]. In addition to this, the block algorithm proposed in this paper does not require that the length of the ANC filter and the length of the estimated acoustic paths to be equal, unlike the algorithms in [19-21]. Detailed computational complexity analysis is carried out to demonstrate the advantages of the proposed algorithm.

The organization of this paper is as follows. Section II briefly outlines the remote microphone based virtual ANC algorithm that uses the FXLMS algorithm. In Section III, the block frequency-domain virtual ANC algorithm is presented. The computational complexity of both algorithms is analysed in Section IV. A discussion on the implementation aspects of the proposed ANC structure is presented in Section

V. Results of simulation experiments in a lightly damped acoustic duct are shown in Section VI. The paper concludes in Section VII.

II. VIRTUAL FXLMS ALGORITHM

The conventional FXLMS algorithm may be modified by using the remote microphone technique (RMT) [5] to estimate the total error signal at the virtual location, $\hat{e}_v(n)$, using the error signal from a physical microphone, $e_p(n)$. The RMT requires a preliminary identification stage in which a second physical microphone is temporarily placed at the virtual location. The secondary transfer functions at the physical and virtual locations, $\hat{S}_p(z)$ and $\hat{S}_v(z)$ respectively, are measured during the preliminary identification stage along with the primary transfer function between the physical and virtual locations, $\hat{H}(z)$. In this paper, the ^ symbol indicates estimated quantities. Let us define $\hat{\mathbf{s}}_p$, $\hat{\mathbf{s}}_v$, $\hat{\mathbf{h}}$ as the impulse response vectors of $\hat{S}_p(z)$, $\hat{S}_v(z)$ and $\hat{H}(z)$, respectively.

A block diagram of the remote microphone technique is shown within the dotted box of Fig. 1. As shown in Fig. 1, an estimate of the primary disturbance, $\hat{d}_p(n)$, at the physical microphone is first calculated using

$$\hat{d}_p(n) = e_p(n) - \hat{y}_p(n) = e_p(n) - \hat{s}_P(n) * y(n), \qquad (1)$$

where $\hat{y}_p(n)$ is an estimate of the secondary disturbance at the physical microphone and $y(n)$ is the control signal. The '*' denotes the convolution operation. Next, an estimate of the primary disturbance, $\hat{d}_v(n)$, at the virtual location is estimated as

$$\hat{d}_v(n) = \hat{h}(n) * \hat{d}_p(n). \qquad (2)$$

Finally, an estimate, $\hat{e}_v(n)$, of the total virtual error signal from both sources is calculated using

$$\hat{e}_v(n) = \hat{d}_v(n) + \hat{y}_v(n) = \hat{d}_v(n) + \hat{s}_v(n) * y(n), \qquad (3)$$

where $\hat{y}_v(n)$ is an estimate of the secondary disturbance at the virtual microphone location. Thus, an estimate of the virtual error signal has been calculated from the physical error signal.

The control signal, $y(n)$, is computed by convolving the reference microphone signal $x(n)$ with the ANC filter coefficient as follows

$$y(n) = x(n) * \hat{w}(n). \qquad (4)$$

This virtual error signal is used to update the ANC weights vector $\hat{\mathbf{w}}(n)$ according to

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) - \mu \hat{e}_v(n)\mathbf{x}'(n), \qquad (5)$$

where $\mathbf{x}'(n) = [x'(n) \quad x'(n-1) \quad \ldots \quad x'(n-N+1)]$ is the filtered reference signal, that is formed by filtering the reference signal $x(n)$ with the secondary path estimate to the virtual microphone location and is expressed as

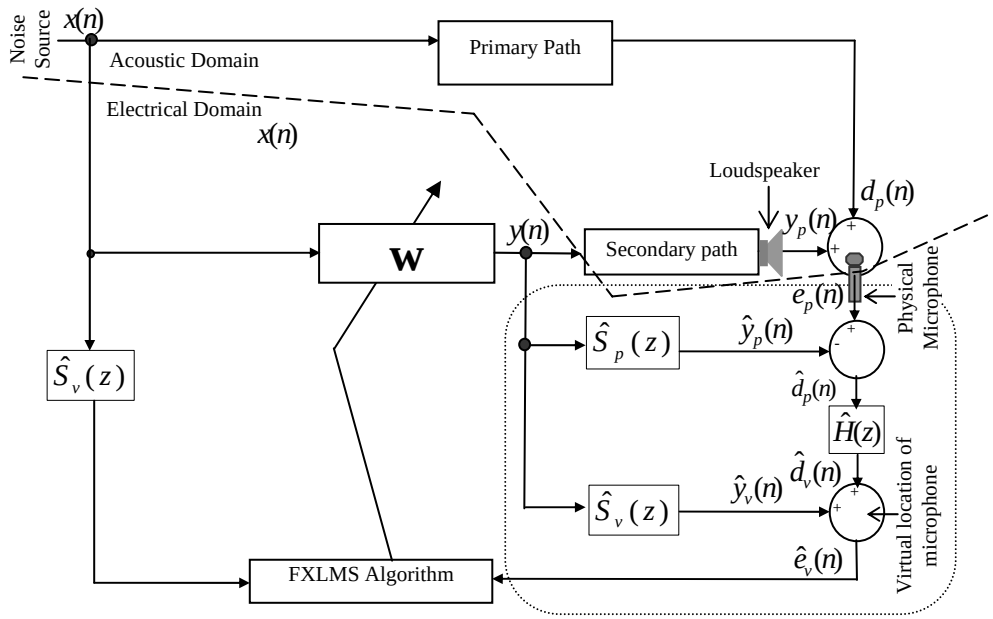


Fig. 1 Block diagram of conventional virtual FXLMS algorithm. Dashed line divides the acoustic and electrical domain. The dotted line contains the elements of the remote microphone technique.

$$x'(n) = x(n) * \hat{s}_v(n). \qquad (6)$$

In (1) to (6), five convolution operations and one cross-correlation operation are required. The convolution operations are shown in (1)-(4) and (6) and the cross-correlation operation is shown in (5). It should be noted that the convolution operation used in (4) is responsible for producing the control signal to generate the secondary noise to control the primary noise, where as the other convolutions and correlations are responsible for updating (tuning) the ANC weights, $w(n)$. In the time-domain, the control signal is generated and the ANC weights are updated on sample-by-sample (real-time) basis.

III. BLOCK FREQUENCY-DOMAIN VIRTUAL FXLMS ALGORITHM

In this section, we implement four of the five convolution operations and the correlation operation in the frequency-domain using the overlap-save method [22]. The convolution operation in (4) is computed in the time-domain to make the algorithm delayless [22, 23]. In this implementation scheme, the ANC filter is of length N and all the estimated acoustic path transfer functions are of length L where L and N are a power of 2 and $N = 2^m L$, where m is a positive definite integer. This assumption is made so that both the lengths L and N are suitable for computing the Fourier transform using the fast Fourier transform (FFT). Let us define the following three signal vectors of length L.

$\mathbf{x}(n) = [x(n) \quad x(n-1) \quad \ldots \quad x(n-L+1)]$ is the reference signal vector,

$\mathbf{y}(n) = [y(n) \quad y(n-1) \quad \ldots \quad y(n-L+1)]$ is the control signal vector and

$\mathbf{e}_p(n) = [e_p(n) \quad e_p(n-1) \quad \ldots \quad e_p(n-L+1)]$ is the physical error signal vector.

The estimated impulse response vectors of transfer functions $\hat{\mathbf{s}}_p$, $\hat{\mathbf{s}}_v$ and $\hat{\mathbf{h}}$ are converted to the frequency-domain offline by appending a zero vector of length L and taking their Fourier transforms, which are represented as $\hat{S}_p$, $\hat{S}_v$ and $\hat{H}$ respectively as follows

$$\hat{S}_p = F[\hat{\mathbf{s}}_p \quad 0_L], \tag{7}$$

$$\hat{S}_v = F[\hat{\mathbf{s}}_v \quad 0_L], \tag{8}$$

$$\hat{H} = F[\hat{\mathbf{h}} \quad 0_L], \tag{9}$$

where 0_L is a $1\times L$ vector of zeros and $F[.]$ is the Fourier transform operation. Since the above three complex vectors are not time-varying, no index is used to represent them.

To compute the convolution operation $\hat{s}_p(n) * y(n)$ shown in (1), two blocks (past and present) of the $\mathbf{y}(n)$ vector of length L need to be concatenated and then the Fourier transform is computed as follows

$$Y(k_1) = F[\mathbf{y}(nk_1 - L) \quad \mathbf{y}(nk_1)], \tag{10}$$

where $\mathbf{y}(nk_1 - L)$ and $\mathbf{y}(nk_1)$ are the past and present control signal vectors of size $1\times L$ each of the k_1th block. The k_1th block frequency-domain control signal vector of length $2L$ is represented by $Y(k_1)$.

The L- point filtered output of the control signal $y(n)$ through filter $\hat{s}_p(n)$ is computed for the k_1th block as follows

$$\hat{\mathbf{y}}_p(k_1) = \text{ last } L \text{ points}\{F^{-1}[Y(k_1) \otimes \hat{S}_p]\}, \tag{11}$$

where $\otimes$ and $F^{-1}[.]$ represent point-by-point multiplication and the inverse Fourier transform operation, respectively.

Similarly, the L- point filtered output of the control signal $y(n)$ through filter $\hat{s}_v(n)$ is computed for the k_1th block as follows

$$\hat{\mathbf{y}}_v(k_1) = \text{ last } L \text{ points}\{F^{-1}[Y(k_1) \otimes \hat{S}_v]\}. \tag{12}$$

The $L-$point primary disturbance estimate, $\hat{d}_p(n)$, at the physical microphone location is computed for k_1th block as

$$\hat{\mathbf{d}}_p(k_1) = \hat{\mathbf{e}}_p(k_1) - \hat{\mathbf{y}}_p(k_1). \tag{13}$$

Concatenating two blocks of this primary disturbance estimate, $\hat{\mathbf{d}}_p(nk_1 - L)$ and $\hat{\mathbf{d}}_p(nk_1)$, and taking the Fourier transform produces the $2L-$point complex vector given by

$$\hat{D}_p(k_1) = F[\hat{\mathbf{d}}_p(nk_1 - L) \quad \hat{\mathbf{d}}_p(nk_1)]. \tag{14}$$

The $L-$point estimate of the primary disturbance, $\hat{d}_v(n)$, at the virtual location is computed for the k_1th block as

$$\hat{\mathbf{d}}_v(k_1) = \text{ last } L \text{ points}\{F^{-1}[\hat{D}_p(k_1) \otimes \hat{H}]\}. \tag{15}$$

Finally, the $L-$point error signal at virtual location is estimated as

$$\hat{\mathbf{e}}_v(k_1) = \hat{\mathbf{d}}_v(k_1) + \hat{\mathbf{y}}_v(k_1). \tag{16}$$

Before doing the correlation operation, computation of the filtered reference signal is required. The reference signal $x(n)$ is to be filtered through the secondary path estimate to the virtual location, $\hat{s}_v(n)$, as shown in (6). To do this filtering in the frequency-domain, two blocks of the reference signal need to be concatenated and then the Fourier transform is calculated as follows

$$X(k_1) = F[\mathbf{x}(nk_1 - L) \quad \mathbf{x}(nk_1)]. \tag{17}$$

The $L-$point filtered reference signal, $x'(n)$, is computed as

$$\mathbf{x}'(k_1) = \text{ last } L \text{ points}\{F^{-1}[X(k_1) \otimes \hat{S}_v]\}. \tag{18}$$

Since the ANC filter is of length N, m blocks of the filtered reference signal $\mathbf{x}'(k_1)_{1\times L}$ and the estimated virtual error signal $\hat{\mathbf{e}}_v(k_1)_{1\times L}$ have to be used to create $X'(k)$ and $\hat{E}_v(k)$, respectively. It is proposed that a buffer of size N may be used for each of $\mathbf{x}'(k_1)$ and $\hat{\mathbf{e}}_v(k_1)$. Now

$$X'(k) = F[\mathbf{x}'(nk - N) \quad \mathbf{x}'(nk)]_{1\times 2N}, \tag{19}$$

and

$$\hat{E}_v(k) = F[\hat{\mathbf{e}}_v(nk - N) \quad \hat{\mathbf{e}}_v(nk)]_{1\times 2N}, \tag{20}$$

where

$$\mathbf{x}'(nk) = [\mathbf{x}'(nk_1), \mathbf{x}'(2nk_1), ..., \mathbf{x}'(mnk_1)]_{1\times N}, \tag{21}$$

and

$$\hat{\mathbf{e}}_\mathbf{v}(nk) = [\hat{\mathbf{e}}_\mathbf{v}(nk_1), \hat{\mathbf{e}}_\mathbf{v}(2nk_1), ..., \hat{\mathbf{e}}_\mathbf{v}(mnk_1)]_{1\times N}. \tag{22}$$

The change in the time-domain weights can be computed in the frequency-domain as

$$\Delta\hat{\mathbf{w}}(k) = \text{first } N - \text{points } \{F^{-1}[\hat{E}_v(k) \otimes X'(k)^H]\}, \tag{23}$$

where $X'(k)^H$ = conjugate of $X'(k)$.

The change in weight $\Delta\hat{\mathbf{w}}(k)$ is evaluated at every block time (in other words every N samples) and added to the ANC filter coefficients $\hat{\mathbf{w}}(n)$. However, the

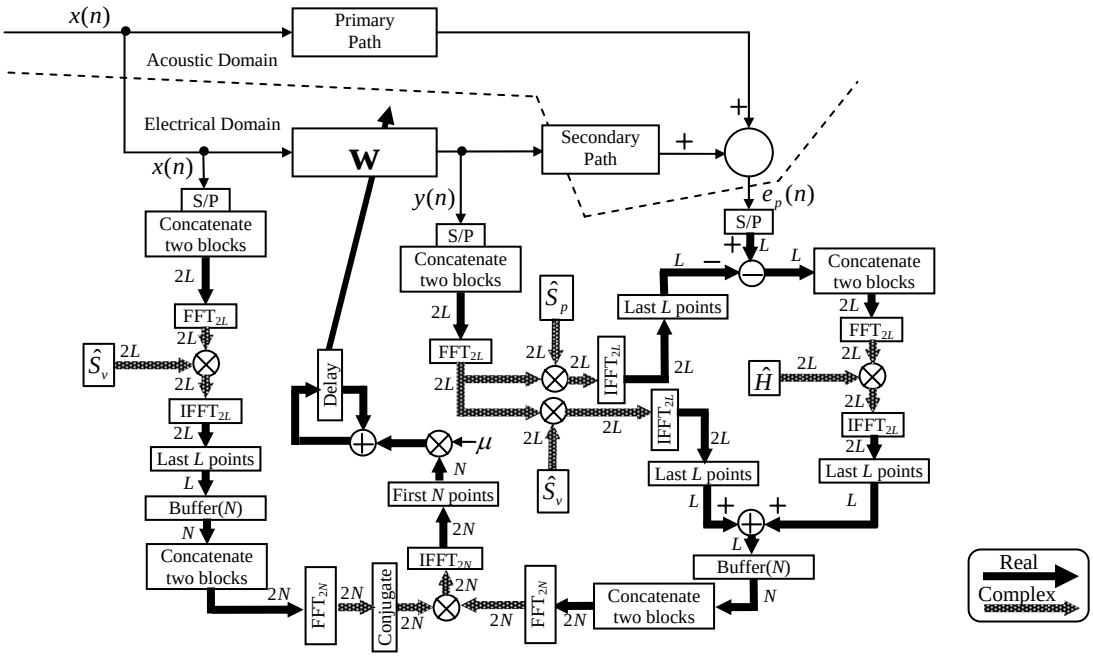


Fig. 2 Block diagram of the exact frequency-domain virtual ANC algorithm. The thin solid line represents real sample-by-sample data, the thick solid lines are buffered real samples and the thick shaded lines are complex frequency-domain data.

control signal output of the ANC system is still computed every sample as per (4). The filter coefficients, $\mathbf{w}(n)$, remain constant for the block period (for N consecutive samples) and change after the kN th sample period. The block diagram of this algorithm is presented in Fig. 2. The Fourier transforms are implemented using fast Fourier transforms (FFT) and the inverse Fourier transform is implemented by the inverse FFT (IFFT). In this figure, the thin solid line represents real sample-by-sample data, the thick solid lines are buffered real samples and the thick shaded lines are complex frequency-domain data. The buffer size represented by each line is also given in the figure.

IV. COMPUTATIONAL COMPLEXITY ANALYSIS

The main contribution of this paper is to achieve computational advantage over the time-domain virtual FXLMS algorithm without any performance degradation. This section is devoted to computational complexity analysis of the frequency-domain virtual ANC algorithm compared to its time-domain counterpart. For comparison purposes, it is assumed that one radix-2 FFT of length N uses $\frac{N}{2}\log_2 N$ complex multiplications and $N\log_2 N$ complex additions [24]. In other words, the same FFT uses $2N\log_2(N)$ real multiplications and $2N\log_2(N)$ real additions. Each complex multiplication is 4 real multiplications and 3 real additions.

A. Block frequency-domain virtual ANC

During frequency-domain ANC implementation, it is assumed that all estimated acoustic paths $\hat{S}_p(z)$, $\hat{S}_v(z)$ and $\hat{H}(z)$ are fixed FIR filters of length L and the ANC

is an adaptive FIR filter of length N. The proposed block frequency-domain algorithm uses N multiplications and $N-1$ additions at every sample. At every Lth sample, the algorithm uses 7 sets of $2L-$point FFTs, 4 sets of $2L-$point complex multiplications and 2 sets of $L-$point real additions. However, after the end of every Nth sample it uses 3 sets of $2N-$point FFTs, 1 set of $2N-$point complex multiplications and one set of $N-$point real additions. The multiplication of μ is neglected as it can be computed by a shift operation by choosing a value equal to the negative power of two. Accordingly, the total number of real multiplications over a period of N samples (where $N=2^m L$) will be

$$M(L,N)=N^2+2^m[28L\log_2(2L)+32L]+12N\log_2(2N)+8N=$$

$$N[28\log_2(L)+12\log_2(N)+N+80],$$

and the total number of real additions will be

$$A(L,N)=N(N-1)+2^m[28L\log_2(2L)+24L+2L]+12N\log_2(2N)+6N+N=$$

$$N[28\log_2(L)+12\log_2(N)+N+72].$$

There is the option to convert the estimated acoustic paths of length L to a vector of length N by appending it with zeros. If this is done, then L will be equal to N. For the special case of $L=N$, the multiplication count will be

$$M(N)=N[40\log_2(N)+82+N],$$

and the addition count is

$$A(N)=N[40\log_2(N)+N+72].$$

B. Time-domain virtual ANC algorithm

The time-domain virtual ANC algorithm does not require equal filter orders for all of the secondary paths and the ANC filter. Therefore, for comparison purposes, it is assumed that the length of each estimated acoustic path (secondary and the primary)

is L and the ANC adaptive filter is of length N. There are a total of 5 sets of convolution and one correlation operation involved in the time-domain virtual ANC algorithm. The forward path ANC filter and the cross-correlation operation used for the weight update are of length N and the remaining four convolution operations are of length L. Each convolution of length N uses N real multiplications and $N-1$ real additions. The cross-correlation operation used to update the weights of the ANC filter uses N real multiplications and N real additions. Hence the total number of real multiplications required every sample is $4L+2N$ and the total number of additions are $4(L-1)+(2N-1)$. The total real multiplications and additions required during N samples is $2N(2L+N)$ and $N(4L+2N-5)$, respectively.

C. Computational complexity comparison

Figure 3 shows graphically the computational complexity of both the time-domain and the block frequency-domain virtual FXLMS algorithm as a function of N for different $L \le N$. It is clearly shown that the complexity of the time-domain algorithm is much higher than that of the frequency-domain algorithm.

To see the computational complexity quantitatively, the multiplication counts are tabulated in Table I for various values of N and L. Also tabulated is the computational complexity required in both algorithms at every sample, L th sample and N th sample and the total computation over N samples. The percentage saving of the total computation over N samples in both algorithms is tabulated and shows that the proposed frequency-domain algorithm is more energy efficient which is critical for embedded applications. The energy saving would be approximately proportional to the multiplication count of the algorithm. Table I clearly shows that the computational saving is greater for higher values of N and L. The saving is

maximum when N=L. When L is small, the saving is poor which is due to the fact that the frequency-domain convolution is not efficient for small sized vectors.

D. Delayed block frequency-domain virtual FXLMS algorithm

From the above computational complexity analysis it may be noted that the large computational load at the end of every L th and N th sample period can be shifted to be computed in the next block period. This approach is termed the delayed block frequency-domain virtual FXLMS algorithm. The computational saving of the proposed delayed block frequency-domain FXLMS algorithm (detailed in Section V(C)) with respect to the time-domain FXLMS algorithm is analysed here. In the delayed block frequency-domain FXLMS algorithm, the large computation at the end of the N th sample point is distributed over the next block period. Hence, in every block period, at each sample, the ANC processor has to compute the forward path filter (N multiplications and $N-1$ additions) and a part of the extra computation that was supposed to be computed at the N th sample of the previous block. Therefore, the computational load per sample in the delayed block FXLMS algorithm is 1/Nth of the total computational requirement over a period of N samples. To emphasize the saving at each sample point, the multiplication count for various N and L values are tabulated in Table II. This table shows the computational saving of the proposed algorithm over the time-domain algorithm at every sample.

It can be seen from both Table I and II that the frequency-domain ANC filter is computationally most efficient when $N = L$. In an ANC system, the ANC adaptive filter is effectively the convolution of the primary path and the inverse of the virtual secondary path. Hence the length of the ANC filter should be always greater than the length of the acoustic path filters. The proposed algorithm is a generalized implementation of the frequency-domain ANC algorithm for $L \le N$. However, the

length of the acoustic path can be made equal to the length of the ANC filter N by appending it with zeros. The computational effectiveness of the algorithm for the case $N = L + M$, where M is the number of zeros appended, is now examined. Figure 4 shows the computational complexity of the time-domain FXLMS algorithm compared to the block frequency-domain FXLMS algorithm for $L < N$, where the multiplication and addition counts are plotted for all values of L with $N = 1024$. This figure shows that in the block algorithm, for $L = N$, the computational counts remain constant for all values of L. The computational count of the time-domain FXLMS algorithm, however, shows a rapid increase as $L \rightarrow N$ and is significantly higher compared to the frequency-domain FXLMS algorithm for all values of L. Figure 4 shows that the computational effort of the block FXLMS algorithm is least when $L << N$.

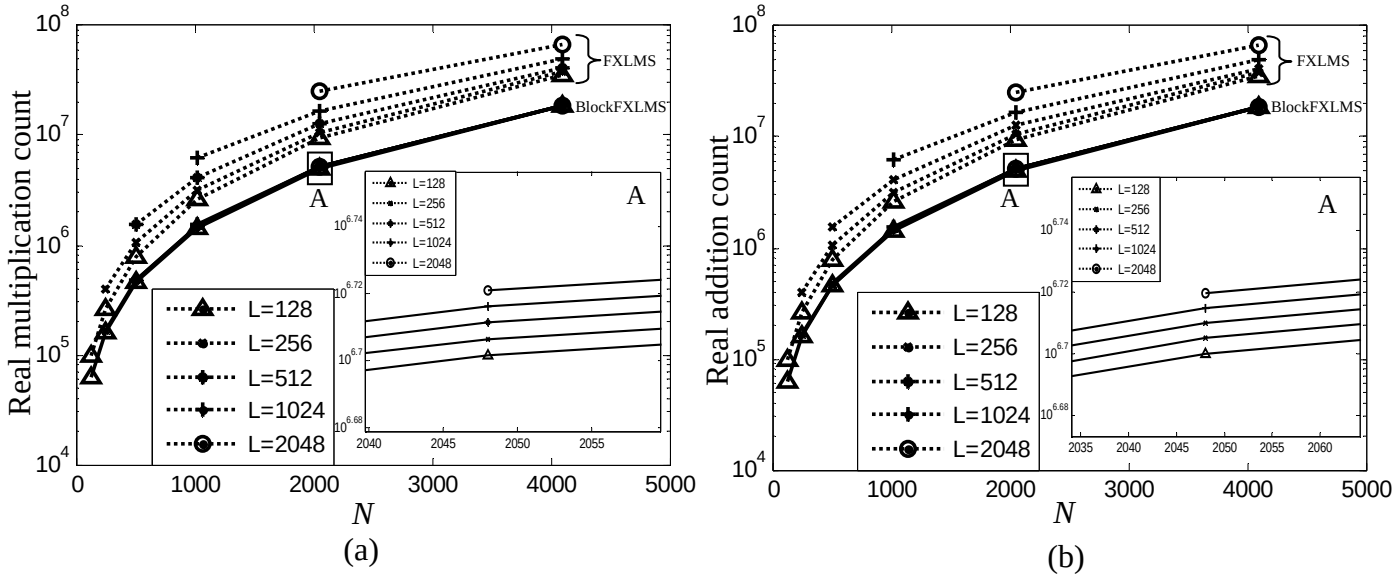


Fig. 3. Comparison of the computational complexity of the time-domain virtual FXLMS and the block frequency-domain virtual FXLMS; (a) multiplication count, (b) addition count. The inset box shows the difference in the block algorithm complexity.

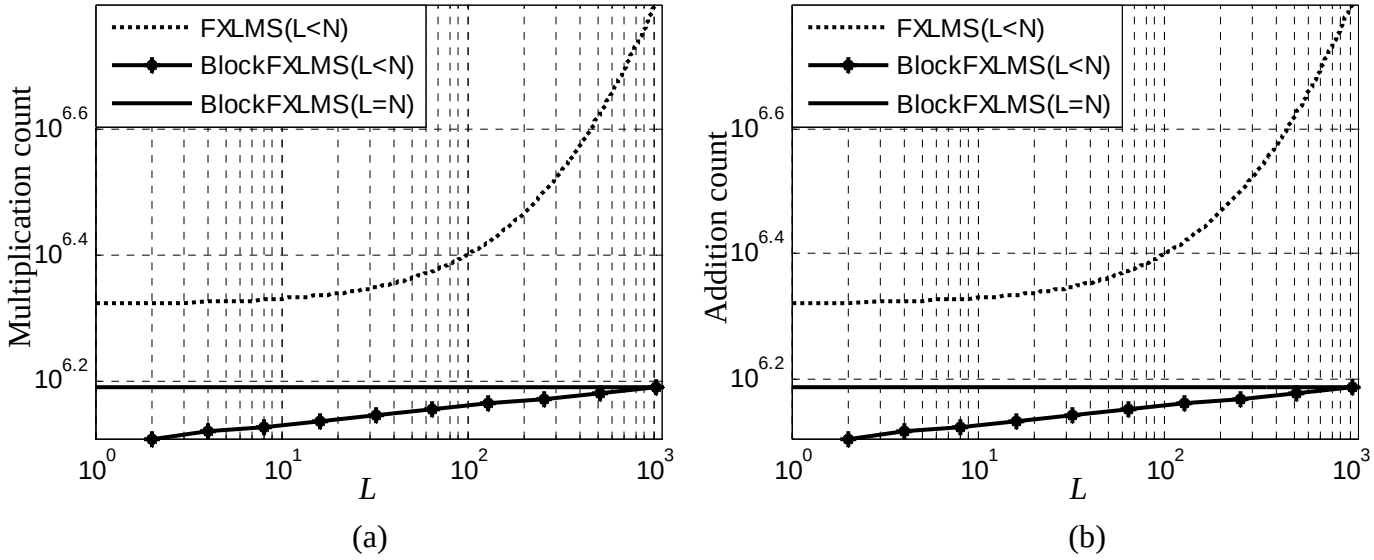


Fig. 4. Comparison of the computational complexity of the time-domain and the block frequency-domain virtual ANC algorithm for different orders of secondary paths ($N = 1024$); (a) multiplication count, (b) addition count.

TABLE-I

COMPARISON OF MULTIPLICATION COUNT BETWEEN BLOCK FREQUENCY-DOMAIN AND TIME-DOMAIN VIRTUAL ANC ALGORITHMS.

Order	Time-Domain (*TD*)		Block Frequency-Domain (*FD*)				Percentage saving over *N* samples
	Every sample	Total computation over *N* Samples	Every sample except *L*th and *N*th sample	Every *L*th Sample ($L < N$)	Every *N*th sample	Total computation over *N* Samples	
N	$2(2L+N)$	$2N(2L+N)$	*N*	$28L\log_2(2L)+32L+N$	$28L\log_2(2L)+32L+12N\log_2(2N)+9N$	$N[28\log_2(L)+12\log_2(N)+N+80]$	$\left(\frac{TD-FD}{TD}\right)100$
$L = N$							
64	384	24576	64	-	20608	24064	02.08
128	768	98304	128	-	46336	61440	37.50
256	1536	393216	256	-	102912	165888	57.81
512	3072	1572864	512	-	226304	483328	69.27
1024	6144	6291456	1024	-	493568	1531904	75.65
2048	12288	25165824	2048	-	1069056	5242880	79.16
4096	24576	100663296	4096	-	2301952	19038208	81.08
$L = N/2$							
64	256	16384	64	6464	9728	22272	-35.93
128	512	65536	128	14720	22016	57856	11.71
256	1024	262144	256	33024	49152	158720	39.45
512	2048	1048576	512	73216	108544	468992	55.27
1024	4096	4194304	1024	160768	237568	1503232	64.16
2048	8192	16777216	2048	350208	516096	5185536	69.09
4096	16384	67108864	4096	757760	1114112	18923520	71.80
$L = N/4$							
64	192	12288	64	2816	4736	20480	-66.66
128	384	49152	128	6528	10752	54272	-10.41
256	768	196608	256	14848	24064	151552	22.91
512	1536	786432	512	33280	53248	454656	42.18
1024	3072	3145728	1024	73728	116736	1474560	53.12
2048	6144	12582912	2048	161792	253952	5128192	59.24
4096	12288	50331648	4096	352256	548864	18808832	62.63

TABLE-II

PER-SAMPLE MULTIPLICATION REQUIREMENT IN DELAYED BLOCK FXLMS ALGORITHM COMPARED TO THE TIME-DOMAIN FXLMS ALGORITHM.

	Time-Domain FXLMS			Frequency-domain Block FXLMS with Delayed adaption (Total computation/*N*)			Percentage saving per sample		
N	*L*=*N*	*L*=*N*/2	*L*=*N*/4	*L*=*N*	*L*=*N*/2	*L*=*N*/4	*L*=*N*	*L*=*N*/2	*L*=*N*/4
64	384	256	192	376	348	320	2.08	-35.93	-66.66
128	768	512	384	480	452	424	37.50	11.71	-10.41
256	1536	1024	768	648	620	592	57.81	39.45	22.91
512	3072	2048	1536	944	916	888	69.27	55.27	42.18
1024	6144	4096	3072	1496	1468	1440	75.65	64.16	53.12
2048	12288	8192	6144	2560	2532	2504	79.16	69.09	59.24
4096	24576	16384	12288	4648	4620	4592	81.08	71.80	62.63

V. DISCUSSION AND IMPLEMENTATION SCHEME

The proposed virtual ANC algorithm computes only one forward filter every sample, three estimated acoustic path filters at every L th sample period and the weight vector is updated at the end of the sample-number equal to the length of this forward filter, N . If the computational complexity over the duration of ANC filter-length times the sample period is examined, it can be seen that the proposed method uses much less computation than its conventional time-domain counter part. The computation required by the proposed frequency-domain algorithm at the L th and N th sample period is, however, much higher than that of the conventional FXLMS based virtual sensing algorithm. This burden may be too great for some processors to handle. To overcome this issue, three implementation schemes for the block frequency-domain virtual FXLMS algorithm are proposed as follows.

A. Using dual-processor or processor-FPGA architecture:

Nowadays, a number of dual processor architectures and processors with field programmable gate array (FPGA) architecture are available [25]. The computation of the ANC filter should be made only by the master processor at every sample. The acoustic path transfer function filters and the weight update algorithm may be computed at the end of the Lth and N th sample period by the slave processor or the FPGA. The slave processor will be in sleep mode for all sample periods except during the Lth and N th sample period of the block. The slave processor would store all partial results computed every L th sample period and return the change in weight vector at the N th sample period of the block. The change in weight vector (see 23)

can be suitably transferred to the main processor during that sample period to replace the old weight vector.

B. Using graphics processing units (*GPU*)*:*

Graphics processing units (GPU) are very popular nowadays in a number of image processing applications where the Fast Fourier Transform (FFT) is used for the reconstruction of images from raw data. The current generation of graphics cards have the power, programmability and floating point precision required to perform the FFT efficiently. Since the proposed algorithm is FFT intensive, it is feasible that the GPU is used as the processor. The authors in [26] show the implementation of an FFT on NVIDIA Quadro NV4x family GPUs. They also compare the FFT run on a GPU with an FFT run on a traditional CPU. In [27], a new FFT implementation scheme is proposed exploiting the vectorized multiply-and-add operations of the GPU. The authors strongly believe that as computational complexity is one of the bottlenecks of multichannel ANC algorithms, the proposed FFT based method being a computationally efficient algorithm, would be extremely attractive when designing application specific ICs (ASICs) like GPUs for ANC applications. This is because each FFT can be computed with equal speed as a multiplication operation. The ASICs would have FFT cores and a single instruction for the FFT operation of certain length.

C. Delayed adaptation:

Another way of implementing this energy efficient frequency-domain algorithm using the general purpose CPUs is by delayed adaptation of the ANC filter. In this scheme, the ANC filter is computed at every sample and the weight update algorithm that is supposed to be computed at the last sample period of the block will be calculated

during the next block period. In this scheme, weight adaptation will be computed after a two block delay instead of a one block delay as done in the proposed frequency-domain virtual ANC algorithm.

VI. COMPUTER SIMULATIONS

To investigate the performance of the proposed block frequency-domain virtual ANC algorithm, simulation experiments were carried out using experimentally measured acoustical path transfer functions measured in a wooden rectangular duct of size 240 cm × 20.5 cm × 20.5 cm. One end of the duct was open and the other end was rigidly terminated. The experimental setup is shown in Fig. 5 where a dSpace 1104 ACE kit was used to generate the loudspeaker control signals via a KROHN HITE lowpass filter and a PLAYMASTER speaker amplifier. The system also measured the signals of two microphones which were separated by approximately 5 cm, through microphone amplifiers and a KROHN HITE anti-aliasing lowpass filter. The sampling rate for the entire data acquisition process was set at 1 kHz and hence the lowpass filter cut-off frequency was set at 0.5 kHz. Signal acquisition was done in two phases. First the secondary loudspeaker was actuated with a white signal with the primary loudspeaker off, and then the primary loudspeaker was actuated with a similar signal with the secondary loudspeaker off.

Offline models of the transfer functions were calculated in Matlab using Welch's averaged periodogram method where a 1024 point FFT was used with a 75% overlapped Hanning window. In addition to this, the coherence and the impulse response functions were also evaluated. The conditions under which the transfer functions were estimated are given in Table III. The frequency and phase response, coherence function and impulse response function of the estimated FIR filters are shown in Fig. 6 for the different acoustic paths. In all simulations, the actual primary

and secondary paths were implemented as FIR filters of 256 taps and the estimated secondary paths and the primary path between physical and virtual microphones were truncated to be FIR filters of 128 taps. The first 128 taps shaded in Fig. 6 were used as the estimated acoustic paths to make the simulation more realistic.

The simulations were done using experimentally measured transfer functions. However, real-time results are not shown as the hardware necessary for real-time implementation is no longer accessible to the authors. However, the paper does include four simulation experiments (listed below), to verify the efficacy of the new algorithm. The authors believe that although not completely exhaustive, the experiments conducted provide a broad cross section of the types of signals and algorithm arrangements that would find application in practice.

A. Experiment 1: Tonal noise
B. Experiment-2: Recorded noise of an air conditioner
C. Experiment-3: Delayed adaptation both for tonal and recorded noise
D. Experiment-4: Estimated Secondary path and the ANC Filter are of different length $(L \leq N)$

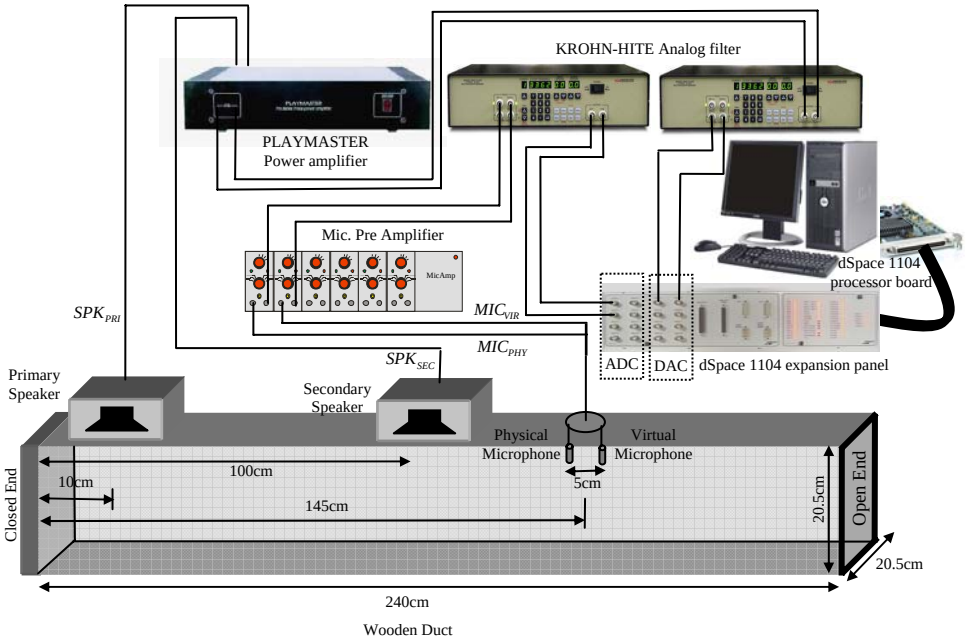


Fig. 5. Experimental setup for data acquisition for modelling the acoustic path used in the simulation.

TABLE-III
CONDITIONS WHEN MODELLING THE ACOUSTIC PATH TRANSFER FUNCTIONS FOR SIMULATION.

Acoustic path	Input (x)	Output (y)	Secondary loudspeaker	Primary loudspeaker
Secondary path to the physical microphone, S_p	Secondary loudspeaker control signal, SPK_{SEC}	Physical microphone signal, MIC_{PHY}	ON	OFF
Secondary path to the virtual microphone position, S_v	Secondary loudspeaker control signal, SPK_{SEC}	Virtual microphone signal, MIC_{VIR}	ON	OFF
Primary path to the physical microphone position, P_p	Primary loudspeaker control signal, SPK_{PRI}	Physical microphone signal, MIC_{PHY}	OFF	ON
Primary path to the virtual microphone position, P_v	Primary loudspeaker control signal, SPK_{PRI}	Virtual microphone signal, MIC_{VIR}	OFF	ON
Primary path between physical and virtual microphone, H	Physical microphone signal, MIC_{PHY}	Virtual microphone signal, MIC_{VIR}	OFF	ON

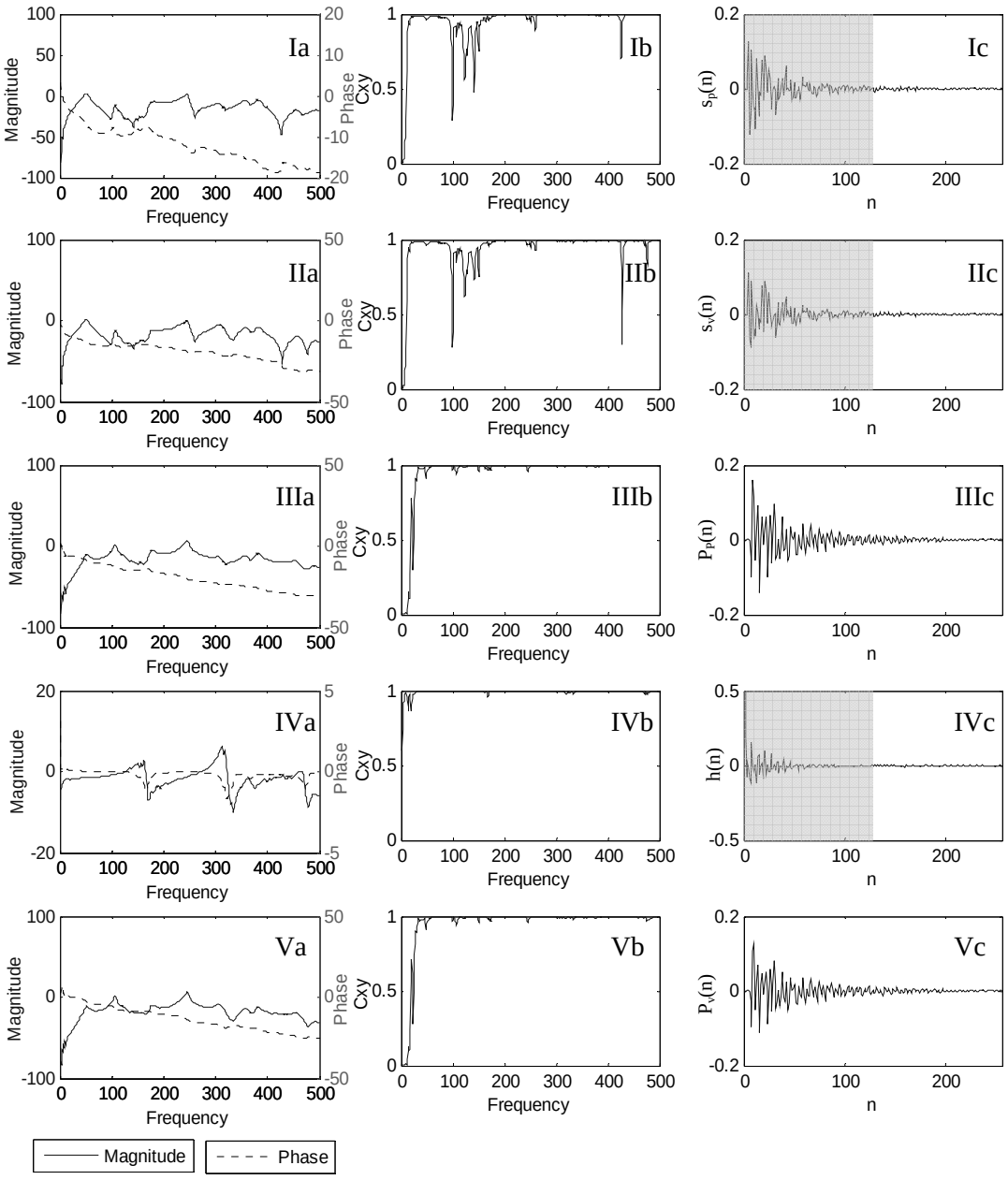


Fig. 6. Estimation of acoustic paths and their responses; a: frequency and phase response, b: coherence function, c: impulse response; I: secondary path to the physical microphone, II: secondary path to the virtual microphone position, III: primary path to the physical microphone position, IV: primary path between physical and virtual microphone, V: primary path to the virtual microphone position; the shaded portion of the impulse responses are used as estimated acoustic paths.

A. Experiment 1: Tonal noise

In this experiment, tonal noise was used as the primary noise. The noise was generated with the following equation

$$x(n) = 0.5\sin\left(\frac{2\pi nf}{f_s} + \varphi\right), \tag{24}$$

where the source frequency was $f = 200$ Hz, the sample rate was $f_s = 1$ kHz and φ is the phase. This experiment was conducted 20 times with the phase being randomly selected each time to ensure independence. The phase, φ, was set to be $2\pi(rand)$, where *rand* is a random number between 0 and 1. The step size for the time-domain and the block frequency-domain virtual FXLMS algorithms was selected to be 0.002. The actual primary and the secondary paths were of length 256. All estimated acoustic paths were a truncated version of the original acoustic path filter and were the 128 initial coefficients, as shown in Fig. 6. The length of the ANC filter is $N = 128$. Accordingly, in this simulation, case of $L = N$ is considered. The mean square error (MSE) plotted with log scale for both FXLMS algorithms is shown in Fig. 7. This figure shows that both the time-domain and frequency-domain algorithms converge to the same noise floor and hence their performance is identical. This experiment also confirms their equivalence performance under the tonal noise case. In Fig. 7, e_p and e_v are the physical and virtual error signals respectively. By zooming in on the transient portion of the MSE convergence (Fig. 7 (b)), it is shown that the block algorithm steps its convergence performance at every block length (here $N = 128$), and it tracks closely to the time-domain FXLMS algorithm which is updated at every sample. The zoomed plot at steady-state (Fig. 7 (c)) shows the equivalent performance of both algorithms when the filters are fully adapted to the acoustic environment and the noise source.

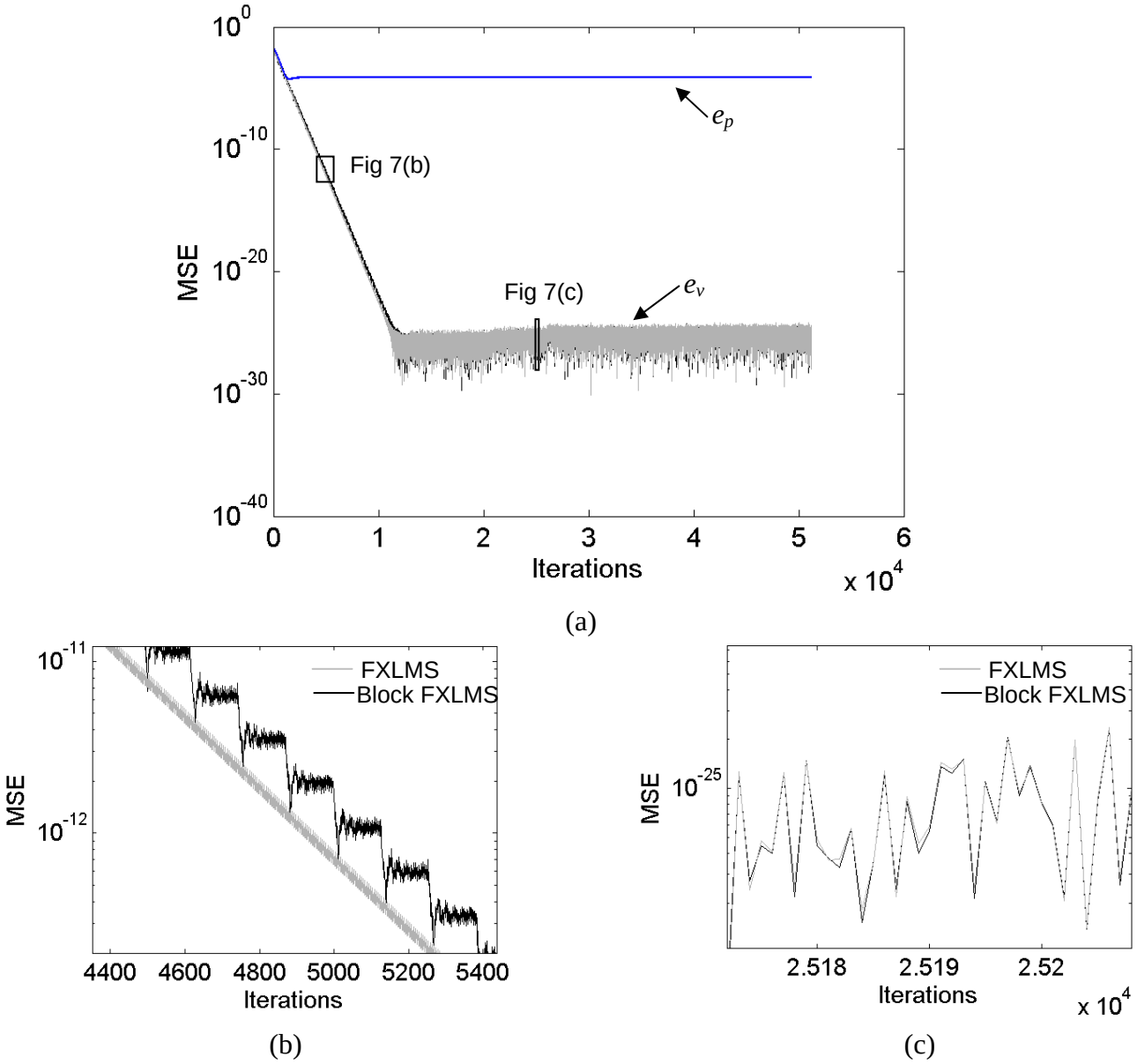


Fig. 7. MSE plots using the tonal noise for both FXLMS and the Block FXLMS algorithms (a) complete (b) during convergence and (c) at steady state.

B. Experiment-2: Recorded noise of an air conditioner

To make the simulation more realistic, noise was recorded from an air-conditioner near the cool air vent using the dSpace system. The sampling frequency during recording was 1 kHz. Figure 8 shows the performance of both algorithms in controlling the air-conditioner noise. As for Experiment-1, $L = N = 128$ is also used in this case. The step-size for both algorithms was fixed at 0.01. The MSE plot obtained for both algorithms is shown in Fig. 8. In this figure, the ensemble average of 20 independent experiments is smoothed by a moving-average method using a window of 1000 samples. The proposed block algorithm shows slight performance improvement over the conventional sample based FXLMS algorithm. This may be because the impulsive nature of the recorded input noise restricts the sample based algorithm from converging to an optimal solution. In addition to the MSE plot, the power spectra of the various noise signals after convergence are plotted in Fig. 9, which allows comparison of the frequency specific noise reduction performance of both algorithms. It can be seen that the signal at the noise source is high and is reduced at the virtual location where noise cancellation is desired. This is due to the noise attenuating property of the acoustical duct over the distance of propagation from the noise source to the virtual location. Figure 9 clearly shows that both the time-domain FXLMS and the proposed block frequency-domain FXLMS algorithm display noise attenuating properties at all major peaks except in the frequency range marked by the shaded area. This frequency range corresponds to that where the secondary paths display low amplitudes as shown in Fig. 6. It can be seen in Fig. 6 that the coherence functions of the two secondary paths deviate from unity in the same frequency range indicating that it is not possible to achieve significant noise control in

this frequency band using this ANC system architecture (that includes the type of secondary source and its placement).

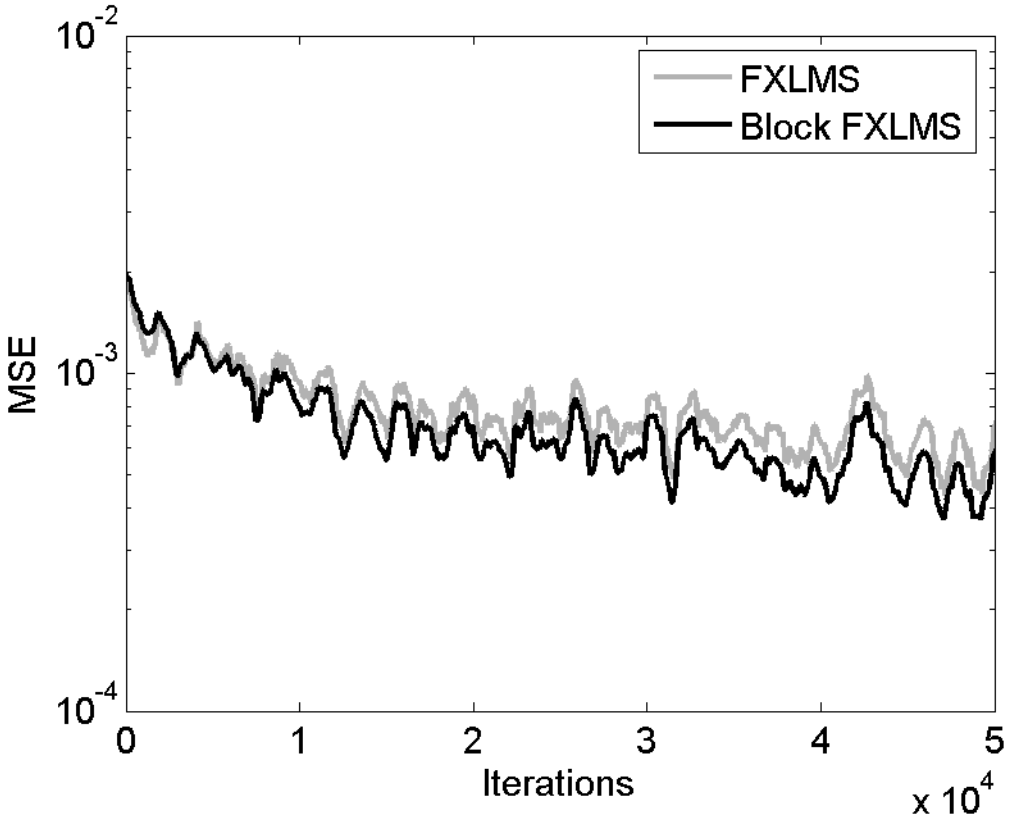


Fig. 8. The smoothed MSE plot of both FXLMS and block frequency-domain FXLMS algorithm for recorded air-conditioner noise.

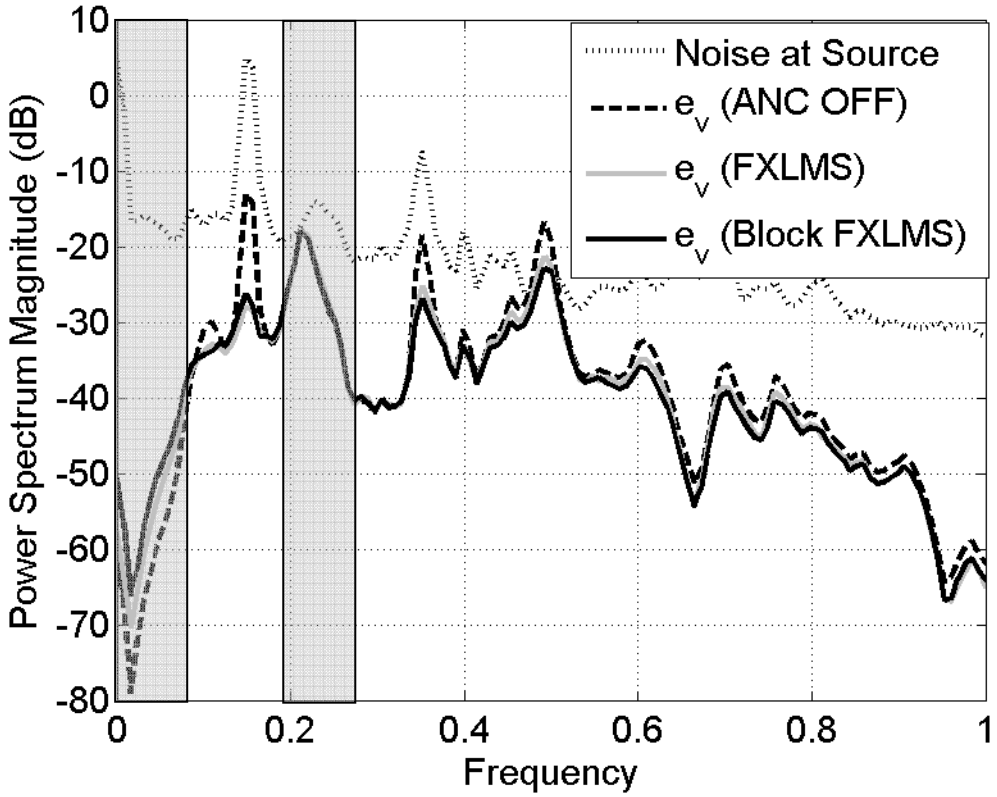


Fig. 9. Comparative spectrums of various noise signals during simulation with air-condition noise. The shaded area represents the spectrum where noise control is the poorest.

C. Experiment-3: Delayed adaptation:

Further experiments were conducted to evaluate the performance of the proposed algorithm in the case of the delayed adaptation of the weights (as discussed in Section V. B). In this case, the change in weights were computed during the time over which the next block of data was received. The change in weights computed using the even block of signal were added to the ANC weights at the end of the odd block time and vice versa. Hence there was a delay in weight adaptation equal to one block sample time ($N-$samples). The delayed adapted algorithm and the original version are named the "Delayed" and "NoDelay" algorithm respectively. Both these algorithms were simulated for the tonal and the recorded air-conditioner noise cases. Figures 10 and 11 present the simulation results of the tonal and recorded air-conditioner noise cases, respectively. In the case of tonal noise, the step-size was selected to be μ=0.002 for both the Delayed and NoDelay algorithms. The step-sizes for the recorded air-conditioner noise case for the Delayed and NoDelay algorithms was μ=0.01. From the mean square error (MSE) plots in Figs. 10 and 11, it can be seen that the convergence speed of delayed adaptation is slightly degraded. However, the mean square noise decreases to the same noise floor in the delayed adaptation of both the tonal and the recorded air-conditioner noise cases.

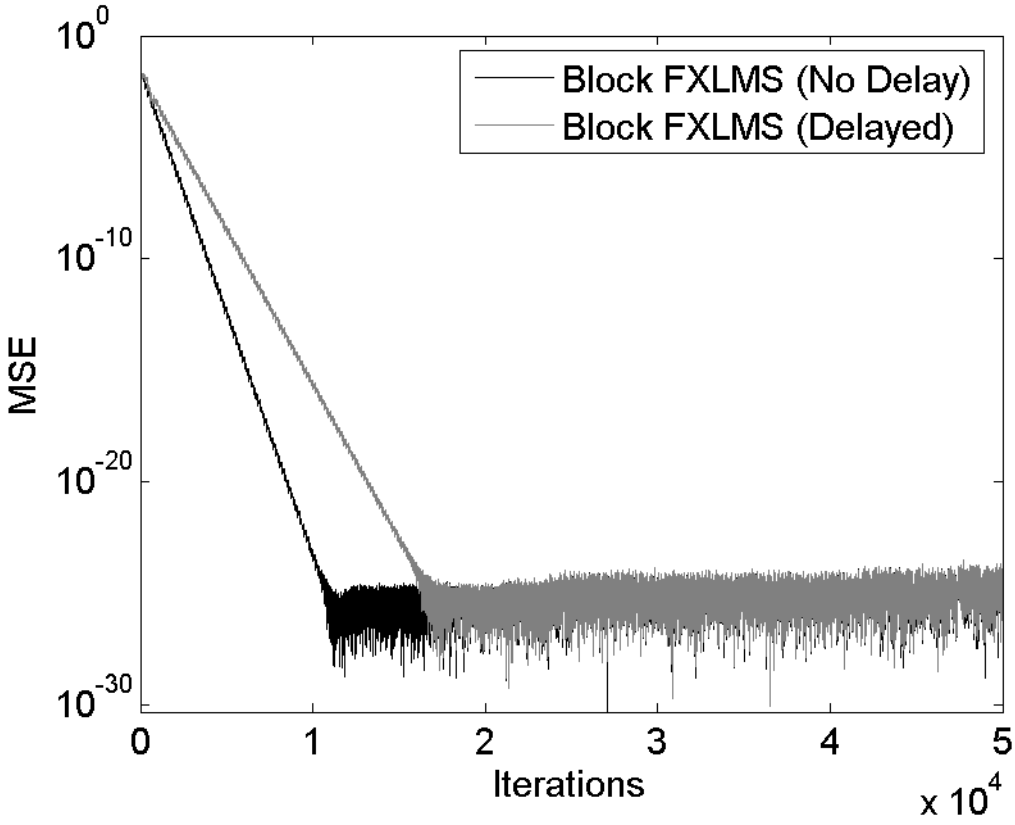
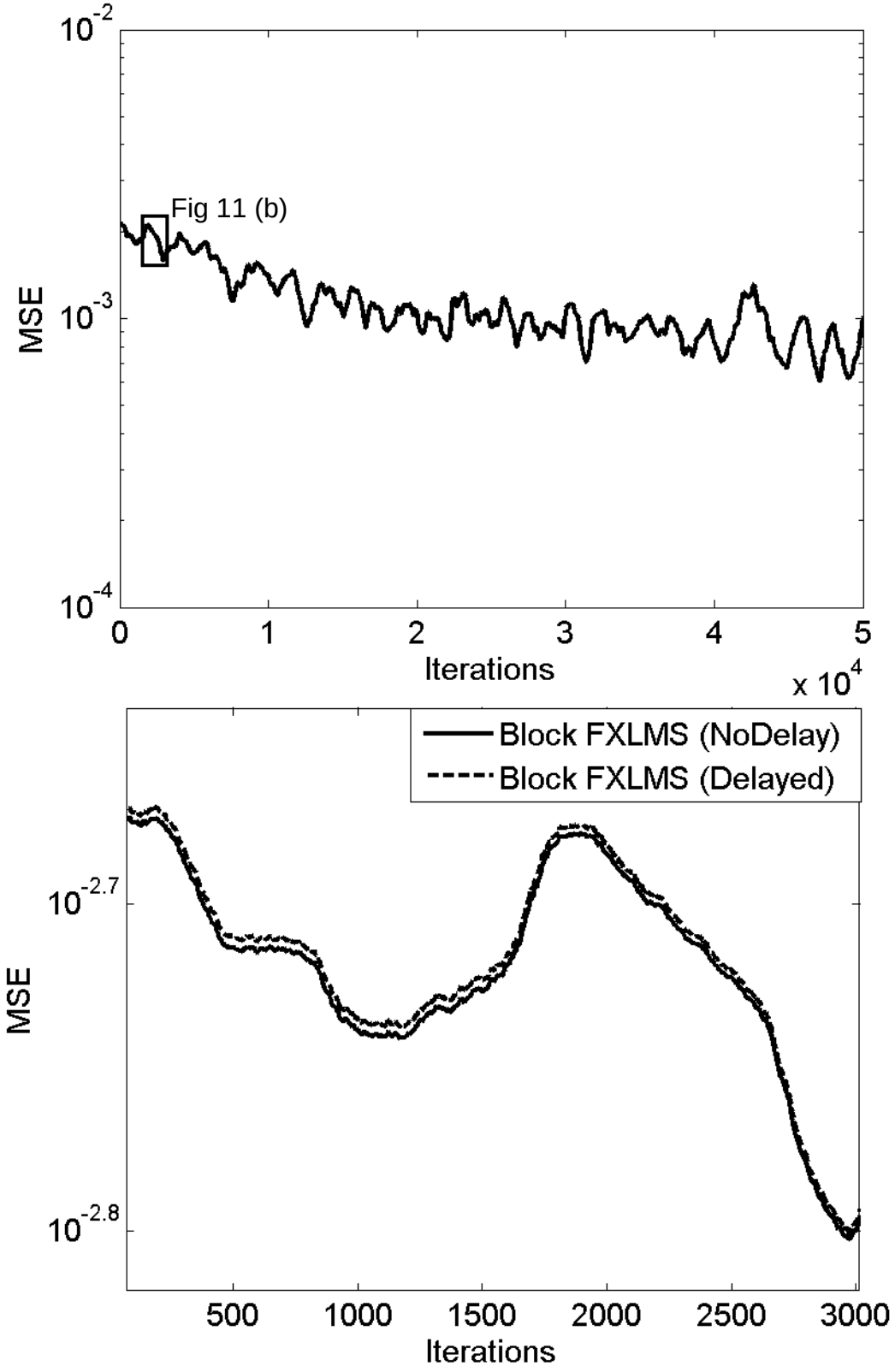


Fig. 10. Convergence performance of MSE with the block frequency-domain FXLMS algorithm for both Delayed and NoDelay adaptation in the case of tonal noise.



(b)

Fig. 11. Convergence performance of MSE with the block frequency-domain FXLMS algorithm for both Delayed and NoDelay adaptation in the case of air-conditioner noise (a) Complete (b) Zoomed portion.

D. *Experiment-4: Estimated Secondary path and the ANC Filter are of different length ($L \le N$):*

In this experiment the estimated acoustic transfer function filters $\hat{S}_p$, $\hat{S}_v$ and $\hat{H}$ are selected to be $L = 128$. However, unlike all the previous experiments (where $L = N = 128$), the length of the ANC filter was 128, 256 and 512 in three independent simulation runs. The primary noise source was the recorded air-conditioner noise. The step-size for all three cases was μ=0.01. The MSE plot of the block frequency-domain FXLMS algorithm is shown in Fig. 12. The zoomed portion of the plot (Fig. 12 (b)) shows that the MSE performance of the longest ANC filter is marginally better than the performance of the shorter ones.

There is benefit to using different size filters for the estimated acoustic paths and the ANC filter. Unlike the previously proposed block frequency-domain algorithms [19-21], this new algorithm provides an opportunity to use higher order filters in the ANC controller. Appending zeros to the shorter estimated acoustic path filters so that they are the same length as the ANC filter (to make $L = N$), as is used in [19-21], unnecessarily introduces extra computational complexity.

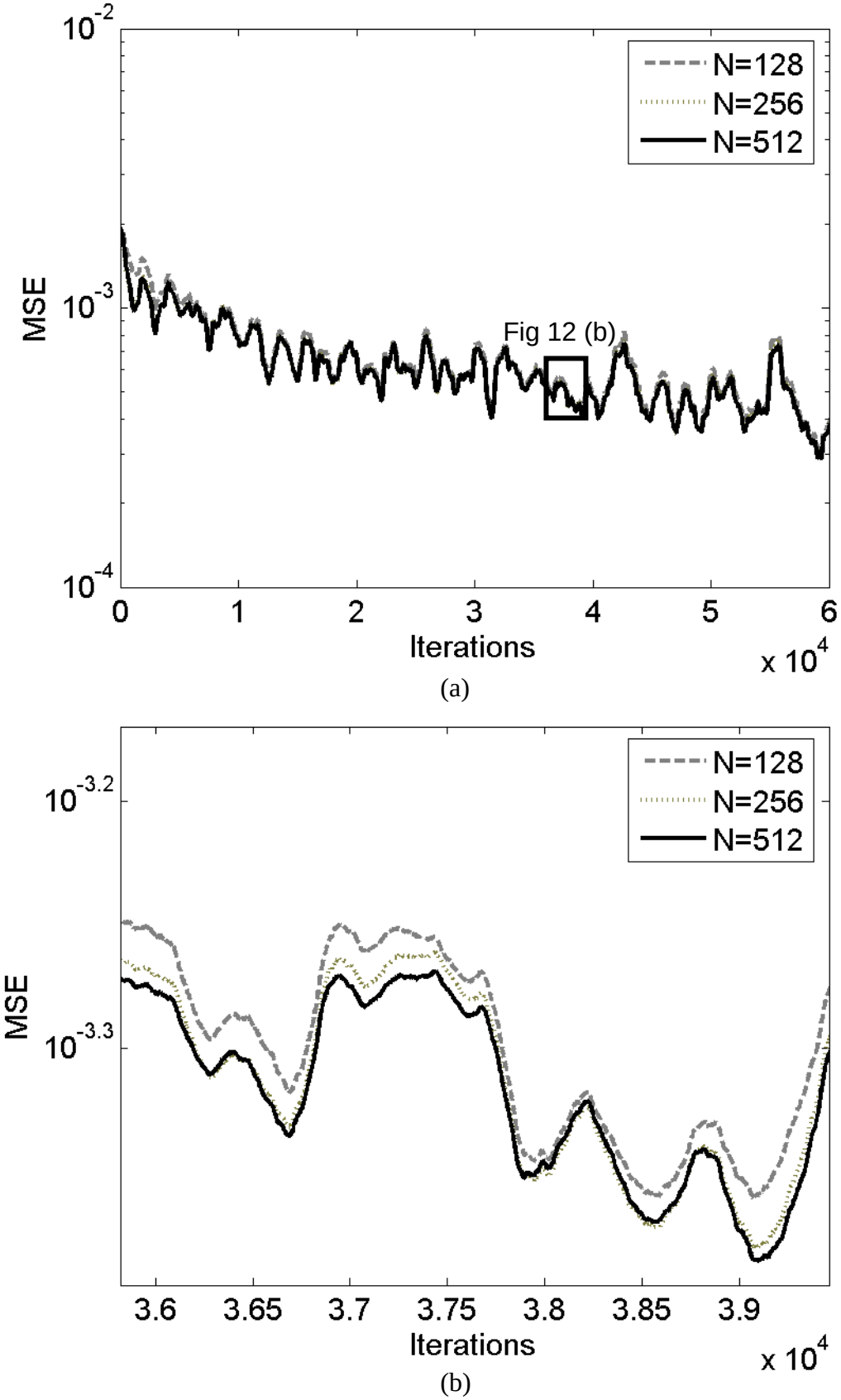


Fig. 12. Convergence performance of MSE with the block frequency-domain FXLMS algorithm for different lengths of the ANC filter (N), by keeping $L = 128$, in the case of air-conditioner noise (a) complete (b) zoomed portion.

VII. CONCLUSIONS

In this paper, frequency-domain implementation of a virtual ANC algorithm has been proposed. The block frequency-domain virtual ANC algorithm was shown to be more computationally and thus energy efficient over its time-domain counterpart. The proposed algorithm was also made delayless, unlike previously proposed frequency-domain ANC algorithms. In addition to this, the proposed algorithm enables use of different filter lengths for the estimated acoustic path transfer functions and the ANC controller. The simulation results demonstrate the performance equivalence of the proposed algorithm with the time-domain one. The computational complexity analysis demonstrated the advantages and bottlenecks of the proposed frequency-domain algorithm. The only disadvantage of the proposed algorithm is the heavy computational complexity at the last sample of the blocks (L th and N th), and this is overcome by any of the three suitable implementation schemes proposed in the paper. The major advantage of the proposed algorithm is that it is more power efficient due to its lower computational effort as the average power consumption is related to the total number of computations over a period of time. The proposed algorithm is therefore more suitable for use in battery operated or power-aware systems like active headsets and mobile phones.

ACKNOWLEDGMENT

D. P. Das acknowledges the financial support provided by Department of Science and Technology, Govt. of India under BOYSCAST Fellowship program to carry out this work at the University of Adelaide. He also acknowledges IMMT, Bhubaneswar and CSIR, India for considering this period of study as deputation and providing full support to him as an employee.

Reference:

[1] S. M. Kuo, and D. R. Morgan, *Active Noise Control Systems, Algorithms and DSP Implementation*, New York: Wiley, 1996.

[2] C. D. Kestell, *Active control of sound in a small single engine aircraft cabin with virtual error sensors*, Ph.D. thesis, School of Mechanical Engineering, The University of Adelaide, SA, 5005, Australia, 2000.

[3] S. Elliott and A. David, "A virtual microphone arrangement for local active sound control," in *Proc. of the 1st International Conference on Motion and Vibration Control*, Yokohama, 1992, pp. 1027-1031.

[4] A. Roure and A. Albarrazin "The remote microphone technique for active noise control," in *Proc. of Active 1999*, Florida, USA, 2000 pp. 1233–1244.

[5] B. Cazzolato, *Sensing systems for active control of sound transmission into cavities*, Ph.D. thesis, School of Mechanical Engineering, The University of Adelaide, SA, 5005, Australia, 1999.

[6] B. Cazzolato "An adaptive LMS virtual microphone," in *Proc. of Active 02*, Southampton, UK, 2002, pp. 105-116.

[7] C. Petersen, R. Fraanje, B. Cazzolato, A. Zander, and C. Hansen "A Kalman filter approach to virtual sensing for active noise control," *Mechanical Systems and Signal Processing*, vol. 22, issue 2, pp. 490-508, 2008.

[8] D. J. Moreau, J. Ghan, B. S. Cazzolato, and A. C. Zander, "Active noise control in a pure tone diffuse sound field using virtual sensing," *Journal of the Acoustical Society of America*, vol. 125, issue 6, pp. 3742-3755, 2009.

[9] S.C. Douglas, "Fast exact filtered-X LMS and LMS algorithms for multichannel active noise control,' in *Proc. IEEE ICASSP-97*., 1997, vol. 1, pp. 399 – 402.

[10] S.C. Douglas, "Fast implementations of the filtered-X LMS and LMS algorithms for multichannel active noise control," *IEEE Trans. on Speech and Audio Process.*, vol. 7, Issue: 4, pp. 454 – 465, 1999.

[11] S.C. Douglas, "An efficient implementation of the modified filtered-X LMS algorithm," *IEEE Signal Process. Lett.*, vol. 4 , issue 10 pp. 286 – 288, 1997.

[12] H.J. Lee, Y.C. Park, C. Lee and D.H. Youn, "Fast active noise control algorithm for car exhaust noise control," *Electronics Letters*, vol. 36, issue 14, pp. 1250 – 1251, 2000.

[13] M. Bouchard and S. Quednau, "Multichannel recursive-least-square algorithms and fast-transversal-filter algorithms for active noise control and sound

reproduction systems," *IEEE Trans. on Speech and Audio Process.*, vol. 8 , issue 5, pp. 606 – 618, 2000.

[14] M. Ferrer, A. Gonzalez, M. de Diego and G. Pinero, "Fast affine projection algorithms for filtered-x multichannel active noise control," *IEEE Trans. on Audio, Speech, and Lang. Process.*, vol. 16, issue 8, pp. 1396 – 1408, 2008.

[15] A. Montazeri and J. Poshtan, "A computationally efficient adaptive IIR solution to active noise and vibration control systems," *IEEE Trans. on Automatic Control*, vol. 55 , issue 11, pp. 2671 – 2676, 2010.

[16] E. P. Reddy, D. P. Das and K. M. M. Prabhu, "Fast adaptive algorithms for active control of nonlinear noise processes," *IEEE Trans. on Signal Process.*, vol. 56, issue 9, pp. 4530 – 4536, 2008.

[17] E. P. Reddy, D. P. Das and K. M. M. Prabhu, "Fast exact multichannel FSLMS algorithm for active noise control" *Signal Processing*, vol. 89, issue 5, pp. 952-956, May 2009.

[18] S.D. Snyder and N. Tanaka, "Algorithm adaptation rate in active control: is faster necessarily better?," *IEEE Trans. on Speech and Audio Process.*, vol. 5 , issue 4, pp. 378 – 381, 1997.

[19] D. P. Das, G. Panda and S. M. Kuo, "New block filtered-X LMS algorithms for active noise control systems," *IET Signal Processing*, vol 1, issue 2, pp. 73-81, June 2007.

[20] D. P. Das, G. Panda, and D. K. Nayak, "Development of frequency-domain block filtered-s LMS (FBFSLMS) algorithm for active noise control system" in *Proc. IEEE Conference ICASSP-2006*, France, 2006, pp. V-289-V292.

[21] P.V. Kumar, K. M. M. Prabhu and D. P. Das, "Block filtered–s LMS algorithm for active control of nonlinear noise systems", *IET Signal Processing*, vol. 4, issue 2, pp. 168 – 180, April 2010.

[22] M. Wu, X. Qiu and G. Chen, "An overlap-save frequency-domain implementation of the delayless subband ANC algorithm," *IEEE Trans. on Audio, Speech, and Lang. Process.*, vol. 16, issue 8 pp. 1706 – 1710, 2008.

[23] A.A. Milani, I.M.S. Panahi and P.C. Loizou, "A new delayless subband adaptive filtering algorithm for active noise control systems," *IEEE Trans. on Audio, Speech, and Lang. Process.*, vol 17 , issue 5, pp. 1038 – 1045, 2009.

[24] J. G. Proakis, D. G. Manolakis, *Digital signal processing: principles, algorithms, and applications*, Pearson Education, Australia, 2007.

[25] http://www.ni.com/compactrio/

[26] K. Moreland and E. Angel "The FFT on a GPU," in *Proc. ACM SIGGRAPH/EUROGRAPHICS conference on Graphics Hardware (2003)* PP. 1-9, 2003 (available http://portal.acm.org/citation.cfm?id=844191).

[27] N. K. Govindaraju, S. Larsen, J. Gray, D. Manocha, "A Memory Model for Scientific Algorithms on Graphics Processors," UNC Tech. Report 2006. pp. 1-10. (available: http://www.cs.unc.edu/~naga/sc06.pdf)