Digital Signal Processing 27 (2014) 119–128

Contents lists available at ScienceDirect

Digital Signal Processing

www.elsevier.com/locate/dsp

Partitioned block frequency domain acoustic echo canceller with fast multiple iterations

Zoran M. Šarić [a,*], Istvan I. Papp [b], Dragan D. Kukolj [b], Ivan Velikić [a], Gordana Velikić [c]

[a] RT-RK d.o.o., Novi Sad, Serbia
[b] Faculty of Engineering, University of Novi Sad, Novi Sad, Serbia
[c] University of Rochester, Rochester, NY, USA

ARTICLE INFO

Article history:
Available online 30 January 2014

Keywords:
Adaptive filtering
Acoustic echo cancellation
Block adaptive filtering
Frequency-domain adaptive filtering
Iterative methods

ABSTRACT

Acoustic echo degrades the quality of speech in hands-free telephony. The most popular digital signal processing technique to suppress acoustic echo is adaptive filtering. However, adaptive filtering may require the computational cost optimization in particular when adaptive algorithm is implemented on low-cost DSP platforms. We propose a computationally efficient version of the partitioned block frequency domain adaptive filter with multiple iterations on current data block. The algorithm performs as a cascade of two adaptive filters. The first filter minimizes the Least Square (LS) criteria leading to unbiased estimate of a room response. The second filter speeds up the convergence rate using multiple iterations to minimize modified LS criterion. Coefficients updates calculated in a single step substitute for multiple iterations and decrease computational costs. The complexity of the algorithm is $o(\log_2(R))$, where R is a number of iterations. The proposed algorithm was tested in a simulated room and a real reverberant room. Tests proved that our algorithm converges faster compared to algorithms described in literature.

1. Introduction

Rapid progress and strong competition in the field of speech communication increased attention of the leading manufactures to the user's speech comfort. The speech comfort refers to both, the high quality of a telephone call and the potential to multi-task during the call. Thus devices that enable users to talk on a phone without holding it, commonly known as hands-free devices, have become unavoidable part of the communication devices toolkits. However, two problems in design of the hands-free devices remain a challenge. First is a problem of a low speech quality caused by the ambient noise and room reverberation. The problem is particularly challenging for a long-range hands-free devices where the distance between a speaker's mouth and a microphone may run up to five meters [1]. Standard methods for an ambient noise reduction [2–4] are efficient in cases of a stationary ambient noise or when the power spectral density of the ambient noise changes slow in time. Designers often employ microphone arrays and beam forming techniques [5–7] to suppress a non-stationary ambient noise and to reduce a room reverberation. Second common problem is acoustic echo, frequently solved by adaptive filtering techniques [8–20]. Plot in Fig. 1 shows a block diagram of the hands-free audio terminal with an acoustic echo canceller (AEC). The ITU-T G.167 [8] standard sets cumulative acoustic echo attenuation between receiving point RCV_in and sending point SND_out, also referred as a terminal coupling loss (TCL), to over 45 dB [8]. Since an AEC module provides up to 35 dB echo attenuation, the additional attenuation is achieved by a beam-former (BF) [9–11], and an automatic gain control (AGC) module [10–12].

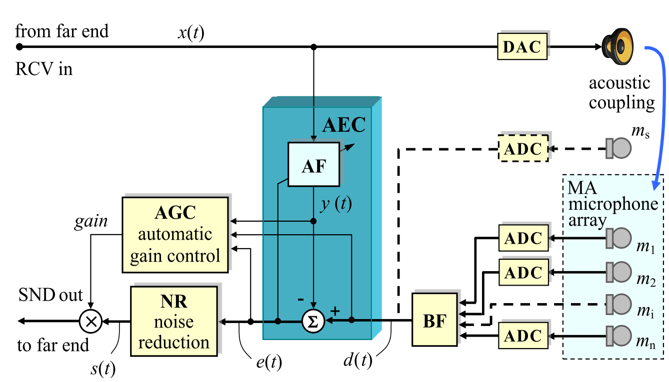

Fig. 1. Signal processing modules in hands-free voice terminal: single microphone m_s or microphone array (MA) for multi-microphone case, beam former (BF), acoustic echo canceller (AEC), noise reduction module (NR), automatic gain control (AGC).

An adaptive filter can be implemented in time or frequency domain [13,14]. Time domain adaptive filtering enables low latency

* Corresponding author. Tel.: +381 11 3208540.
E-mail addresses: zoran.saric@rt-rk.com (Z.M. Šarić), istvan.papp@rt-rk.com (I.I. Papp), dragan.kukolj@rt-rk.com (D.D. Kukolj), ivan.velikic@rt-rk.com (I. Velikić), gvel@ece.rochester.edu (G. Velikić).

1051-2004/$ – see front matter
http://dx.doi.org/10.1016/j.dsp.2014.01.008

at the expense of high computational demands [14]. Frequency domain adaptive filtering has a lower computational cost, with a latency equal to double FFT data block length. Often, a long impulse response is divided into a finite number of partitions [15–18], to reduce latency N_p times, where $N_p = N/P$, N is a length of the impulse response of adaptive filter, and P is the partition length [18]. A convergence rate of an adaptive algorithm can be improved by multiple iterations performed over current data block. Such algorithm is known as the partitioned block frequency domain row action projection algorithm (PBFDRAP) [18]. In spite of a considerable improvement of the computational efficiency of the algorithm, its computational complexity remains high for a large number of iterations. For instance, versions 2 and 3 of the fast PBFDRAP algorithms described in [18], need $4+2R$ FFT transforms per data block, where R is a desired number of iterations. For a typical adaptive filter such as the one specified in Subsection 4.4 (also in Tests conditions, Subsection 5.1), with a number of iterations R between 5 and 10, the computational load is between 12 and 18 million instructions per second (MIPS) (see Fig. 3). Other fast versions of the PBFDRAP algorithm have similar or higher computational load.

Nonlinear PBFDRAP algorithms using a simplified Volterra filter were derived in [19,20]. Computational complexity of these algorithms was considerably reduced compared to the time-domain adaptive filters, however, the number of operations is too high to implement on low-cost digital signal processors as their complexity is order $O(M^2)$ in term of number of filter taps M [19,20].

In this paper we propose a computationally efficient version of the partitioned block frequency domain adaptive filter with multiple iterations on current data block. We named the algorithm as the *partitioned block frequency domain approximated row action projection* (PBFD-ARAP). In order to reduce computational costs, coefficients' updates are calculated in a single step. Thus, computational complexity of the algorithm is almost constant in terms of the number of iterations. Note that coefficients calculated using single step updates are the same as coefficients calculated through multiple iterations.

Generally speaking, we can use *a priori* or *a posteriori* error as an output of the echo canceller. *A priori* error is obtained by filter coefficients before their update, while *a posteriori* error is obtained by filter coefficients after their update. *A priori* error is traditionally used in adaptive filtering because it preserves the speech quality. *A posteriori* error provides desirable extra echo cancellation [18, 20], but introduces speech degradation. Our intention is to exploit good properties of both errors. To obtain higher echo attenuation in a single talk case we propose *a posteriori* error approach. However, to avoid near end speech distortion in a double talk case or only near end presence, we propose *a priori* error approach. Voice activity detector (VAD) controls the switching between *a priori* and *a posteriori* error.

In Sections 2 and 3 we described existing PBFDAF and PBFDRAP algorithms as starting points of our fast version of the PBFDRAP. In Subsections 4.1 and 4.2 we developed the PBFD-ARAP algorithm. In Subsection 4.3 we analyzed performance and asymptotic properties of proposed algorithm. In Subsection 4.4 we analyzed computational cost of proposed algorithm. Experimental tests are presented in Section 5. Performance of the algorithm was evaluated by the echo return loss enhancement (ERLE) measure under two experimental conditions. In the first experimental setup we simulated a real office room by the image method [21]. Under absolutely controlled conditions we compared convergence rates of proposed PBFD-ARAP algorithm and existing algorithms PBFDAF and PBFDRAP [18]. In the second experimental setup we tested proposed PBFD-ARAP algorithm using real signals, which are recorded in an office room by specially designed DSP platform. In both experimental scenarios convergence of proposed PBFD-ARAP algorithm was significantly faster compared to the PBFDAF algorithm. Computational demand of proposed PBFD-ARAP algorithm was considerably lower compared to the PBFDRAP algorithm with approximately same convergence rate.

2. Partitioned Block Frequency Domain Adaptive Filter (PBFDAF)

We will briefly describe the PBFDAF algorithm from [18] to preserve the consistency with the notation used in this paper. Fig. 1 depicts a block diagram of a typical signal processing in a hands-free voice terminal. A desired signal $d(t)$ is an output of either a single microphone m_s, or a beam former (BF) of a microphone array (MA). Noise reduction module (NR) and automatic gain control (AGC) process the output of the acoustic echo canceller (AEC).

The output of the adaptive filter $y(t)$ is

$$y(t) = \sum_{i=0}^{N-1} h_i x(t-i), \tag{1}$$

where $x(t)$ is a reference signal from far end, h_i, $i = 0, \dots, N-1$ are coefficients of adaptive FIR filter, and N is a FIR filter length. Coefficients of the adaptive filter are estimated using the least squares criterion $V([h_0, \dots, h_{N-1}])$,

$$V\big([h_0, \dots, h_{N-1}]\big) = E\big\{e(t)^2\big\}, \quad e(t) = d(t) - y(t), \tag{2}$$

where E is the expectation operator, $d(t)$ is a desired (microphone) signal and $e(t)$ is an error signal.

If N is divisible by a natural number P, the filter impulse response can be divided into $N_p = N/P$ partitions, and we can rewrite (1) by

$$y(t) = \sum_{p=0}^{N_P-1} \sum_{i=0}^{P-1} h_p(i) x(t-i-pP), \tag{3}$$

where $p = 0, \dots, N_p - 1$ is a partition index, P is a number of coefficients in each partition, $h_p(i)$ is ith FIR filter coefficient of pth partition. Signal processing is implemented in the DFT domain using an overlap-save method on data blocks with equal length L. We will mark index of current data block by n, and for simplicity sake put $P = L$. Then an estimate of the FIR filter coefficient vector of pth partition on nth data block is

$$\hat{\boldsymbol{h}}_p^{(n)} \overset{\forall p}{=} \begin{bmatrix} \hat{h}_p^{(n)}(0) \\ \vdots \\ \hat{h}_p^{(n)}(P-1) \end{bmatrix}, \quad p = 0, \dots, N_p - 1.$$

The DFT of pth partition of FIR filter can be calculated by the FFT with M points. In matrix notation it is

$$\boldsymbol{H}_p^{(n)} \overset{\forall p}{=} \mathbf{F} \begin{bmatrix} \hat{\boldsymbol{h}}_p^{(n)} \\ \mathbf{0} \end{bmatrix} \begin{matrix} \updownarrow P \\ \updownarrow M-P, \end{matrix} \tag{4}$$

where $\mathbf{F}$ is M by M DFT transformation matrix. According to the overlap-save filtering method M has to be greater or equal to $P + L - 1$. Index of the last sample in current nth block is $t = (n+1)L$. In accordance with the PBFDAF algorithm [18] we do the following steps:

for each data block n do

$t = (n+1)L$,

$$\boldsymbol{X}_p^{(n)} \overset{\forall p}{=} diag\left\{ \mathbf{F} \begin{bmatrix} x(t-pP-M+1) \\ \vdots \\ x(t-pP) \end{bmatrix} \right\} \begin{matrix} \overline{\uparrow} \\ M \\ \underline{\downarrow} \end{matrix}, \tag{5}$$