Proceedings of the 7th International Conference on Applied Informatics
Eger, Hungary, January 28–31, 2007. Vol. 2. pp. 335–342.

Problems of acoustic echo cancellation

Kálmán Abari, Gábor Páli

Department of Computer Science, Faculty of Informatics
University of Debrecen
e-mail: {abarik,pg0003}@delfin.unideb.hu

Abstract

The need for clean and clearly audible communication channels is still growing nowadays, so we have started to investigate whether a widely known codec optimized for speech, titled Speex, is able to take up this challenge. Because Speex is freely available and open for any kind of experimentations, and because it incorporates a relatively contemporary and acceptable approach of acoustic echo cancellation by now, it would be desirable to push its possibilities and efficiency to the maximal rate. This is also suggested by the fact that many programmers and so real-life applications are already building upon Speex as a popular choice. Nevertheless, it is a great summation of the fresh improvements achieved in this field.

Our experiments and observations are about to give ideas to make the echo cancellation algorithm built in Speex more accurate.

Keywords: acoustic echo cancellation, multidelay block frequency filtering (MDF), speech enhancement, speech processing, adaptive filtering

MSC: 68U99 (Computing Methodologies and Applications)

1. Introduction

The success and fitness of adaptive filter models for nearly proper speech enhancement methods cannot be disregarded easily, because many researchers in the field think filtering noise or echo is about to refine a mathematical filter constructed upon the actual behaviour of the undesired components embedded into the speech signal. Although it is a very promising idea and a logical way to address speech enhancement issues emerged in informatics nowadays, there is no guarantee of satisfactory results.

One of the most disturbing defects can be found in speech signals is the acoustic echo that is generated through a telephone set when sounds directly picked up within the location of the microphone, reflect on objects at that location, and again, with a certain amount of delay, by the microphone. The sounds may initiate from either party as well as from any other ambient noise in the location, including

the telephone set speaker. In this paper, we would like to introduce a free software, Speex, a patent-free audio compression format designed for speech, coping with the topic by its built-in acoustic echo cancellation mechanism from version 1.2 [6], and present an improvement for it.

The organisation of the paper is as follows. In Section 2, we introduce concepts, notations and the implementation that are used to prepare the explanation of our contribution. Then this will be developed into a practical recipe to extend their capabilities in Section 3. Finally, the achieved results are presented in Section 4, supported by the conclusions and comments explained in Section 5.

2. The MDF adaptive filter

A possible and popular way of cancelling acoustic echoes is to employ adaptive (or self-adjusting) filters, which adjust its transfer function defined by an optimizing algorithm. Because of the complexity of these algorithms, most of the adaptive filters are digital filters that perform digital signal processing to adapt their performance by tracking the input signal. The basic idea behind the use of this type of variable filters to extract an estimate of the desired (noiseless/echoless) signal. For the general model we take the following assumptions:

– the input signal $x(n)$ is the sum of a desired signal $d(n)$ and interfering noise $v(n)$:

$$x(n) = d(n) + v(n) \tag{2.1}$$

– the filter has a Finite Impulse Response structure. For such structures the impulse response is equal to the filter coefficients. The coefficients for a filter $\mathbf{w}_n$ of order p are defined as:

$$\mathbf{w}_n = [w_n(0),\ w_n(1),\ \cdots,\ w_n(p)]^T \tag{2.2}$$

– the error signal $e(n)$ or cost function is the difference between the $d(n)$ desired and the estimated $\hat{d}(n)$ signal:

$$e(n) = d(n) - \hat{d}(n) \tag{2.3}$$

– the filter estimates the desired signal by convolving the input signal with the impulse response. In vector form it is expressed as:

$$\hat{d}(n) = \mathbf{w}_n^T \mathbf{x}(n) \tag{2.4}$$

where $\mathbf{x}(n) = [x(n),\ x(n-1),\ \ldots,\ x(n-p)]^T$ is an input signal vector. The filter updates its coefficients at every time instant

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \Delta\mathbf{w}_n \tag{2.5}$$

where $\Delta\mathbf{w}_n$ is a correction factor. In brief, the task of the adaptive algorithm is to generate this correction factor based on the input and error signals.

The investigated adaptive filter implementation is the multidelay block frequency domain adaptive filter algorithm [4]. It is based on the fact that convolutions required to filter the signal may be efficiently computed in the frequency domain by the Fast Fourier Transform. However, to bag all the benefits offered by this solution the block size used should be smaller than the filter length. In the context of this algorithm we will use the following definitions and notations (according to [1]):

$x(n)$ = far-end signal/speech,

$w(n)$ = ambient (background) noise,

$v(n)$ = near-end signal/speech (double-talk),

$\mathbf{x}(n) = [x(n) \ \cdots \ x(n-L+1)]^T$, excitation vector,

$\mathbf{y}(n) = \mathbf{h}^T\mathbf{x}(n) + w(n) + v(n)$, echo + ambient noise + near-end signal,

$\mathbf{h} = [h_0 \ \cdots \ h_{L-1}]^T$, vector representing the echo path,

$\hat{\mathbf{h}}(n) = [\hat{h}_0(n) \ \cdots \ \hat{h}_{L-1}(n)]^T$, estimated echo path vector,

$\hat{y}(n) = \hat{\mathbf{h}}^T(n-1)\mathbf{x}(n)$, estimated echo,

$e(n) = y(n) - \hat{y}(n)$, error signal

where n is the sample-by-sample time index and L is the length of the adaptive filter that we suppose to be equal to the length of the echo path. L is an integer multiple of N, i. e. $L = KN$. We define the block error signal (of length $N \leqslant L$) as:

$$\mathbf{e}(m) = \mathbf{y}(m) - \hat{\mathbf{y}}(m) \tag{2.6}$$

where m is the block time index and

$$\begin{array}{rcl} \mathbf{e}(m) & = & [e(mN) \ \cdots \ e(mN+N-1)]^T, \\ \mathbf{y}(m) & = & [y(mN) \ \cdots \ y(mN+N-1)]^T, \\ \mathbf{X}(m) & = & [\mathbf{x}(mN) \ \cdots \ \mathbf{x}(mN+N-1)], \\ \hat{\mathbf{y}}(m) & = & [\hat{y}(mN) \ \cdots \ \hat{y}(mN+N-1)]^T \\ & = & \mathbf{X}^T(m)\hat{\mathbf{h}}. \end{array} \tag{2.7}$$

It can be easily checked that $\mathbf{X}$ is a Toeplitz matrix of size $L \times N$. It can be also shown that

$$\hat{\mathbf{y}}(m) = \sum_{k=0}^{K-1} \mathbf{T}(m-k)\hat{\mathbf{h}}_k, \tag{2.8}$$

where

$$\mathbf{T}(m-k) = \begin{bmatrix} x(mN-kN) & \cdots & x(mN-kN-N+1) \\ x(mN-kN+1) & \ddots & \vdots \\ \vdots & \ddots & \vdots \\ x(mN-kN+N-1) & \cdots & x(mN-kN) \end{bmatrix} \tag{2.9}$$

is an $N \times N$ Toeplitz matrix and

$$\hat{\mathbf{h}}_k = [\hat{h}_{kN},\ \hat{h}_{kN+1},\ \cdots,\ \hat{h}_{kN+N-1}]^T,\ k = 0,\ 1,\ \ldots,\ K-1, \tag{2.10}$$

are the subfilters of $\hat{\mathbf{h}}$. In (2.8), the filter $\hat{\mathbf{h}}$ (of length L) is partitioned in K subfilters $\hat{\mathbf{h}}_k$ of length N and the rectangular matrix $\mathbf{X}^T$ (of size $N \times L$) is decomposed in K square submatrices of size $N \times N$.

It is known that a Toeplitz matrix $\mathbf{T}$ can be transformed – by doubling its size – to a circulant matrix

$$\mathbf{C} = \begin{bmatrix} \mathbf{T}' & \mathbf{T} \\ \mathbf{T} & \mathbf{T}' \end{bmatrix}, \tag{2.11}$$

where $\mathbf{T}'$ is also a Toeplitz matrix. This circulant matrix is can be decomposed as follows:

$$\mathbf{C} = \mathbf{F}^{-1}\mathbf{D}\mathbf{F}, \tag{2.12}$$

where $\mathbf{F}$ is the Fourier matrix (of size $2N \times 2N$) and $\mathbf{D}$ is a diagonal matrix whose elements are the discrete Fourier transform of the first column of $\mathbf{C}$.

Finally, it is comfortable to define the frequency-domain quantities

$$\underline{\mathbf{y}}(m) = \mathbf{F}\begin{bmatrix} \mathbf{0}_{N\times 1} \\ \mathbf{y}(m) \end{bmatrix},\ \underline{\hat{\mathbf{h}}}_k(m) = \mathbf{F}\begin{bmatrix} \hat{\mathbf{h}}_k(m) \\ \mathbf{0}_{N\times 1} \end{bmatrix},\ \underline{\mathbf{e}}(m) = \mathbf{F}\begin{bmatrix} \mathbf{0}_{N\times 1} \\ \mathbf{e}(m) \end{bmatrix} \tag{2.13}$$

to give the MDF adaptive filter by the following equations:

$$\underline{\mathbf{e}}(m) = \underline{\mathbf{y}}(m) - \mathbf{G}^{01}\sum_{k=0}^{K-1}\mathbf{D}(m-k)\underline{\hat{\mathbf{h}}}_k(m-1) \tag{2.14}$$

$$\mathbf{S}_{MDF}(m) = \lambda\mathbf{S}_{MDF}(m-1) + (1-\lambda)\mathbf{D}^*(m)\mathbf{D}(m) \tag{2.15}$$

$$\underline{\hat{\mathbf{h}}}_k(m) = \underline{\hat{\mathbf{h}}}_k(m-1) + \mu\mathbf{G}^{10}\mathbf{D}^*(m-k) \times [\mathbf{S}_{MDF}(m) + \delta\mathbf{I}_{2N\times 2N}]^{-1}\underline{\mathbf{e}}(m) \tag{2.16}$$

where $k = 0,\ 1,\ \ldots,\ K-1$, $*$ denotes complex conjugate, λ ($0 \ll \lambda < 1$) is an exponential forgetting factor, μ ($0 < \mu \leqslant 2$) is a positive number, δ is a regularization parameter, and

$$\begin{aligned} \mathbf{G}^{01} &= \mathbf{F}\mathbf{W}^{01}\mathbf{F}^{-1}, \quad \mathbf{W}^{01} = \begin{bmatrix} \mathbf{0}_{N\times N} & \mathbf{0}_{N\times N} \\ \mathbf{0}_{N\times N} & \mathbf{I}_{N\times N} \end{bmatrix}, \\ \mathbf{G}^{10} &= \mathbf{F}\mathbf{W}^{10}\mathbf{F}^{-1}, \quad \mathbf{W}^{10} = \begin{bmatrix} \mathbf{I}_{N\times N} & \mathbf{0}_{N\times N} \\ \mathbf{0}_{N\times N} & \mathbf{0}_{N\times N} \end{bmatrix}. \end{aligned} \tag{2.17}$$

3. Enhancing multidelay block filtering in speex

Speex is an Open Source/Free Software patent-free audio compression format designed for speech. The Speex Project aims to lower the barrier of entry for voice applications by providing a free alternative to expensive proprietary speech

codecs. Moreover, Speex is well-adapted to Internet applications and provides useful features that are not present in most other codecs.

Starting from version 1.2, Speex incorporates an experimental implementation of the MDF algorithm presented above, hence providing Acoustic Echo Cancellation [6]. Although it is mainly built upon the formulas sketched in Section 2, its author went ahead and applied numerous enhancements to this algorithm, including the optimisation of the learning rate (μ in 2.16) discussed in [5]. However, it is important to mention, that there is an other difference from the canonical algorithm – a missing link in the theory of the MDF built-in Speex – the insertion of a so-called *proportional adaption rate* into (2.16).

Let us summarise these key modifications in an equational form:

$$\underline{\hat{\mathbf{h}}}_k(m) = \underline{\hat{\mathbf{h}}}_k(m-1) + \mathbf{G}^{10}\mathbf{M}_{opt}(m)p_k(m)\mathbf{D}^*(m-k)[\mathbf{S}_{MDF}(m) + \delta\mathbf{I}_{2N\times 2N}]^{-1}\underline{\mathbf{e}}(m) \quad (3.1)$$

where $p_k(m)$ is the proportional adaption rate of the kth block and $\mathbf{M}_{opt}(m)$ is a diagonal matrix whose elements are the optimal frequency-dependent learning rates. In [5], this matrix also depends on the frame index, however, it is not present in the implementation.

Because it has never been discussed theoretically before, we must assume that is only a rather experimental and practical fine tune of the basic algorithm, but not without a price. It has a sensible impact on the performance, especially on the adaptation rate of the filtering. Our suspicions seem to be supported by the fact that there is appeared a separate subroutine for adjusting this proportional adaptation rate in the late versions of Speex. Although, the first pick of the author, to initialize these values to a constant independently from the iteration step m, was proved to be wrong, and lately he had chosen the same way as us (almost in the same time): started to tweak this tiny missing link.

3.1. Tweaking the missing link

The proportional adaption rate – as a heuristical correction in the implementation – helps to recover values computed by the pure mathematical model when they are failed, because the model is prone to diverge in certain situations. Originally, values of p_k are exponentially decreasing weights for $k = 0, \ldots, K-1$. Their task is to eliminate the fluctuations in the gradient and make the original algorithm robust.

However, it weakens the quality of adaptation, hence we propose two different approaches to solve this problem. Both of them modifies the gradient component lifted from (3.1) as

$$\underline{\mathbf{g}}_k(m) = \mathbf{D}^*(m-k)[\mathbf{S}_{MDF}(m) + \delta\mathbf{I}_{2N\times 2N}]^{-1}\underline{\mathbf{e}}(m). \quad (3.2)$$

3.1.1. Effective $\mathbf{p_k(m)}$

First, choose the actual $p_k(m)$ values so that they enable the gradient components per block to track the average energy distribution of the subfilters. In the next step, we have to introduce $\hat{\gamma}_k$, the average of those energy distributions, that satisfies the conditions $\sum_{k=0}^{K-1} \hat{\gamma}_k(m) = 1$ and $\hat{\gamma}_k(m) \geqslant 0$. In the beginning, let

$$\hat{\gamma}_k(0) = \frac{1}{K}, \tag{3.3}$$

then

$$\hat{\gamma}_k(m) = \lambda \hat{\gamma}_k(m-1) + (1-\lambda) \frac{\gamma_k(m)}{\sum_{k=0}^{K-1} \gamma_k(m)} \tag{3.4}$$

where

$$\gamma_k(m) = \underline{\mathbf{h}}_k(m)^{*T} \underline{\mathbf{h}}_k(m). \tag{3.5}$$

In order to determine the value of $p_k(m)$ in (3.1), we can use the following formulation based on the previous statements:

$$\hat{\gamma}_k(m) = p_k^2(m) \frac{\gamma_k(m)}{\sum_{k=0}^{K-1} \gamma_k(m)} \tag{3.6}$$

3.1.2. Modified gradient

Or we can modify the gradient component instead of the p_k values to make it adaptive, therefore reclaim the lost benefits. This gradient has a property that assumes

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbf{D}(m-k) \underline{\mathbf{g}}_k(m) \;\approx\; \underline{\mathbf{e}}(m) \tag{3.7}$$

because

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbf{D}(m-k) \mathbf{D}^*(m-k) \;\approx\; \mathbf{S}_{MDF}(m) + \delta \mathbf{I}_{2N \times 2N} \tag{3.8}$$

holds. In connection with (2.15) this equation holds in the most of the cases.

Introduce a modified gradient

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbf{D}(m-k) \left(\underline{\mathbf{p}}_k(m-1) + \Delta \underline{\mathbf{p}}_k(m-1) \right) \;\approx\; \underline{\mathbf{e}}(m) \tag{3.9}$$

where $\underline{\mathbf{p}}_k(m-1)$ is an approximation of the actual gradient $\underline{\mathbf{g}}_k(m)$ and $\Delta \underline{\mathbf{p}}_k(m-1)$ is such an additive component that helps to satisfy the property described above.

For this reason, we choose

$$\begin{aligned} \Delta \underline{\mathbf{e}}(m) &= \underline{\mathbf{e}}(m) - \tfrac{1}{K} \textstyle\sum_{k=0}^{K-1} \mathbf{D}(m-k) \underline{\mathbf{p}}_k(m-1) \\ \Delta \underline{\mathbf{p}}_k(m-1) &= [\mathbf{S}_{MDF}(m) + \mathbf{D}^*(m-k) \delta \mathbf{I}_{2N \times 2N}]^{-1} \Delta \underline{\mathbf{e}}(m) \end{aligned} \tag{3.10}$$

and $\underline{\mathbf{p}}_k(m)$ $(k = 0, \ldots, K-1)$ is initialized to zero. It could be computed recursively as

$$\underline{\mathbf{p}}_k(m) = \underline{\mathbf{p}}_k(m-1) + \mu \Delta \underline{\mathbf{p}}_k(m-1), \tag{3.11}$$

where μ is a constant learning rate $(0 < \mu \ll 1)$.

Based on this derivation, (3.1) could be rewritten in the following form:

$$\hat{\underline{\mathbf{h}}}_k(m) = \hat{\underline{\mathbf{h}}}_k(m-1) + \mathbf{G}^{10}\mathbf{M}_{opt}(m) \left(\underline{\mathbf{p}}_k(m-1) + \Delta \underline{\mathbf{p}}_k(m-1) \right). \tag{3.12}$$

4. Results

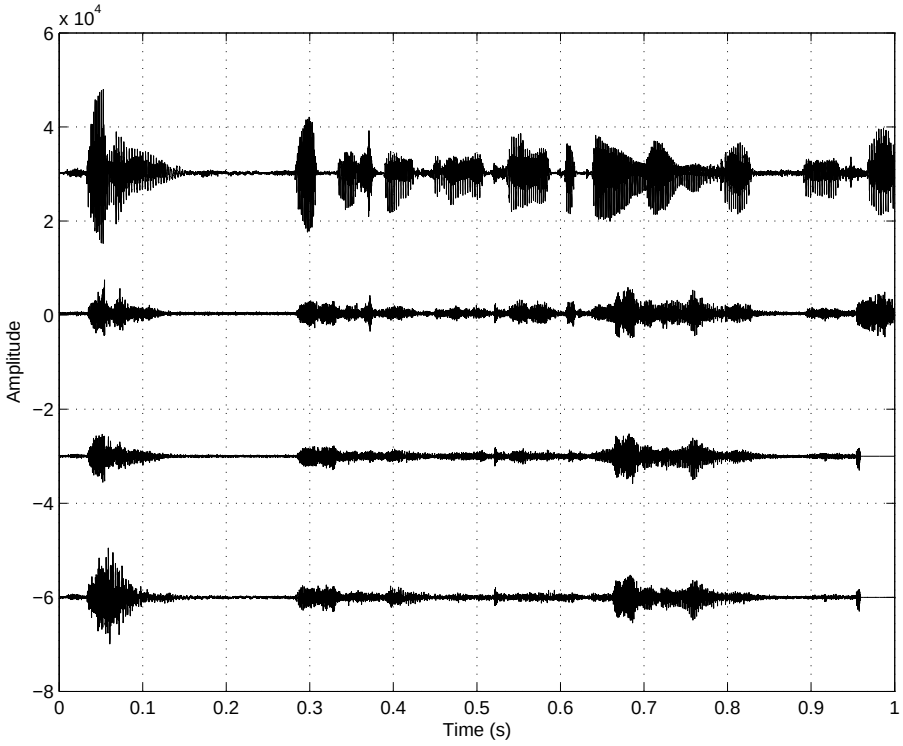

Figure 1: Comparison of different echo cancellation techniques

We also re-implemented the Speex version of the MDF algorithm (revision of Nov 5, 2006) in MATLAB and investigated the alternatives of the possible enhancements in a more abstract form to see the affected relationships clearly.

Figure 1 represents one of the results of our experiments in brief. In this experiment we have applied the echo cancellation algorithm based on MDF implemented

in Speex 1.2beta1 ([6]) and then we have modified this implementation to adjust the $p_k(m)$ values dynamically in response to the changes in the signal. The following dimensions were analysed (starting from the above): far-end, input, output computed by dynamic adaption rate (as described in 3.1), output computed by static adaption rate (in Speex).

There can be easily seen that replacing the static proportional rate with a dynamic one causes faster adaptation (by about 100 ms) and it is also good at keeping this relative quick reaction time in the later segments.

5. Conclusion

In this article, we have proposed an improvement method for the Acoustic Echo Cancellation applied in the open source speech codec, Speex. We have found that the quality of this function could be increased by changing the implemented static proportional adaptation rate to dynamic, and we gave the necessary formal framework to achieve that. Furthermore, we have also demonstrated the tenability of our work in Section 4, based on our experiments.

In future work, several other and more specific ways of extension or replacement of $p_k(m)$ could be described and evaluated, taking the conditions and assumptions listed in 3.1 into account. In our opinion, the differences between practice and theory should be minimalised, and the divergence factors of this method should be researched and explained formally in details.

References

[1] Benesty, J., Gänsler, T., A Multidelay Double-Talk Detector Combined with the MDF Adaptive Filter, *EURASIP Journal on Applied Signal Processing*, 2003:11 (2003), 1056–1063.

[2] Hayes, M. H., Statistical Digital Signal Processing and Modeling, *Wiley and Sons* (1996).

[3] Haykin, S., Adaptive Filter Theory, *Prentice Hall* (2002).

[4] Soo, J.-S., Pang, K. K., Multidelay Block Frequency Domain Adaptive Filter, *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 38, no. 2 (1990), 373–376.

[5] Valin, J.-M., On Adjusting the Learning Rate in Frequency Domain Echo Cancellation with Double-Talk, *IEEE Transactions on Audio, Speech & Language Processing* (2006).

[6] Valin, J.-M., The Speex Codec Manual (version 1.2-beta1), `http://www.speex.org` (2006).