A PROPORTIONATE AFFINE PROJECTION ALGORITHM USING FAST RECURSIVE FILTERING AND DICHOTOMOUS COORDINATE DESCENT ITERATIONS

Felix Albu[1,2]

[1]University Politehnica of Bucharest, [2]Valahia University of Targoviste
felix_albu@ieee.org

ABSTRACT

Recently, a new proportionate-type APA called MIPAPA was developed, taking into account the "history" of the proportionate factors. The use of a fast recursive filtering procedure together with the dichotomous coordinate descent (DCD) method is proposed for MIPAPA. Simulation results indicate that the proposed algorithm has similar performance as the original algorithm with minimum performance losses.

Keywords: *echo cancellation, adaptive filter*

1. INTRODUCTION

The main part of the echo cancellation application can be interpreted as a system identification problem, where an adaptive filter is used to identify an unknown system, i.e., the echo path. Numerous proportionate-type affine projection algorithms (see [1-3]) were proposed. The common idea of this kind of algorithms is to update each coefficient of the filter independently of the others, by adjusting the adaptation step-size in proportion to the magnitude of the estimated filter coefficient.

One of the first proportionate-type algorithms was the proportionate normalized least-mean-square (PNLMS) algorithm proposed by Duttweiler [1], in the context of network echo cancellation. Following this approach, several versions of the PNLMS algorithm were developed (e.g., see [2-3] and the references therein), some of them being suitable for both network and acoustic echo cancellation, e.g., the improved PNLMS (IPNLMS) algorithm [3]. Besides, since the affine projection algorithm (APA) [4] and different versions of it are frequently used for echo cancellation [5]–[8]. Proportionate-type APAs were a natural extension of PNLMS algorithms and were attractive mainly for their fast convergence rate and tracking. Recently, new proportionate-type APA were proposed, called "memory"-IPAPA (MIPAPA) [9] and AMIPAPA respectively [10]. In Section 2 a presentation of the dichotomous coordinate descent (DCD) method is made. The DCD method was first time used in an affine projection algorithm proposed in 2005 [7]. Three years later, a fast recursive filtering proved useful in reducing the complexity of the affine projection algorithm [8]. In this paper, a proportionate APA using previously mentioned fast recursive filtering and DCD methods, called FMIPAPA-DCD, is presented in Section 3. Simulation results are presented in Section 4. The conclusions are given in Section 5.

2. THE DICHOTOMOUS COORDINATE DESCENT

The original DCD algorithm updates a solution of a linear system of equations in directions of Euclidian coordinates in the cyclic order and with a step size α that takes one of M_b (number of bits) predefined values corresponding to a binary representation bounded by an interval $[-H, H]$ [7-8]. The algorithm starts the iterative search from the most significant bits of the solution and continues until the least significant bits were updated (the pseudo-code can be found in [7]). The algorithm complexity is limited by N_u, the maximum number of "successful" iterations. It can be seen that if H is a power of 2, the multiplications are replaced by bit shifts. The algorithm has only shift and accumulate operations (SAC) and no divisions [7]. It will be shown in the next section that the DCD method approximates very well the exact solution of a linear system if enough DCD iterations are executed. The maximum complexity of DCD part for our cases is $p(2N_u + M_b)$ SACs.

3. FMIPAPA-DCD ALGORITHM

The following vectors and matrices are defined in the following lines: $\mathbf{d}(n) = \mathbf{X}(n)\hat{\mathbf{h}}(n)$, where $\mathbf{d}(n) = [d(n), d(n-1),\ldots, d(n-p+1)]^T$ is the desired response vector of length p with p denoting the projection order; the step-size parameter is μ, the regularization constant is δ; $\mathbf{e}(n) = [e(n), e(n-1),\ldots, e(n-p+1)]^T$ is the error signal vector of length p, $\mathbf{X}(n) = [\mathbf{x}(n), \mathbf{x}(n-1),\ldots, \mathbf{x}(n-p+1)]$ is the input signal matrix; $\mathbf{x}(n) = [x(n), x(n-1),\ldots, x(n-L+1)]^T$ is a real-valued vector containing the L most recent samples of the input signal (i.e., far-end signal), $e(n) = d(n) - \hat{\mathbf{h}}^T(n-1)\mathbf{x}(n)$ is the error signal where d(n) denotes the desired response. Let us consider an echo canceller configuration, where an adaptive filter defined by $\hat{\mathbf{h}}(n) = [\hat{h}_0(n), \hat{h}_1(n),\ldots, \hat{h}_{L-1}(n)]^T$ is used to model the echo path. Superscript T denotes transposition, L is the length of the adaptive filter, and n is the time index. The filter output and the output error vectors are given by the next two equations, respectively

$$\mathbf{y}(n) = \mathbf{X}^T(n)\hat{\mathbf{h}}(n-1) \quad (1)$$

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n) \quad (2)$$

The general update of MIPAPA is [11]

$$\hat{\mathbf{h}}(n) = \hat{\mathbf{h}}(n-1) + \mu \mathbf{P}'(n)\left[\delta \mathbf{I}_p + \mathbf{X}^T(n)\mathbf{P}'(n)\right]^{-1}\mathbf{e}(n) \quad (3)$$

where the matrix

$$\mathbf{P}'(n) = \left[\mathbf{g}(n-1)\odot \mathbf{x}(n) \quad \mathbf{P}'_{-1}(n-1)\right] \quad (4)$$

with

$$\mathbf{P}'_{-1}(n-1) = \begin{bmatrix} \mathbf{g}(n-2)\odot \mathbf{x}(n-1) \ ... \\ \mathbf{g}(n-p)\odot \mathbf{x}(n-p+1) \end{bmatrix} \quad (5)$$

contains the first $p-1$ columns of $\mathbf{P}'(n-1)$ and the operator $\odot$ denotes the Hadamard product. The $\mathbf{g}$ vector coefficients are defined as in [9] and [10]

$$g_l(n-1) = \frac{1-\alpha}{2L} + (1+\alpha)\frac{\left|\hat{h}_l(n-1)\right|}{2\sum_{i=0}^{L-1}\left|\hat{h}_i(n-1)\right| + \varepsilon} \quad (6)$$

where $-1 \leq \alpha < 1$ and the small positive constant ε avoids division by zero (especially at the beginning of the adaptation when all the filter taps are initialized to zero).

A recursive filtering approach similar to that of [8] can exploit the time-shift property of $\mathbf{P}'(n)$. The following equation is obtained:

$$\begin{aligned}\mathbf{y}(n-1) &= \mathbf{X}^T(n-1)\hat{\mathbf{h}}(n-2) \\ &= \left[\mathbf{x}^T(n-1)\hat{\mathbf{h}}(n-2) \ ... \ \mathbf{x}^T(n-p)\hat{\mathbf{h}}(n-2)\right]^T \\ &= \left[y^0(n-1) \ y^1(n-1) \ ... \ y^{p-1}(n-1)\right]^T\end{aligned} \quad (7)$$

where $y^k(n-1) = \mathbf{x}^T(n-1-p)\hat{\mathbf{h}}(n-2)$. Also, we obtain:

$$\mathbf{y}(n) = \mathbf{X}^T(n)\hat{\mathbf{h}}(n-1) = \mathbf{z}(n) + \mathbf{F}(n)\hat{\boldsymbol{\varepsilon}}(n-1) \quad (8)$$

where $\mathbf{F}(n) = \mathbf{X}^T(n)\mathbf{P}'(n-1)$ and

$$\begin{aligned}\mathbf{z}(n) &= \mathbf{X}^T(n)\hat{\mathbf{h}}(n-2) \\ &= \left[\mathbf{x}^T(n)\hat{\mathbf{h}}(n-2) ... \mathbf{x}^T(n-p+1)\hat{\mathbf{h}}(n-2)\right]^T \\ &= \left[\mathbf{x}^T(n)\hat{\mathbf{h}}(n-2) \ y^0(n-1) \ ... y^{p-2}(n-1)\right]^T\end{aligned} \quad (9)$$

The proposed FMIPAPA-DCD algorithm equations are:

Initialization $\hat{\boldsymbol{\varepsilon}}(-1) = 0, \hat{\mathbf{h}}(-1) = 0, \ \mathbf{x}(-1) = \mathbf{0}, \mathbf{P}'(-1) = \mathbf{0}$

$$\mathbf{z}(n) = \begin{bmatrix} \mathbf{x}^T(n)\hat{\mathbf{h}}(n-2) \ ... \\ y^0(n-1) \ ... \ y^{p-2}(n-1) \end{bmatrix}^T \quad (10)$$

$$\mathbf{F}(n) = \mathbf{X}^T(n)\mathbf{P}'(n-1) \quad (11)$$

$$\mathbf{y}(n) = \mathbf{z}(n) + \mathbf{F}(n)\hat{\boldsymbol{\varepsilon}}(n-1) \quad (12)$$

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{y}(n) \quad (13)$$

$$\mathbf{P}'(n) = \begin{bmatrix} \mathbf{g}(n-1)\odot \mathbf{x}(n) ... \\ \mathbf{P}'_{-1}(n-1) \end{bmatrix} \quad (14)$$

$$\mathbf{S}(n) = \delta \mathbf{I}_p + \mathbf{X}^T(n)\mathbf{P}'(n). \quad (15)$$

Solve $\mathbf{S}(n)\boldsymbol{\varepsilon}(n) = \mathbf{e}(n)$ with DCD. (16)

$$\hat{\mathbf{h}}(n) = \hat{\mathbf{h}}(n-1) + \mu \mathbf{P}'(n)\hat{\boldsymbol{\varepsilon}}(n). \quad (17)$$

It can be seen that only $\mathbf{x}^T(n)\hat{\mathbf{h}}(n-2)$ need to be computed for $\mathbf{z}(n)$ and requires L multiplications and $L-1$ additions. Also, only the first row and first column of $\mathbf{F}(n)$ needs to be computed at each iteration. The rest of the matrix elements are the first $p-1$ rows and $p-1$ columns of $\mathbf{F}(n-1)$. The update of $\mathbf{S}(n)$ requires the computation of its first row. The other columns of $\mathbf{S}(n)$ are given by the first $p-1$ columns of $\mathbf{F}(n)$. The FMIPAPA-DCD algorithm solves the linear system of (16) with DCD iterations. Usually we have $p << L$ and therefore, the DCD part complexity is smaller than that of the main filtering part of the algorithms. If the linear system is solved with a direct method in double precision, FMIPAPA provides identical results with the original MIPAPA [9] and almost identical results with AMIPAPA as shown in [10]. The fast filtering procedure cannot be applied to the simpler AMIPAPA algorithm due to its special updating matrix autocorrelation procedure. Therefore, the influence of the DCD method is under scrutiny in the Simulations section. Otherwise, the algorithm retains the fast convergence, and good tracking abilities of the MIPAPA/AMIPAPA.

4. SIMULATIONS

Simulations were performed in the context of network echo cancellation. The echo path has 512 coefficients (the sampling frequency is 8 kHz); it is the first impulse response from ITU-T G168 Recommendation [11] (padded with zeros). The same length is used for the adaptive filter. The input signal is either a white Gaussian noise or a speech signal. Figure 1a shows the echo path, while Figure 1b shows the variable background noise and Figure 1c shows the speech signal. The output of the echo path is corrupted by an independent white Gaussian noise with different values of the signal-to-noise ratio (SNR). The performance measure is the normalized misalignment (in dB), defined as $20\log_{10}(\|\mathbf{h} - \hat{\mathbf{h}}(n)\|_2/\|\mathbf{h}\|_2)$, where $\mathbf{h}$ denotes the true impulse response of the echo path; the results are averaged over 30 independent trials. The MIPAPA algorithm has been chosen as the reference and was compared with the proposed version. Both algorithms use the same values for their parameters, i.e., the step-size is $\mu = 0.2$, the regularization constant is $\delta = 40\sigma_x^2/2L$ (where σ_x^2 is the input signal variance), and $\alpha = 0$. The misalignment difference between the original algorithms and their DCD based counterparts were plotted separately.

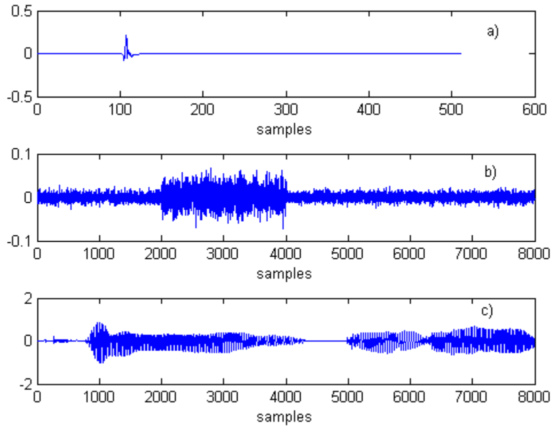

Figure 1. *a) The echo path; b) the variable background noise; c) the speech signal*

The closer to zero are the misalignment difference curves, the better approximation of the DCD versions is obtained of the original algorithms behavior (M_b was 16 in our simulations).

First, the performance of the algorithm is evaluated for different values of the projection order and different values of SNR. In next two figures the input signal is a white Gaussian noise with SNR=10 dB, p = 2, and an abrupt change of the echo path was introduced at time 0.5, by shifting the impulse response to the right by 12 samples.

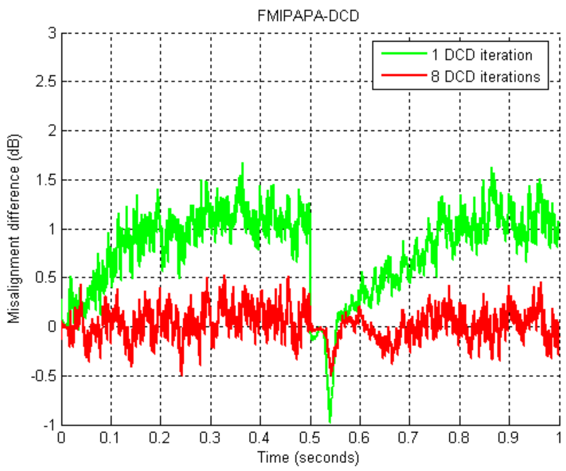

Figure 2. *Misalignment difference between MIPAPA and FMIPAPA-DCD with different number of DCD iterations (1 and 8 respectively).*

Figure 2 shows the misalignment difference between the misalignment curves of the original algorithms and their DCD based counterparts for two particular values of N_u ($N_u = 1$ and $N_u = 8$). It can be seen that this alignment difference decreases with an increased value of N_u and 8 iterations are sufficient if less than 0.5 dB from the ideal solution performance is sought. The performance improvement obtained with more DCD iterations can be explained by examining the error norm between the exact solution and the solution obtained with different number of DCD iterations. As expected, and proved in other applications of the DCD method ([7-8]), the error norm between the exact solution and the solution obtained with DCD iterations decreases with increased number of iterations. It can be seen from Figure 3 that the error norm with one DCD iteration is much higher than that obtained with 8 DCD iterations.

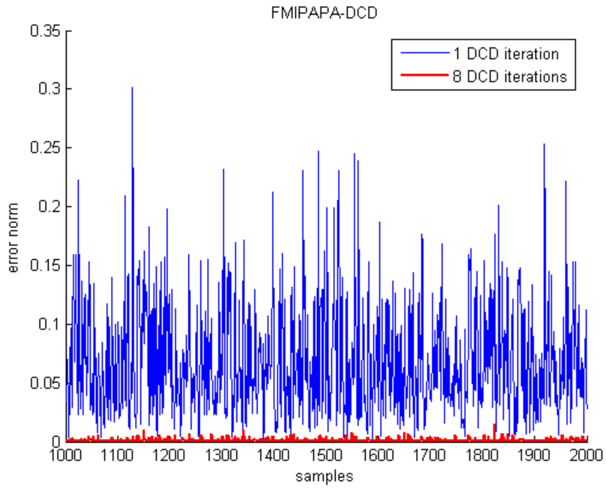

Figure 3: *The Error Norm for different number of DCD iterations for FMIPAPA-DCD. The input signal is a white Gaussian noise.*

The same conclusions can be obtained in other situations considered in [9-10] such us variable background noise SNR. In Figure 4, a variation of the background noise is considered (see Figure 1b); in this scenario, the SNR decreases from 20 dB to 10 dB between times 0.25 and 0.5. The input signal is a white Gaussian noise and the projection order is p = 2. The number of DCD iterations is increased to 16.

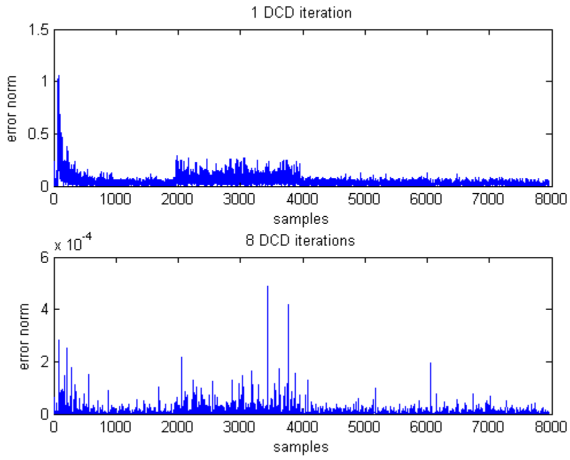

Figure 4. *The Error Norm for different number of DCD iterations of FMIPAPA-DCD in case of variable background noise (SNR decreases from 20 dB to 10 dB between samples 2000 and 4000).*

It can be seen from Figure 4 that, apart from the initial converging part from the first 1000 samples, the error norm increases in the region with more background noise (between samples 2000 and 4000). Also, 16 DCD iterations lead to a very small error norm.

Finally, a speech sequence (see Figure 1c) is used as input in Figure 4, SNR = 20 dB, and echo path changes), p = 8. It can be noticed that more DCD iterations are needed in case of using speech as input signal. One DCD iteration is clearly not a choice for the investigated situation. If less than 1.5 dB misalignment difference is allowed, 8 DCD iterations are enough for the DCD based algorithm. In case of using 16 DCD iterations in FMIPAPA-DCD the misalignment difference is smaller than 1 dB.

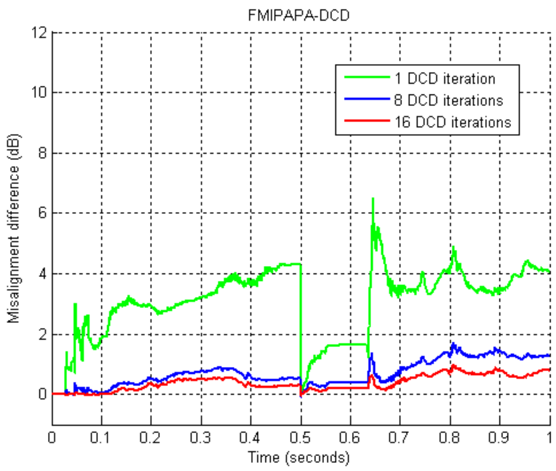

Figure 4. *Misalignment difference between MIPAPA and FMIPAPA-DCD with different number of DCD iterations.*

5. CONCLUSIONS

A proportionate affine version using fast recursive filtering and dichotomous coordinate descent iterations is proposed. Simulations of FMIPAPA-DCD in the context of network echo cancellation proved that the use of at least 8 DCD iterations only slightly alter the properties of the original algorithm.

REFERENCES

[1] D. L. Duttweiler, “Proportionate normalized least-mean-squares adaptation in echo cancellers,” *IEEE Trans. Speech Audio Process.*, vol. 8, no. 5, pp. 508–518, Sept. 2000.

[2] H. Deng and M. Doroslovački, “Proportionate adaptive algorithms for network echo cancellation,” *IEEE Trans. Signal Process.*, vol. 54, no. 5, pp. 1794–1803, May 2006.

[3] J. Benesty and S. L. Gay, “An improved PNLMS algorithm,” in *Proc. IEEE ICASSP*, 2002, pp. II-1881–II-1884.

[4] K. Ozeki and T. Umeda, “An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties,” *Electron. Commun. Jpn.*, vol. 67-A, no. 5, pp. 19–27, May 1984.

[5] S.L. Gay “A fast converging, low complexity adaptive filtering algorithm”, *Third International Workshop on Acoustic Echo Control*, Plestin les Greves, France, 1993, pp. 223-226.

[6] F. Albu, H.K. Kwan, “Fast block exact Gauss-Seidel pseudo affine projection algorithm”, *Electronics Letters*, Oct. 2004, pp. 1451-1453, Vol. 40, Issue:22

[7] Y. Zakharov and F. Albu, “Coordinate descent iterations in fast affine projection algorithm,” *IEEE Signal Processing Letters*, vol. 12, pp. 353–356, May 2005

[8] Y. Zakharov, “Low complexity implementation of the affine projection algorithm”, *IEEE Signal Processing Letters*, vol. 15, pp. 557-560, 2008

[9] C. Paleologu, S. Ciochina, J. Benesty, “An Efficient Proportionate Affine Projection Algorithm for Echo Cancellation”, *IEEE Signal Processing Letters*, vol. 17, issue 2, pp. 165-168

[10] F. Albu, C. Paleologu, J. Benesty, and S. Ciochina, “A low complexity proportionate affine projection algorithm for echo cancellation,” in *Proc. EUSIPCO*, Aalborg, Denmark, August 2010, pp. 6-10

[11] Digital Network Echo Cancellers, *ITU-T Rec. G.168*, 2002.