Design of Multichannel AP-DCD Algorithm using Matlab

Sasmita Deo *

Abstract— **This paper presented design of a low complexity multichannel affine projection (AP) algorithm using Dichotomous Coordinate descent(DCD) iterations. The computational complexity of this proposed algorithm is analyzed, and it is shown through simulation that the new multichannel algorithm not only provides the same performance as affine projection algorithm, it could also be implemented in hardware. It is expected that the new algorithm will consume less memory and power because it requires less number of multiplications and additions. The performance of the multichannel AP-DCD algorithm depends on the number of updates. When the number of update (N_u) increases, the performance of the multichannel AP-DCD algorithm provides the same performance as AP algorithm. Furthermore, to offering a good convergence speed, the new algorithm provides the expected tradeoff between convergence performance and computational complexity.**

Keywords: Adaptive filter, affine projection, AP-DCD algorithm, multichannel.

1 Introduction

Generally, the multichannel adaptive filtering problem's solution depends on the correlation between the number of channels, order and nature of the impulse response involved in the system. In last few decades multichannel adaptive affine projection algorithms are used in many different applications such as active noise control, acoustic echo cancelation and sound reproduction system. The family of AP algorithms presents good stability, fast convergence speed and modest computational cost. Nevertheless, the computational complexity of AP algorithm increases with the projection order and efficient strategies require to achieve fast and robust algorithms for real time systems. Therefore, different fast affine projection (FAP) algorithms have been proposed [1] basing on efficient matrix inversions. However, these FAP algorithms were numerically unstable, computationally complex, and also it did not provide the same convergence speed. Hence, FAP algorithm was further modified using Gauss-Seidel inversion [2] scheme for better numerical stability and low complexity. Although, this GSFAP [3] algorithm was good to implement for multichannel active noise control (ANC) system, but the complexity was not so small.

To reduce the complexity further, a Pseudo Affine Projection (PAP) algorithm [4] using Levinson-Durbin recursion [2] was proposed. The proposed algorithm was moderately complex but the design was more complex than GSFAP. Hence, the Levinson-Durbin recursion was replaced with the Gauss-Seidel method [3, 5] to derive a simpler algorithm [6, 7], called the Gauss-Seidel Pseudo Affine Projection (GSPAP) algorithm. The GSPAP algorithm provided the similar convergence performance and the complexity of the GSPAP algorithm was typically lower than that of the FAP-RLS and GSFAP algorithms. However, the GSPAP algorithm is still used at least one inverse matrix computation. This algorithm could become very complex for large matrices and prone to numerical instability. Therefore, to improve the stability and to reduce the complexity, a new pseudo affine projection algorithm called the modified filter-x Dichotomous Coordinate Descent Pseudo Affine Projection (MFX-DCDPAP) algorithm was introduced [8]. However, it requires large number of multiplications.

Thus, this paper presented a low complexity multichannel AP-DCD algorithm. The new algorithm is an extension work of single channel AP-DCD algorithm [9]. This proposed algorithm could be used in different applications of signal processing such as active noise control, acoustic echo cancelation and multimedia systems etc. The new multichannel algorithm has the potential to provide low complexity and better numerical stability, in addition, it also requires less number of multiplications. The multichannel AP-DCD algorithm is implemented in Matlab and discussed in the following sections.

2 Implementation of Multichannel AP-DCD Algorithm

The block diagram of a multichannel affine projection algorithm using dichotomous coordinate descent iterations is illustrated in Figure 1. $X1(n)$ and $X2(n)$ are the excitation signal matrix of $L \times K$ vector, where L is the filter length and $\mathbf{K}$ is the projection order. $h1(n)$ and $h2(n)$ are the additive noise and $w1(n)$ and $w2(n)$ are the adaptive filter of $L \times 1$ vector. y_n is the output of the adaptive filter and $K \times 1$ is the vector of y_n and e_n is the filter error. The complete analysis and design of the multichannel AP-DCD algorithm are given in table 1.

In the AP-DCD algorithm γ, μ and $\hat{\varepsilon}$ are regularization factors, step-size parameters and the solution vector of the linear system respectively; $\mathbf{x}1_n$ and $\mathbf{x}2_n$ are the excitation signal, $\mathbf{d}_n$ is the desired signal ,$\hat{w1}_n$ and $\hat{w2}_n$ are the adaptive filter taps, at n instant time. $X1_n = [\mathbf{x1}_{n-K+1} \ \ldots \ \mathbf{x1}_{n-1} \ \mathbf{x1}_n]^T$,$\mathbf{X2}_n =$

*July 26th 2010 of the manuscript submission and revised paper submission August 16th 2010. The author is with the Department of Electronics, University of York, & Communication Group, York, UK, YO10 5DD Tel/Fax: 00441904432308 Email: sasmitadeo@yahoo.co.uk,sd540@ohm.york.ac.uk

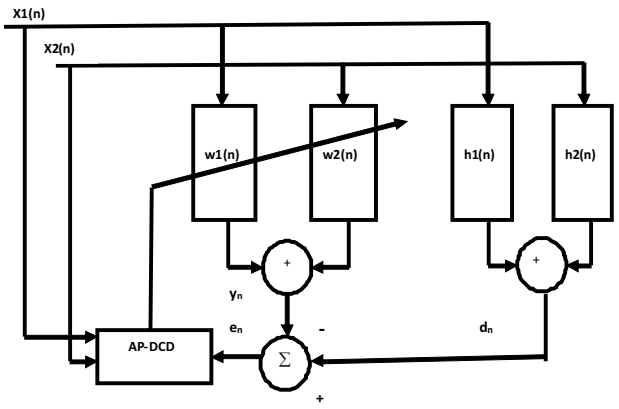

Figure 1: Block Diagram of Multichannel AP-DCD Adaptive Filtering Algorithm

Table 1: Multichannel AP-DCD Algorithm

Step	Equation	×	+
	Initialization: $\hat{\boldsymbol{\varepsilon}}_n = \mathbf{0}$,$\hat{w1}_n = \mathbf{0}$,$\hat{w2}_n = \mathbf{0}$ and $\mathbf{x1}_n = \mathbf{0}$,$\mathbf{x2}_n = \mathbf{0}$ and $\mathbf{y}_n = \mathbf{0}$ for $n < 0$		
	for $n = 0, 1, \ldots$		
1	$\mathbf{z}_n = [[\mathbf{y}_{n-1}]_{2:K} \ \mathbf{x}_n^T \hat{\mathbf{w}}_{n-2}]^T$	$2L$	$2L-1$
2	Calculate $\mathbf{G}_n = X_n X_{n-1}^T$	$2K$	$2K$
3	$\mathbf{y}_n = \mathbf{z}_n + \mathbf{G}_n \hat{\boldsymbol{\varepsilon}}_{n-1}$	$2K^2$	$2K^2$
4	$\mathbf{e}_n = \mathbf{d}_n - \mathbf{y}_n$	–	K
5	$\mathbf{r} = \mu \mathbf{e}_n$	K	–
6	Calculate $\mathbf{R}_n = \mathbf{X}_n \mathbf{X}_n^T + \gamma \mathbf{I}_K$	2	2
7	$\hat{\boldsymbol{\varepsilon}}_n = \mathbf{0}$, $\alpha = H/2$, $m = 1$,	–	–
	for $k = 1, \ldots, N_u$		
8	$p = \arg\max_{p=1,\ldots,K}\{\lvert r_p \rvert\}$		$K-1$
9	while $\lvert r_p \rvert \le (\alpha/2)[\mathbf{R}_n]_{p,p}$ and $m \le M_b$ do $m = m+1$,$\alpha = \alpha/2$	–	1
10	if $m > M_b$, the algorithm stops	–	–
11	$[\hat{\boldsymbol{\varepsilon}}_n]_p = [\hat{\boldsymbol{\varepsilon}}_n]_p + \text{sign}(r_p)\alpha$	–	1
12	$\mathbf{r} = \mathbf{r} - \text{sign}(r_p)\alpha \mathbf{R}_n^{(p)}$	–	K
13	$\hat{\mathbf{w}}_n = \hat{\mathbf{w}}_n + \text{sign}(r_p)\alpha \mathbf{X}_{n-p}^T$	–	$2L$
	Total: $2L + 2K^2 + 3K + 2$ mults and $2L(N_u+1) + 2K^2 + K(2N_u+3) + M_b + 1$ adds		

$[\mathbf{x2}_{n-K+1} \ \ldots \ \mathbf{x2}_{n-1} \ \mathbf{x2}_n]^T$ where $\mathbf{x1}_n = [x1_{n-L+1} \ \ldots \ x1_{n-1} \ x1_n]^T$; $\mathbf{x2}_n = [x2_{n-L+1} \ \ldots \ x2_{n-1} \ x2_n]^T$; $\mathbf{h1}_n = [\mathbf{h1}_{n-K+1} \ \ldots \mathbf{h1}_{n-1} \mathbf{h1}_n]^T$; $\mathbf{h2}_n = [\mathbf{h2}_{n-K+1} \ \ldots \mathbf{h2}_{n-1} \mathbf{h2}_n]^T$; $\mathbf{I}_K$ is an $K \times K$ identity matrix and $(\cdot)^T$ denotes the transpose matrix.

In step1, for calculation of $\mathbf{z}_n$ requires $\mathbf{x}_n^T \hat{\mathbf{w}}_{n-2}$, because the filter output $\mathbf{y}_n = \mathbf{X}_n \hat{\mathbf{w}}_{n-1}$ is calculated using a recursive approach as shown in steps 1 to 3, $[\mathbf{y}_{n-1}]_{2:K}$ values known from the previous $(n-1)$th sample. $\mathbf{X}_n$ is the matrix of $[\mathbf{x1}_n; \mathbf{x2}_n]$ and $\hat{\mathbf{w}}_n$ is the matrix of $[w1(n) \ ; \ w2(n)]$. Therefore, it requires $2L$ multiplications and $2(L-1)$ additions.

In step2, the first row of $\mathbf{G}_n$ matrix is the vector $[\mathbf{x1}_n^T \mathbf{x1}_{n-1} \ \mathbf{x1}_n^T \mathbf{x1}_{n-2} \ \ldots \ \mathbf{x1}_n^T \mathbf{x1}_{n-K}]$ and $[\mathbf{x2}_n^T \mathbf{x2}_{n-1} \ \mathbf{x2}_n^T \mathbf{x2}_{n-2} \ \ldots \ \mathbf{x2}_n^T \mathbf{x2}_{n-K}]$, calculation of these vector requires $2K$ multiplications and $2K$ additions. step 3, the vector $\mathbf{y}_n$ is calculated using the vector $\mathbf{z}_n$, matrix $\mathbf{G}_n$ and vector $\hat{\boldsymbol{\varepsilon}}_{n-1}$. It requires $2K^2$ multiplications and $2K^2$ additions [9]. Step 4, the error vector $\mathbf{e}_n$ is calculated using the vector d_n, and the vector y_n; it needs K additions. Instead of normalizing $\hat{\boldsymbol{\varepsilon}}$ by the step-size μ in the original AP algorithm [10], the AP-DCD algorithm does this normalization with $\mathbf{e}_n$ (step 5) and obtains a vector $\mathbf{r}$. This algorithm allows to avoid multiplications when updating the filter weights (step 13) for any value of μ and d_n is the addition of two additive noise signals $h1(n)$ and $h2(n)$.

Step 6, The correlation matrix $\mathbf{R}_n$ of the input matrix $\mathbf{X}_n$. As having calculated the matrix $\mathbf{G}_n$, the only element of the matrix $\mathbf{R}_n$ that requires calculation $[\mathbf{R}_n]_{K,K} = X_n^T X_n + \gamma \mathbf{I}_k$. The other elements of $\mathbf{R}_n$ can be taken from matrices $\mathbf{R}_{n-1}$ and $\mathbf{G}_n$. Thus, the update of the matrix $\mathbf{R}_n$ requires only 2 multiplications and 2 additions

Steps 7 to 13, solve the liner system $\mathbf{R}_n \boldsymbol{\varepsilon}_n = \mathbf{r}$ using the DCD [11] algorithm and obtain an approximation solution $\hat{\boldsymbol{\varepsilon}}_n$. The DCD algorithm uses coordinate descent iterations with variable step-size parameter α for solving the normal equations without multiplications and divisions [11]. In the DCD algorithm, the step-size α can take one of M_b with predefined values corresponding to representation of elements $[\hat{\boldsymbol{\varepsilon}}_n]_p$, $p = 1, \ldots, K$, of the vector $\hat{\boldsymbol{\varepsilon}}_n$ as fixed-point words with M_b bits within an amplitude range $[-H, H]$. The parameter H should preferably be larger than the absolute maximum $H_{\max}$ among elements of the true solution of the system. In the case of uncertainty, H should be chosen high enough for a worst-case situation. Note that the choice $H > H_{\max}$ does not affect the adaptive filter performance and only results in an increase of M_b by $\log_2(H/H_{\max})$ bits that has a little impact on the complexity. If $H < H_{\max}$, it may slow down the convergence of the adaptive filter at the initial part of the learning process. It is convenient to choose H as a power-of-two number, resulting in the step-size parameter α also a power of two. Then all multiplications by α can be implemented by bit-shifts, which significantly simplifies the algorithm implementation. Elements of the residual vector $\mathbf{r}$ are denoted as r_p, $p = 1, \ldots, K$. The DCD algorithm starts the iterative search from the most significant bits of elements in $\hat{\boldsymbol{\varepsilon}}_n$. As the most significant bits have been updated, the algorithm starts updating the next bit, and so on. One update requires K bit-shifts, K additions, and K comparisons; the latter are counted as additions. With the N_u updates, the complexity of the DCD algorithm is upper limited by $2N_uK + M_b$ additions [12]. The filter weights update at step 13 is included in the DCD iterations; it is implemented with addition and bit-shift operations only [9].

3 Computational Complexity of Multichannel AP-DCD Algorithm

The maximum number of multiplications per iteration for the AP-DCD algorithm is:

$$M_{AP-DCD} = 2L + 2K^2 + 3K + 2 \qquad (1)$$

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Table 2: Comparison of the Number of Multiplications ($\times$) and Additions ($+$) per Iteration of the Multichannel AP-DCD Algorithm

Multichannel AP-DCD	($\times$) per iteration	($+$) per iteration
$L = 16$,$K = 8$, $N_u = 16$	186	969
$L = 64$,$K = 8$, $N_u = 16$	282	2601
$L = 150$, $K = 5$,$N_u = 4$	367	1622
$L = 512$,$K = 8$, $N_u = 16$	1178	17833

The maximum number of additions per iteration for the mutichannel AP-DCD algorithm is:

$$A_{AP-DCD} = 2L(N_u+1)+2K^2+K(2N_u+3)+M_b+1 \quad (2)$$

In the multichannel AP-DCD algorithm design required 367 multiplications and 1622 additions at filter length $L = 150$, projection order $K = 5$ and number of updates $N_u = 4$. Whereas the reported paper of Albu *et. al* [8] MFX-DCDPAP algorithm required 3198 multiplications and 3524 additions at filter length $L = 150$, projection order $K = 5$ and number of updates$N_u = 4$. Due to more number of multiplication in MFX-DCDPAP algorithm, it consumes more power in hardware design. However, in the multichannel AP-DCD design, it will consume less power and memory space in FPGA implementation, because it has less number of multiplications and additions.

The number of multiplications and additions that are required for AP-DCD algorithm is illustrated in table 2. It is clearly evident from the results that while MFX-DCDPAP algorithm required 3198 multiplications and 3524 additions at filter length $L = 150$, AP-DCD algorithm only required 367 multiplications and 1622 additions to provide same results. Therefore, the memory requirements of the multichannel affine projection algorithm is significantly less than MFX-DCDPAP algorithm, because in MFX-DCDPAP needed several matrices and vectors in comparision to multichannel AP-DCD algorithm.

4 Numerical Results of Multichannel AP-DCD Algorithm

The numeric results of multichannel AP-DCD algorithm were obtained by computer simulation using Matlab and then compared the misalignment with the original AP algorithm. The desired data are generated according to

$$h_{n1} = h1_n x_{1n} + v_n \quad (3)$$

$$h_{n2} = h2_n x_{2n} + v_{n1} \quad (4)$$

where v_n and v_{n1} are independent zero-mean Gaussian random numbers. Therefore the desired signal is the addition of the equation 3 and 4

$$d_n = h_{n1} + h_{n2} \quad (5)$$

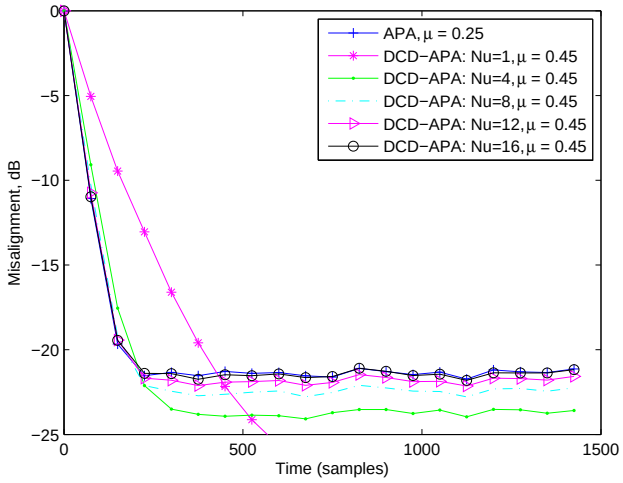

Figure 2: $L = 16$, $SNR = 30dB$, $M_b = 16$, $H = 4$, $\mu(AP) = 0.25$, $\mu(DCD) = 0.45$, $K = 8$, $N_{mc} = 200$

The data vectors x_{1n} and x_{2n} contain autoregressive correlated random numbers are given by

$$x_{1n} = \nu x1_{n-1} + w_n \quad (6)$$

$$x_{2n} = \nu x2_{n-1} + w_n \quad (7)$$

Where ν is the autoregressive factor ($\nu = 0.9$) and w_n is independent zero-mean random Gaussian numbers of unit variance. Elements of impulse responses are $h1_i$ and $h2_i$, where $i = 0 \ldots L-1$, is independent zero-mean Gaussian random numbers with variance exp $(-\beta_i)$ in each simulation trial, new vector h1 and h2 are generated. The misalignment in a multichannel simulation trial is calculated as

$$||h1 - \hat{w}_n 1||^2 + ||h2 - \hat{w}_n 2||^2 / ||h1 + h2||^2 \quad (8)$$

The values obtained in N_{mc} trials are averaged and divided by a number of channels and plotted against the time index n. The signal to noise ratio (SNR) is set 30dB for all simulation scenarios.

Figure 2 plotted against misalignment and time (iterations) of AP-DCD algorithm for $K = 8$, DCD step size $(\mu) = 0.45$ and $L = 16$ with different number of update (N_u) values . In case of one update ($N_u = 1$) against AP algorithm, the AP-DCD algorithm has a lower convergence speed than the AP algorithm and the steady state is out of range. When the number of updates increase at $N_u = 4$ and 8 the convergence speed stays nearly same with AP algorithm and the steady state value approaches towards AP algorithm. But at $N_u = 12$, the convergence speed is same as AP and the steady state values further approaches towards AP. However, for $N_u = 16$, AP-DCD algorithm provides the same performance as AP. This is due to the solution of the equation $\mathbf{R}_n \boldsymbol{\varepsilon}_n = \mathbf{r}$ provided by the DCD algorithm convergences to the true solution.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

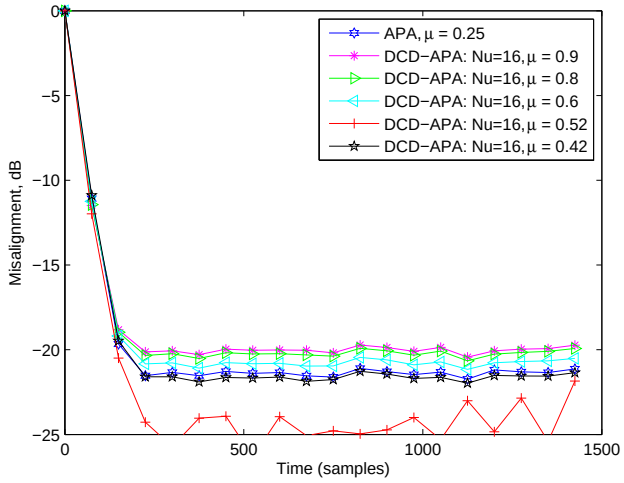

Figure 3: $L = 64$, $SNR = 30dB$, $M_b = 16$, $H = 4$, $\mu(APA) = 0.25$, $N_u = 16$, $K = 8$ and $N_{mc} = 200$

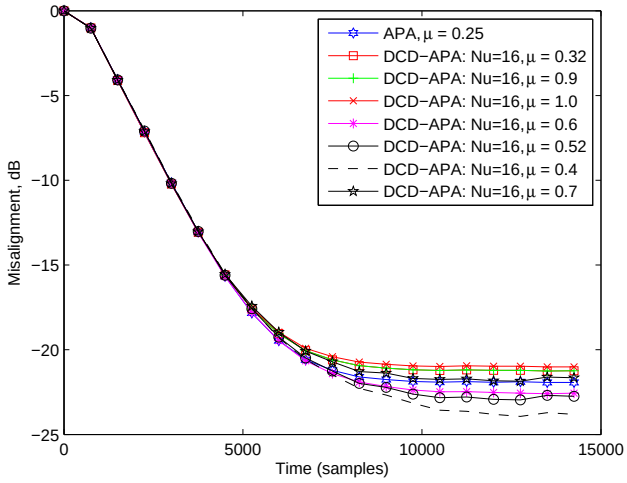

Figure 4: $L = 512$, $SNR = 30dB$, $M_b = 16$, $H = 4$, $\mu(APA) = 0.25$, $N_u = 16$, $K = 8$

The figure 2 suggests, at $N_u = 16$, $K = 8$ the AP-DCD algorithm provides the same performance as AP. Hence in figure 3, at $N_u = 16$, $K = 8$ and $L = 64$ by adjusting the step size (μ) of DCD against AP algorithm, the performance of mislignments with time (iterations) are analyzed. At $\mu = 0.52$ the AP-DCD algorithm provides slightly faster convergence speed and slower steady-state value than AP algorithm. With the increase in μ, the AP-DCD algorithm provides same convergence speed as AP but the steady-state approaches towards AP algorithm. When the step-size (μ) $= 0.42$ the performance of convergence speed and steady-state values of the AP-DCD algorithm are same as AP algorithm.

Figure 4 shows that, for $L = 512$, $K = 8$, $N_u = 16$, μ value of DCD is adjusted with AP. After couple of μ value adjustment it is seen that at $\mu = 0.7$, the learning curve is completely merzed with AP algorithm and has a perfect solver of the system equations. That means the convergence speed and steady state value of AP-DCD algorithm provides the same performance as AP.

5 Conclusions

The proposed multichannel low complexity affine projection algorithm using Dichotomous Coordinate descent (DCD) iterations implemented in Matlab. This proposed algorithm has better numerical stability than the other multichannel affine projection algorithm (MFX-DCDPAP). The performance of the multichannel AP-DCD algorithm depends on the number of updates. When the number of updates (N_u) increase, the performance of the multichannel AP-DCD algorithm provides the same performance as AP algorithm. Furthermore, the computational complexity of the proposed algorithm is also analyzed, and it is shown through simulations that the new algorithm not only provides the same performance as AP algorithm, but also it could be implemented in hardware (FPGA, ASICs). It is also expected that AP-DCD algorithm will consume very less memory and power. Because it requires less number of additions and multiplications. In addition, it is also shown that the new algorithm provides the expected tradeoff between convergence performance and computational complexity. Further investigation is required for the proposed algorithm to implement in hardware (ASICs, FPGA).

Acknowledgment

The author would like to thank Dr. Yuriy V. Zakharov for his support and suggestions during this work, and also the author would like to thank Department of Electronics, York University for providing the assistantship.

References

[1] Ding, H.,"Fast affine projection adaptation algorithms with stable and robust symmetric linear system solvers," *IEEE Trans. Signal Processing*, V55, N5, pp. 1730-1740, 2007.

[2] Bouteille, F., Scalart, P.,Corazza, M., "Pseudo Affine Projection Algorithm New Solution for Adaptive Identication," *Sixth European Conference on Speech Communication and Technology* , 1999.

[3] Bouchard, M.,"Multichannel affine and fast affine projection algorithms for active noise control and acoustic equalization systems,"*IEEE Transactions on Speech and Audio Processing*, V11, N1, pp. 54-60, 2003.

[4] Albu, F., Paleologu, C.,"The Variable Step-Size Gauss-Seidel Pseudo Affine Projection Algorithm," *International Journal of Computer Science*, V5, N1, 2010.

[5] Albu, F., Kadlec, J., Coleman, N., Fagan, A., "The Gauss-Seidel fast affine projection algorithm," *IEEE*

Workshop on Signal Processing Systems, pp. 109-114, 2002.

[6] Albu, F., Fagan, A., " The Gauss-Seidel pseudo affine projection algorithm and its application for echo cancellation," *Asilomar conference on signals system and computers*, V2, pp. 1303-1308, 2003.

[7] Albu, F., Bouchard, M.," A low-cost and fast convergence Gauss-Seidel pseudo affine projection algorithm for multichannel active noise control," *Proceedings ICASSP*, V5, pp. 121-124, 2004.

[8] Albu, F., Zakharov, Y., Paleologu, C., "Modified filtered-x dichotomous coordinate descent recursive affine projection algorithm," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, V0, pp. 257-260, 2009.

[9] Zakharov,Yuri.,"Low Complexity implementation of the Affine Projection Algorithm,"*IEEE Signal Processing letters*, V15, pp. 557-560, 2008.

[10] Sayed, A. H.,"Fundamentals of adaptive filtering,"*Hoboken, N. J.: Wiley*, 2003.

[11] Zakharov, Y. V., Tozer, T. C.," Multiplication-free iterative algorithm for LS problem," *Electronics Letter*, V40, N9, pp. 567-569,2004.

[12] Zakharov, Y. V., White, G. P., Liu, J.,"Low-complexity RLS algorithms using dichotomous coordinate descent iterations," *IEEE Transactions on Signal Processing*, V56, N7, pp. 3150-3161, 2008.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)