HIRSCHMAN OPTIMAL TRANSFORM DFT BLOCK LMS ALGORITHM

Victor E. DeBrunner
victor.debrunner@eng.fsu.edu
Dept. of Elec. & Comp. Eng.
Florida State University

Osama Alkhouli
{osama_k}@ou.edu
School of Elec. & Comp. Eng.
University of Oklahoma

ABSTRACT

In this paper a new block LMS algorithm is introduced. This algorithm is based on a fast HOT convolution developed by our group [1]. We call our algorithm the block HOT-DFT LMS algorithm. Our algorithm uses the premise that the filter size is much smaller than the block size. Our developed algorithm is very similar to the block DFT LMS algorithm, but provides a reduced computational complexity of about 30%. The computational efficiency of the block HOT-DFT LMS algorithm is verified and its convergence analysis is analyzed.

Index Terms— Adaptive filtering, Hirschman Uncertainty, Transform domain LMS algorithm

1. INTRODUCTION

The computational saving of any fast block LMS algorithm depends on how efficiently each of the two convolutions involved in the LMS algorithm are calculated [2], [3]. When the DFT convolution is used, the block DFT LMS algorithms result. They are most efficient when the block and filter sizes are equal. Recently, we developed a fast convolution based on the Hirschman Optimal Transform (HOT) [1]. The HOT convolution is more efficient than the DFT convolution when the disparity in the lengths of the sequences being convolved is large. The block HOT-DFT LMS algorithm used the fast HOT convolution to calculate the filter output and update the weights.

The HOT, similar to the DFT, is a newly developed transform [4], [5]. For example, the 3^2-point HOT matrix is

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & \omega_3 & 0 & 0 & \omega_3^2 & 0 & 0 \\ 0 & 1 & 0 & 0 & \omega_3 & 0 & 0 & \omega_3^2 & 0 \\ 0 & 0 & 1 & 0 & 0 & \omega_3 & 0 & 0 & \omega_3^2 \\ 1 & 0 & 0 & \omega_3^2 & 0 & 0 & \omega_3^4 & 0 & 0 \\ 0 & 1 & 0 & 0 & \omega_3^2 & 0 & 0 & \omega_3^4 & 0 \\ 0 & 0 & 1 & 0 & 0 & \omega_3^2 & 0 & 0 & \omega_3^4 \end{bmatrix}, \quad (1)$$

where $\omega_3=e^{-j2\pi/3}$. In general, the NK-point HOT basis are generated from the N-point DFT basis as follows. Each of the DFT basis functions are interpolated by K and then circularly shifted to produce the complete set of orthogonal basis signals that define the HOT. The fact that the HOT basis has many zero-valued samples, and their resemblance to the DFT basis sequences, makes the HOT computationally attractive. Recently the HOT transform was used to develop the HOT LMS algorithm [6], [7], which is a transform domain LMS algorithm, and the HOT block LMS algorithm [8], which a fast block LMS algorithm. The block HOT-DFT LMS algorithm presented here is different from the HOT block LMS algorithm presented in [8]. Our block HOT-DFT LMS algorithm developed in this paper uses the fast HOT convolution [1] which is summarized in the next paragraph.

The main idea behind the HOT convolution is to partition the longer sequence into sections of the same length as the shorter sequence and then convolve each section with the shorter sequence efficiently using fast DFT convolution. The relevance of the HOT will become apparent when the all of the (sub)convolutions are put together concisely in a matrix form.

The following notations are used in this paper. Nonbold lowercase letters are used for scalar quantities, bold lowercase for vectors and bold uppercase for matrices. Nonbold uppercase letters are used for integer quantities such as length or dimensions. The lowercase letter k is reserved for the block index. The lowercase letter n is reserved for the time index. The time and block indexes are put in brackets, whereas subscripts are used to refer to elements of vectors and matrices. The subscripts F and H are used to highlight the DFT-domain and HOT-domain quantities, respectively. In Section 2, the HOT-DFT block LMS algorithm is developed. Its computational cost is analyzed in section 3. Section 4 contains the convergence analysis. Simulations are provided in Section 5. Finally, we conclude.

2. DEVELOPMENT OF THE BLOCK HOT-DFT LMS ALGORITHM

Recall that in the block LMS algorithm, two convolutions are required. The first convolution is the convolution between the filter impulse response and the filter input. This one simply computes the output of the filter in each block. The second convolution is a convolution between the filter input and the error and is needed to estimate the gradient in the filter weight update equation. If the filter size is much larger than the block size, then the fast HOT convolution can be used to calculate the first convolution. However, the second convolution is a convolution between two signals of the same length, a situation where the fast HOT convolution is not superior to regular convolution. We propose a modification to the HOT convolution to improve this

situation. The fast HOT convolution, as described in [1], is based on the overlap-and-add method [9]. Since the overlap-save method is more convenient in the block LMS algorithm, the fast HOT convolution is developed with the overlap-save method and then applied to the block LMS algorithm. Let N be the filter length and $L = NK$ be the block size, where N, L, and K are all integers. Let

$$\hat{\mathbf{w}}(k) = \begin{bmatrix} w_0(k) & w_1(k) & \cdots & w_{N-1}(k) \end{bmatrix}^T \quad (2)$$

be the filter tap-weight vector in the k[th] block and

$$\mathbf{u}(k) = \begin{bmatrix} u(kL-N+1) & u(kL-N+2) & \cdots & u(kL+L-1) \end{bmatrix}^T \quad (3)$$

be the vector of input samples needed in the k^{th} block. This vector is then is divided into $K+1$ N-overlapping sections. Such sections can be formed by multiplying $\boldsymbol{u}(k)$ with the following matrix

$$\mathbf{J} = \begin{bmatrix} I_N & 0 & 0 & \cdots & 0 & 0 \\ 0 & I_N & 0 & \ldots & 0 & 0 \\ 0 & I_N & 0 & \cdots & 0 & 0 \\ 0 & 0 & I_N & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & I_N & 0 \\ 0 & 0 & 0 & \cdots & I_N & 0 \\ 0 & 0 & 0 & 0 & 0 & I_N \end{bmatrix}. \quad (4)$$

The vector $\boldsymbol{z}(k) = \boldsymbol{J}u(k)$ will be then of length $2NK$. Let the matrix $\boldsymbol{P}$ be a $2NK$ x $2NK$ matrix such that

$$\begin{bmatrix} \mathbf{z}_o(k) \\ \mathbf{z}_1(k) \\ \vdots \\ \mathbf{z}_{K-1}(k) \end{bmatrix} = P\mathbf{u}(k), \quad (5)$$

where $\boldsymbol{z}_i(k)$ is the i^{th} polyphase component of $\boldsymbol{z}(k)$. Then take the $2NK$-point HOT of $\boldsymbol{P}u(k)$. Post append the tap-weight vector $\boldsymbol{w}(k)$ with $\boldsymbol{N}$ zeros. The resulting vector is denoted by $\boldsymbol{w}(k)$. The DFT of w(k) is given by

$$\mathbf{w}_F(k) = \mathbf{F}_{2N}\mathbf{w}(k). \quad (6)$$

Let $\boldsymbol{e}_K$ be a column vector of all ones. i.e.

$$\mathbf{e}_K = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \end{bmatrix}^T, \quad (7)$$

then the $2NKx2N$ matrix $\boldsymbol{E}$ is given by

$$\mathbf{E} = \begin{bmatrix} \mathbf{e}_K & 0 & \cdots & 0 \\ 0 & \mathbf{e}_K & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{e}_K \end{bmatrix}. \quad (8)$$

Pointwise multiply the vectors $\boldsymbol{E}\boldsymbol{w}_F(k)$ and $\boldsymbol{HPu}(k)$ and then take the inverse HOT of the resulting vector. Define the matrices

$$\mathbf{A} = \begin{bmatrix} 0 & I_N \end{bmatrix} \quad (9)$$

and

$$\mathbf{G} = \begin{bmatrix} \mathbf{A} & 0 & \cdots & 0 \\ 0 & \mathbf{A} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{A} \end{bmatrix}. \quad (10)$$

According to the overlap-and-save method, the output of the adaptive filter in the k^{th} block

$$\mathbf{y}(k) = \begin{bmatrix} y(kL) & y(kL+1) & \cdots & y(kL+L-1) \end{bmatrix} \quad (11)$$

is given by

$$\mathbf{y}(\mathrm{k}) = \mathbf{GPH}^{-1}\left(\mathbf{Ew}_{\mathrm{F}}(\mathrm{k})\right) \otimes \left(\mathbf{HPu}(\mathrm{k})\right), \quad (12)$$

The symbol $\otimes$ indicates pointwise matrix multiplication, and throughout this discussion, pointwise matrix multiplication takes a higher precedence than does conventional matrix multiplication. The desired signal vector and the filter error in the k^{th} block are given by

$$\mathbf{d}(k) = \begin{bmatrix} d(kL) & d(kL+1) & \cdots & d(kL+L-1) \end{bmatrix}^T \quad (13)$$

and

$$\mathbf{e}(k) = \begin{bmatrix} e(kL) & e(kL+1) & \cdots & e(kL+L-1) \end{bmatrix}^T, \quad (14)$$

respectively, where

$$\mathbf{e}(k) = \mathbf{d}(k) - \mathbf{y}(k). \quad (15)$$

The filter update equation is given by

$$\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \frac{\mu}{L}\sum_{i=0}^{L-1} \begin{bmatrix} u(kL+i) \\ u(kL+i) \\ \vdots \\ u(kL+i-N+1) \end{bmatrix} e(kL+i). \quad (16)$$

The sum in equation (16) can be efficiently calculated using an L-point DFT of the error vector $\boldsymbol{e}(k)$ and input vector $\boldsymbol{u}(k)$. However, the L-point DFT of $\boldsymbol{u}(k)$ is not available and only the $2N$-point DFTs of the K sections of $\boldsymbol{u}(k)$ are available. Therefore, the sum in equation (16) should be divided into K sections as follows.

$$\begin{aligned} &\sum_{i=0}^{L-1} \begin{bmatrix} u(kL+i) \\ u(kL+i-1) \\ \vdots \\ u(kL+i-N+1) \end{bmatrix} e(kL+i) \\ &= \sum_{l=0}^{K-1}\sum_{i=0}^{N-1} \begin{bmatrix} u(kL+lK+i) \\ u(kL+lK+i-1) \\ \vdots \\ u(kL+lK+i-N+1) \end{bmatrix} e(kL+lK+i). \end{aligned} \quad (17)$$

For each l the sum over i can be calculated as follows. Form the vectors

$$\mathbf{u}_l(k) = \begin{bmatrix} u(kL+lK-N) & \cdots & u(kL+lK+N+1) \end{bmatrix}^T \quad (18)$$

and

$$\mathbf{e}_l(k) = \begin{bmatrix} 0_N & u(kL+lK) & \cdots & e(kL+lK+N+1) \end{bmatrix}^T. \quad (19)$$

The sum over *i* is just the first *N* elements of the circular convolution of $\boldsymbol{e}_l(k)$ and the circularly shifted $\boldsymbol{u}_l(k)$. Therefore, the filter update equation according to the HOT-DFT LMS algorithm can be written as

$$\mathbf{w}_F(k+1) = \mathbf{w}_F(k) + \frac{\mu}{L}\sum_{l=0}^{K-1} \mathbf{U}\left(\mathbf{F}_{2N}\mathbf{u}_1\ (\mathrm{k})\right)^* \otimes \left(\mathbf{F}_{2N}\mathrm{e}_1(\mathrm{k})\right), \quad (20)$$

where

$$\mathbf{U} = \mathbf{F}_{2N}\begin{bmatrix} I_{2N} & 0_{2N} \\ 0_{2N} & 0_{2N} \end{bmatrix}\mathbf{F}_{2N}^{-1}. \quad (21)$$

3. COMPUTATIONAL COST OF THE HOT-DFT LMS

Before looking at the convergence analysis of the new adaptive filter, we examine its computational cost. To calculate the output of the k^{th} block, *2K+1 2N*-point DFTs are needed. Therefore, $(2K+1)2N\log_2(2N)+2NK$ multiplications are needed to calculate the output. To calculate the gradient estimate in the filter update equation, *3K 2N*-point DFTs are required. Therefore, $6KN\log_2 2N+2NK$ multiplications are needed. The total multiplication count of our new algorithm is then $(4K+1)2N\log_2+4NK$. The multiplication count for the DFT block LMS algorithm is $8KN\log_2 2NK+4NK$. Therefore, as *N* gets smaller and *K* gets higher, the HOT-DFT block LMS algorithm becomes more efficient than the DFT block LMS algorithm. For example, for *N = 100* and *K = 10* the HOT-DFT is about *30%* more efficient and for *N = 50* and *K = 20* the HOT-DFT is about *40%* more efficient.

4. CONVERGENCE ANALYSIS AF THE HOT-DFT LMS

Now the convergence of the new algorithm is analyzed. Let the desired signal be generated using the linear regression model

$$d(n) = w^o(n) * u(n) + e^o(n), \quad (22)$$

where $w^o(n)$ is the impulse response of the Wiener optimal filter and $\mathbf{e^0(n)}$ is the irreducible estimation error, which is white noise and statistically independent of the adaptive filter input. In the k^{th} block the above equation can be written in the DFT domain as

$$\mathbf{d}(\mathrm{k}) = \mathbf{GPH}^{-1}\left(\mathbf{E}\mathbf{w}_F^o\ (\mathrm{k})\right) \otimes \left(\mathbf{HPu}(\mathrm{k})\right) + e^o(k). \quad (23)$$

The l^{th} section of the error is given by

$$\mathbf{F}\mathbf{e}_1(\mathrm{k}) = \mathbf{L}\left(\mathbf{w}_F^o\ (\mathrm{k}) - \mathbf{w}_F\ (\mathrm{k})\right) \otimes \left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right) + \mathbf{F}\mathbf{e}_l^o(k), \quad (24)$$

where

$$\mathbf{L} = \mathbf{F}_{2N}\begin{bmatrix} 0_{2N} & 0_{2N} \\ 0_{2N} & I_{2N} \end{bmatrix}\mathbf{F}_{2N}^{-1}. \quad (25)$$

Using equation (20), the error in the estimation of the adaptive filter weight vector $\boldsymbol{\varepsilon_F(k)=Fw^o - Fw(k)}$ is updated according to

$$\boldsymbol{\varepsilon}_F(k+1) = \boldsymbol{\varepsilon}_F(k) - \frac{\mu}{L}\sum_{l=0}^{K-1} \mathbf{U}\left(\mathbf{F}_{2N}\mathbf{u}_1\ (\mathrm{k})\right)^* \otimes \left(\mathbf{F}_{2N}\mathrm{e}_1(\mathrm{k})\right). \quad (26)$$

Using equation (24) we can write that

$$\left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)^* \otimes \left(\mathbf{F}\mathbf{e}_l(\mathrm{k})\right) = \mathrm{Diag}\{\mathbf{F}\mathbf{u}_l(\mathrm{k})\}^*\left(\mathbf{L}Diag\{\mathbf{F}\mathbf{u}_l(\mathrm{k})\}\boldsymbol{\varepsilon}_\mathrm{F}(k) + \mathbf{F}\mathbf{e}_l^o(k)\right). \quad (27)$$

Using the fact that

$$Diag\{\mathbf{v}\}\mathbf{R}Diag\{\mathbf{u}\} = (\mathbf{v}\mathbf{u}^T)\otimes\mathbf{R}, \quad (28)$$

Equation (27) can be simplified to

$$\left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)^* \otimes \left(\mathbf{F}\mathbf{e}_l(\mathrm{k})\right) = \left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)^*\left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)^T \otimes \mathbf{L}\boldsymbol{\varepsilon}_\mathrm{F}(k) + \left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)^* \otimes \mathbf{F}\mathbf{e}_l^o(k). \quad (29)$$

Substituting equation (29) into equation (26) we should have

$$\boldsymbol{\varepsilon}_\mathrm{F}(k+1) = \left(\mathbf{I} - \frac{\mu}{L}\sum_{l=0}^{\mathrm{K-1}} \mathbf{U}\left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)^*\left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)^T \otimes \mathbf{L}\right)\boldsymbol{\varepsilon}_\mathrm{F}(k) + \frac{\mu}{L}\sum_{l=0}^{\mathrm{K-1}}\left(\mathbf{F}\mathbf{u}_l(\mathrm{k})\right)\otimes\mathbf{F}\mathbf{e}_l^o(k). \quad (30)$$

Taking the expectation of the above equation yields

$$E\boldsymbol{\varepsilon}_\mathrm{F}(k+1) = \left(\mathbf{I} - \frac{\mu}{N}\sum_{l=0}^{\mathrm{K-1}} \mathbf{U}\mathbf{F}^H\mathbf{R}\mathbf{F}\otimes\mathbf{L}\right)E\boldsymbol{\varepsilon}_\mathrm{F}(k), \quad (31)$$

This result is similar to the result that corresponds to the DFT block LMS algorithm [10]. Therefore, the convergence of the block HOT-DFT LMS algorithm is similar to that of the block DFT LMS algorithm. The convergence speed of the HOT-LMS can be improved if we normalize using the estimated power of the tap-input vector in the DFT domain [11]. The complete HOT-DFT block LMS is given by

$$\mathbf{w}_F(k+1) = \mathbf{w}_F(k) + \frac{\mu}{L}\sum_{l=0}^{K-1} \mathbf{U}\Lambda_l^{-1}(k)\left(\mathbf{F}_{2N}\mathbf{u}_1\ (\mathrm{k})\right)^* \otimes \left(\mathbf{F}_{2N}\mathrm{e}_1(\mathrm{k})\right) \quad (32)$$

and

$$\Lambda_l(k+1) = \tfrac{k-1}{k}\Lambda_l(k) + \tfrac{1}{kL}Diag\{\left(\mathbf{F}_{2N}\mathbf{u}_1\ (\mathrm{k})\right)^* \otimes \left(\mathbf{F}_{2N}\mathrm{e}_1(\mathrm{k})\right)\}. \quad (33)$$

5. SIMULATION OF THE HOT-DFT LMS ALGORITHM

The learning curves of HOT-DFT block LMS algorithm are simulated. The desired input is generated using the linear model $d(n) = w(n)*u(n) + e_o(n)$, where $e_o(n)$ is the measurement white gaussian noise with variance of 10^{-8}. The input is a first-order Markov signal with autocorrelation functions given by $r(k) = 0.9^{|k|}$. The filter is low-pass with cutoff frequency of $\pi/2$ rad.

Fig. 1 shows the learning curves for the HOT-DFT block LMS with these conditions for the LMS and DFT block LMS with *N*

= *4* and *K* = *3*. Fig. 2 shows the same curves for *N* = *50* and *K* = *10*. Both figures show that the HOT-LMS block converges at the same rate as the DFT block LMS and yet it is computationally more efficient.

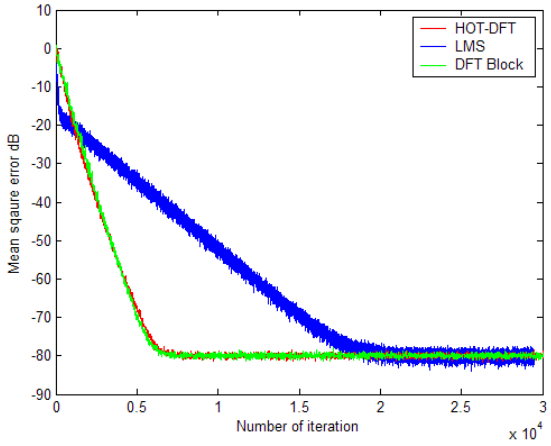

Figure 1 The learning curves for LMS, HOT-DFT block LMS, and DFT block LMS. *N* = *4* and *K* = *3*.

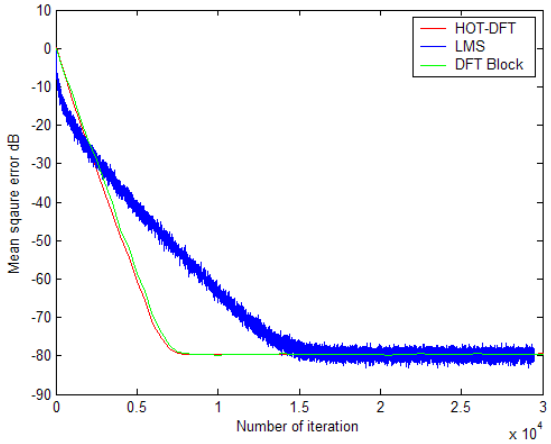

Figure 2: The learning curves for LMS, HOT-DFT block LMS, and DFT block LMS. *N* = *50* and *K* = *10*.

5. CONCLUSIONS

This paper has introduced a new block LMS algorithm, the block HOT-DFT LMS algorithm, based on the fast HOT convolution. The algorithm assumes that the filter size is much smaller than the block size. This algorithm reduces the computational complexity of the block DFT LMS by about 30% without reducing convergence performance. The convergence analysis and simulations confirm that the block HOT-DFT LMS and block DFT LMS converge at the same rate. Thus, our algorithm is suitable for a wide variety of applications as a more efficient implementation of the block LMS algorithm.

7. REFERENCES

[1] V. DeBrunner and E. Matusiak, "An algorithm to reduce the complexity required to convolve finite length sequences using the Hirschman optimal transform (HOT)," *ICASSP'03*, Hong Kong, China, pp. II-577-580, Apr 2003.

[2] E. R. Ferrara, "Fast implementation of LMS adaptive filters," *IEEE Trans. ASSP*, vol. ASSP-28, NO. 4, Aug 1980.

[3] G. Clark, S. Mitra, and S. Parker, "Block implementation of adaptive digital filters," *IEEE Trans. ASSP*, pp. 744-752, Jun 1981.

[4] V. DeBrunner, M. Özaydın, and T. Przebinda, "Resolution in time-frequency," *IEEE Trans. SP*, pp. 783-788, Mar 1999.

[5] T. Przebinda, V. DeBrunner, and M. Özaydın, "The optimal transform for the discrete Hirschman uncertainty principle," *IEEE Trans. Infor. Theory*, pp. 2086-2090, Jul 2001.

[6] O. Alkhouli, V. DeBrunner, Y. Zhai and M. Yeary, "FIR Adaptive filters based on Hirschman optimal transform," IEEE/SP 13th Workshop on Statistical Signal Processing, 2005.

[7] O. Alkhouli and V. DeBrunner, "Convergence Analysis of Hirschman Optimal Transform (HOT) LMS adaptive filter," IEEE/SP 14th Workshop on Statistical Signal Processing, 2007.

[8] O. Alkhouli and V. DeBrunner, "Hirschman optimal transform block adaptive filter," *International conference on Acoustics, Speech, and Signal Processing (ICASSP),* 2007.

[9] S. Mitra, *Digital Signal Processing*. Mc Graw Hill, Second edition, 2000.

[10] B. Farhang-Boroujeny and Kheong Sann Chan, "Analysis of the Frequency-Domain Block LMS Algorithm," *IEEE Trans. ASSP*, pp. 2332, Aug. 2000.

[11] Jae Chon Lee and Chong Kwan Un, "Performance analysis of frequency-domain block LMS adaptive digital filters," *Circuits and Systems, IEEE Transactions on*, Volume 36, Issue 2, Page(s):173-189, Feb. 1989.