Unconstrained Hartley Domain Least Mean Square Adaptive Filter

P. K. Meher and G. Panda

Abstract—**An efficient adaptive filtering algorithm named as the unconstrained Hartley domain least mean square (UHLMS) algorithm has been proposed. It is found from computer simulation that the proposed algorithm has similar performance as the time domain least mean square (LMS) algorithm [1] for uncorrelated signal; but yields faster and better convergence compared to the other for highly correlated signal. The UHLMS algorithm has identical performance as those of the unconstrained frequency domain least mean square (UFLMS) algorithm [3] and [4], but requires significantly less computation compared to the others.**

I. Introduction

The tap delay line (TDL) filters whose filter weights are updated by the Widrow–Hoff least mean square (LMS) algorithm [1] may be considered as the simplest known adaptive filter. The convergence speed of the LMS algorithm is, however, greatly reduced with the increase of the eigenvalue ratio $\lambda \max/\lambda \min$ of the input auto correlation matrix. The LMS algorithm, therefore, has poor convergence for highly correlated input [2]. Mansour and Gray [3] have proposed the unconstrained frequency domain least mean square (UFLMS) algorithm which offers faster convergence compared to the LMS heavily coloured signal by squeezing the eigenvalue ratio. Wong and Kwong [4] have proposed to compute the linear convolution of UFLMS [3] using the discrete Hartley transform (DHT) of the input signal and the real and the imaginary parts of the Fourier domain weights. For the sake of convenience, we have referred this algorithm as unconstrained split Fourier least mean square algorithm (USFLMS) because it separately uses the real and the imaginary parts of the Fourier domain weights updated according to the UFLMS [3]. It has been shown [4] that the USFLMS costs less computation compared to the UFLMS algorithm.

In this paper we have proposed the Hartley domain realization of the UFLMS algorithm [3], hereafter mentioned as the unconstrained Hartley domain least mean square (UHLMS) algorithm to distinguish it from its Fourier domain counterpart [3]. It is shown that the proposed algorithm has similar performance as the time domain LMS algorithm [1] for uncorrelated signal; but yields faster and better convergence compared to the other for highly correlated signals. Apart from that, the proposed adaptive algorithm has similar performance as those of [3] and [4] when simulated under identical conditions. Another advantage of the UHLMS is that for an N-th order filter it requires only $2N$ real filter weights, while the UFLMS requires $2N$ complex weights and USFLMS requires $4N$ real weights [4] for the filter of the same order. Besides, it requires significantly less computation compared to the USFLMS for all possible filter orders.

Manuscript received June 15, 1992; revised manuscript received February 3, 1993. This paper was recommended by Associate Editor G. S. Moschytz.

The authors are with the Department of Applied Electronics and Instrumentation Engineering, Regional Engineering College, Rourkela 769 008, India.

IEEE Log Number 9210037.

II. Unconstrained Hartley Domain Least Mean Square Adaptive Algorithm

According to the UFLMS algorithm [3] the $(k+1)$-th weight vector $\boldsymbol{w}_{k+1}$ is obtained from the k-th weight vector $\boldsymbol{w}_k$ by the following set of equations,

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + s_k[\bar{\boldsymbol{X}}_k \odot \boldsymbol{E}_k], \tag{1}$$

$$\boldsymbol{E}_k = \boldsymbol{F}\boldsymbol{e}_k, \tag{2}$$

$$\boldsymbol{e}_k = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I}_N \end{bmatrix}[\boldsymbol{d}_k - \boldsymbol{y}_k], \tag{3}$$

$$\boldsymbol{y}_k = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I}_N \end{bmatrix}\boldsymbol{F}^{-1}[\boldsymbol{X}_k \odot \boldsymbol{w}_k]. \tag{4}$$

The over bar in (1) represents complex conjugate, and the symbol '$\odot$' denotes the scalar product. $\boldsymbol{I}_N$ is an $N \times N$ identity matrix, $\boldsymbol{0}$ is an $N \times N$ null matrix and $\boldsymbol{F}$ is $2N \times 2N$ DFT matrix.

$\boldsymbol{X}_k$ is a $2N$-point column vector given by,

$$\boldsymbol{X}_k = \boldsymbol{F}\boldsymbol{x}_k, \tag{5}$$

where $\boldsymbol{X}_k$ contains the $2N$-point input signal given by,

$$\boldsymbol{x}_k = [x(kN-N)\cdots\cdots x(kN-1)x(kN)\cdots \cdot(kN+N-1)]^T. \tag{6}$$

$\boldsymbol{d}_k$ is the N-point column vector constituting the desired signal and $\boldsymbol{e}_k$ is the $2N$-point error vector of k-th iteration. $\boldsymbol{S}_k$ is the $2N$-point step-size vector, whose elements are given by,

$$S_k(i) = \alpha / z_k(i), \tag{7}$$

where

$$z_k(i) = (1-\beta)z_{k-1}(i) + \beta|X_k(i)|^2. \tag{8}$$

α and β are called as the energy smoothing factors whose values lie between 0 and 1. $z_k(i)$ is an estimate of energy of at the i-th frequency.

The Fourier domain circular correlation in the second term on the R.H.S. of (1) and the Fourier domain circular convolution of (4) associated with the UFLMS algorithm are required to be replaced by, the equivalent DHT based representations, in the Hartley domain adaptive algorithm. Therefore, to obtain the UHLMS algorithm, (1) and (4) may, respectively, be replaced by

$$H_{k+1}(i) = H_k(i) + \mu_k(i)[Xe_k(i)C_k(i) - Xo_k(i)C_k(2N-i)] \tag{9}$$

and

$$\{y_k(i)\} = \mathrm{IDHT}\{[Xe_k(i)H_k(i) + Xo_k(i)H_k(2N-i)]\} \quad \text{for } i = 0,1,2,\cdots,2N-1 \tag{10}$$

where $\{H_k(i)\}$ and $\{H_{k+1}(i)\}$ are the old and the new Hartley domain weights, respectively. $\{Xe_k(i)\}$ and $\{Xo_k(i)\}$, respectively, represent the even and the odd parts of the $2N$-point DHT of x_k. $\{C_k(i)\}$ represents the $2N$-point DHT of e_k.

Similarly, the step size adaptation (7) and (8) are equivalently replaced by the following Hartley domain equations,

$$\mu_k(i) = \alpha / z_k(i) \tag{11}$$

$$z_k(i) = (1-\beta)z_{k-1}(i) + \beta\left[Xe_k(i)^2 + Xo_k(i)^2\right] \quad \text{for } i = 0,1,\cdots,2N-1 \tag{12}$$

The adaptive filter structure employing the proposed adaptive algorithm given by (3), (6), and (9)–(12) is depicted in Fig. 1.

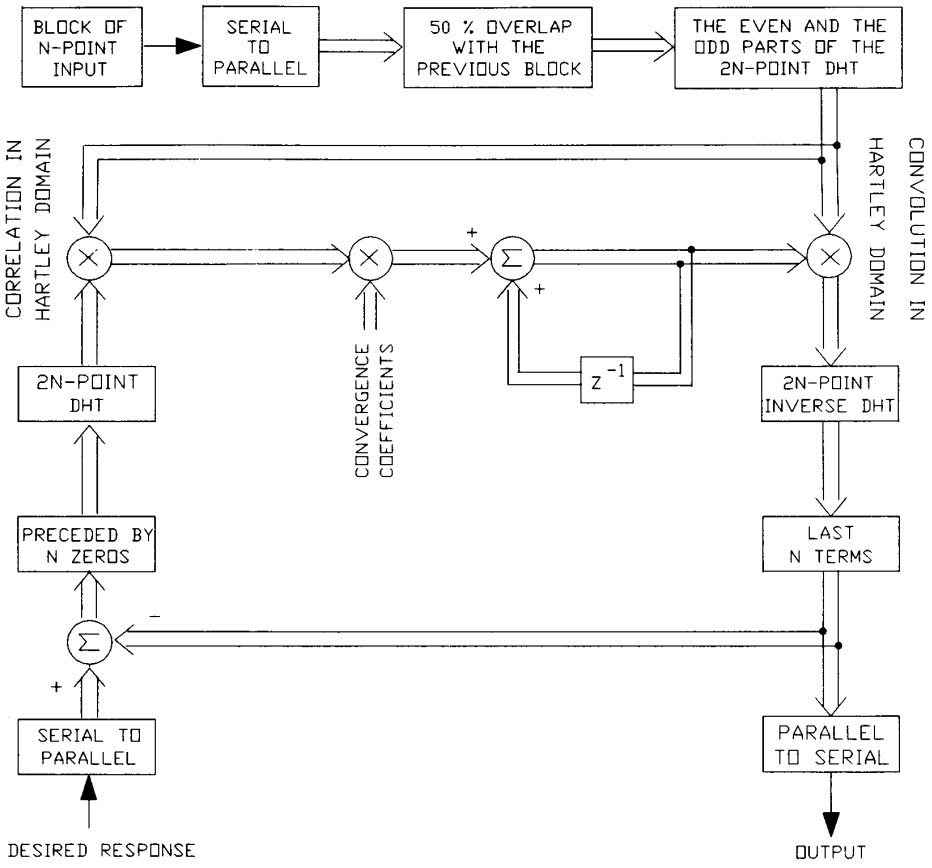

Fig. 1. The structure of the proposed unconstrained Hartley domain least mean square adaptive filter.

III. Simulation Results

The LMS, UFLMS, USFLMS, and the proposed algorithm are simulated for a system identification problem. The simulation configuration is shown in Fig. 2. The system to be identified is a 32-point finite impulse response (FIR) filter. The values of the filter coefficients of the system used for the simulation are taken from [3]. The simulation is carried out in two parts. In the first part of the simulation random white noise (distributed uniformly between -0.5 and $+0.5$) is used as the input. For the second part of the simulation a highly correlated input is obtained by passing the random white noise of the first part of the simulation, through a 12-th order all-pole filter. The coefficients of the all-pole filter are also taken from [3]. In both the parts of the simulation the fixed system is contaminated with uncorrelated white noise of -40 db strength compared to unity signal power. The smoothing factor β is taken to be 0.8 for both correlated as well as uncorrelated signals. The values of α are taken to be 0.4 and 0.6 for correlated and uncorrelated signals, respectively. For the time domain LMS the convergence factor is taken to be 0.006 and 0.011 for the correlated and the uncorrelated signals, respectively, to maintain the same misadjustments as those of the others.

The noise to signal ratio (NSR) in decibels, obtained by the ratio of the error signal power to the desired response power for different algorithms are shown in Fig. 3 for uncorrelated signal and in Fig. 4 for correlated signal. It may be noted that each of the convergence curve is obtained by averaging the results of 20 ensembles. From the convergence curves one may observe that the LMS algorithm as well as the proposed one yield similar performances for uncorrelated input. But, for correlated input the proposed algorithm offers much better convergence performance over the other. The UFLMS, the USFLMS and the proposed algorithm, however, have similar performances for the correlated as well as uncorrelated signals when simulated under identical conditions.

IV. Comparison of Computational Complexity

For every iteration of UHLMS algorithm one has to compute:

(i) the even and the odd parts of a $2N$-point DHT of the signal block $\boldsymbol{x}_k$ to be used for weight updating and Hartley domain convolution by (9) and (10), respectively,
(ii) a $2N$-point DHT of the error vector $\boldsymbol{e}_k$ for (9) and
(iii) an inverse DHT for (10).

The even and the odd parts of the DHT may, however, be conveniently obtained from a $2N$-point DFT of real-valued data. Again, the inverse DHT is identical to the forward DHT, except a scale factor. The computational load per every iteration of the UHLMS algorithm, therefore, amounts to one $2N$-point DFT of real-valued data and two $2N$-point DHT's along with $(6N - 2)$ multiplications and $(4N - 2)$ additions for weight updating by (9), $(4N - 2)$ multiplications and $(2N - 2)$ additions for convo-

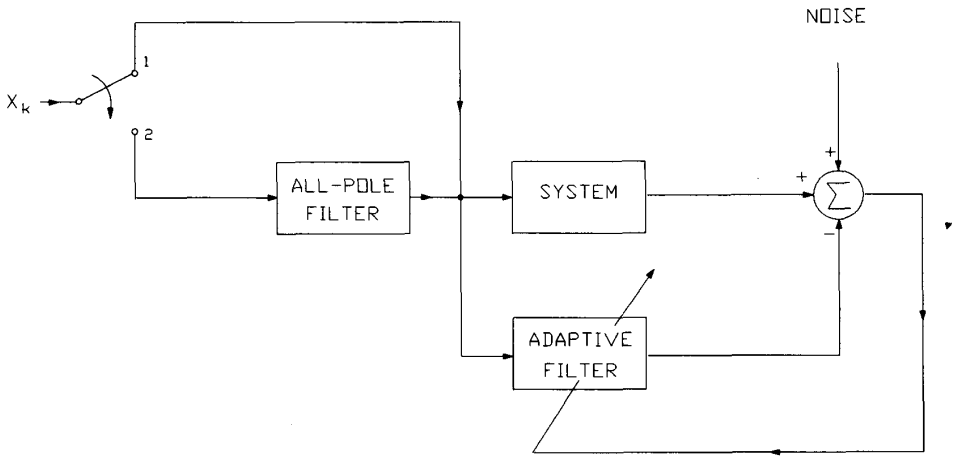

Fig. 2. The simulation configuration.

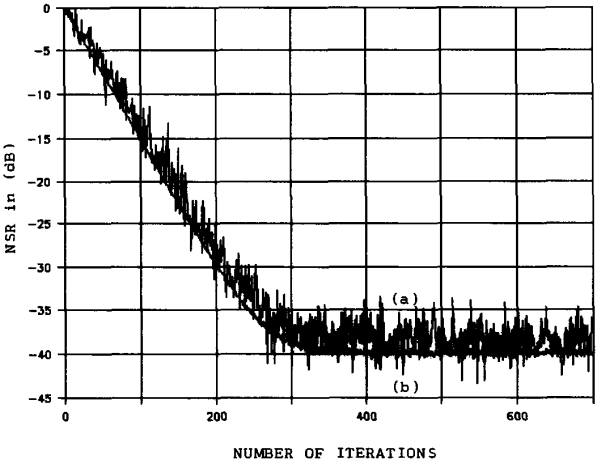

Fig. 3. The convergence curves of different algorithms for uncorrelated input. (a) For the LMS algorithm and (b) for the UFLMS, USFLMS, and the proposed algorithms.

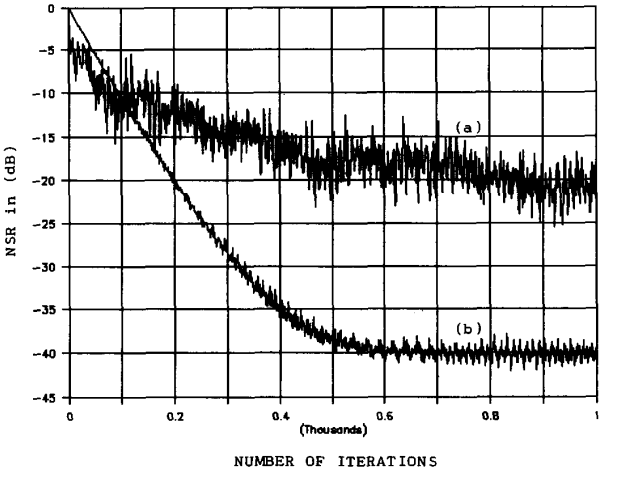

Fig. 4. The convergence curves of different algorithms for correlated input. (a) For the LMS algorithm and (b) for the UFLMS, USFLMS, and the proposed algorithms.

lution (10) and N additions for the calculation of N number of error components by (3). Out of the $2N$ number step-size adaptation coefficients '$\mu_k(i)$' $(N-1)$ number are redundant. Therefore, only $(4N+2)$ multiplications and $2N$ additions are required for the step size adaptation by (12). The actual number of arithmetic operations required by the adaptive algorithm depends on the fast algorithms used to compute the DHT and the DFT. Using the radix-2 FFT of real-data and FHT [5] the UHLMS algorithm would require $[6N(\log_2 N) + 2N + 10]$ multiplications and $[9N(\log_2 N) + 7N + 4]$ additions. Accordingly, we have calculated the number of multiplications and additions required by the UHLMS algorithm and listed them in Table I, along with those of USFLMS algorithm, for comparison. For calculating the computational requirement of the USFLMS algorithm $(4N-2)$ multiplications and $2N$ additions are deducted from the operation counts given by Wong and Kwong [4] to account for the redundancies in step-size adaptation.

It may be observed from Table I that the UHLMS algorithm offers significant saving of multiplications as well as additions over the USFLMS algorithm [4] for all possible filter lengths.

TABLE I
COMPUTATIONAL REQUIREMENTS OF USFLMS AND UHLMS ALGORITHMS

Filter Order N	UHLMS Algorithm		USFLMS Algorithm	
	Multiplications	Additions	Multiplications	Additions
8	170	276	222	350
16	426	692	526	838
32	1034	1668	1230	1958
64	2442	3908	2830	4486
128	5642	8964	6414	10118

V. CONCLUSION

An efficient Hartley domain adaptive algorithm is presented. It is shown that the proposed algorithm has similar performance as the time domain least mean square (LMS) algorithm [1] for uncorrelated signal; but yields faster and better convergence

compared to the other for highly correlated signal. Besides, the proposed algorithm has similar performance as UFLMS [3] and USFLMS [4] for correlated as well as uncorrelated input. Another advantage of this algorithm over the UFLMS and the USFLMS is that it uses only $2N$ real filter weights to be updated in every iteration while the UFLMS requires $2N$ complex weights and the USFLMS on the other hand requires $4N$ real filter weights [4] for an N-point filter. Apart from these, the proposed algorithm offers considerable saving of multiplications as well as addition over the USFLMS algorithm for various filter orders.

References

[1] B. Widrow *et al.*, "Adaptive noise canceling: Principles and applications," *Proc. IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.

[2] S. S. Narayan, A. M. Peterson, and M. J. Narasimha, "Transform domain LMS algorithm," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-31, pp. 609–615, June 1983.

[3] D. Mansour and A. H. Gray, Jr., "Unconstrained frequency-domain adaptive filter," *IEEE Trans. Acoust. Speech, Signal Process.*, vol. ASSP-30, pp. 726–734, Oct. 1982.

[4] T. W. Wong and C. P. Kwong, "Adaptive filtering using Hartley transform and overlap-save method," *IEEE Trans. Signal Process.*, vol. SP-39, pp. 1708–1711, July 1991.

[5] H. V. Sorensen *et al.*, "On computing the discrete Hartley transform," *IEEE Trans. Acoust. Speech, Signal Process.*, vol. ASSP-33, Oct. 1985.

Circuit Implementation of a Peak Detector Neural Network

G. L. Dempsey and E. S. McVey

Abstract—**Peak detection is a basic data analysis problem which is essential in a large number of applications. In applications such as image processing, the large computational effort to locate peaks may prohibit operation in real-time. A Hopfield neural network is proposed for the peak detector to solve the real-time problem. Analytical expressions are derived for input separation, neuron gain, and restrictions on initial conditions. Hardware limitations are discussed and a modified circuit model is suggested for the Hopfield neuron. Solution time under thirty microseconds is obtainable with general purpose operational amplifiers independent of the number of inputs. Results obtained from a twenty-five neuron hardware implementation of the network lend credence to the theoretical results.**

I. Introduction

Image processing and waveform analysis problems are applications where a peak detector (PD) that is capable of operating in real-time may be required. The authors have suggested [1], [2] using the PD described here as one of two neural networks to implement the Hough transform [3]. Other researchers have investigated the peak detection problem using conventional digital technology [4]–[6] but because of the digital architecture real-time applications are limited. Recent neural research includes [7]–[10]. Results are presented in this paper that have not been addressed in other research.

The reduction in the computational time to locate peak values in a large input array is the advantage of a neural implementation over digital architectures. The convergence time of the network suggested here is independent of the number of inputs and can be designed to be less than 30 us with general purpose operational amplifiers (op-amps) which is several orders of magnitude less than the digital algorithms suggested in [4]–[6]. Experimental results from a twenty-five neuron hardware implementation are presented.

The PD is based on the optimization network (analog version) of Hopfield and Tank [11], [12]. Hardware implementation of large networks using the Hopfield neuron model becomes impractical because the neuron gain must be increased in direct proportion to the network size. A modified neuron model is suggested which requires a gain of approximately three for the PD application independent of the network size. The modified model can be used in other Hopfield applications which attaches additional importance to the results.

II. Peak Detector Architecture

The PD was designed to converge to a digital solution to allow interfacing with digital logic circuitry or a main processing unit (MPU). One neuron is required per input. The neuron with the peak input converges to logic "1." All other neurons converge to logic "0." A conventional digital logic circuit as suggested in [2] can be used to condense or encode the PD outputs into a form suitable for a MPU.

The PD neuron using the Hopfield network topology is shown in Fig. 1. The nonlinear differential equation describing each neuron i is

$$C_i(dU_i/dt) = -U_i/R_i + I_i + \sum_{\substack{j=1 \\ j\neq i}}^{n} T_{ij}V_j, \tag{1}$$

where n is the number of neurons, U_i is the input voltage of the neuron, I_i is the input current, C_i is the input capacitance, and R_i is the input resistance which is the parallel combination of the neuron input resistance p_i and the connection weights $1/T_{ij}$ or R_{ij}. The neuron time constant is defined as

$$\text{neuron time constant } \tau = R_iC_i. \tag{2}$$

Because each neuron can be described by (1), the network can be simulated by solving n differential equations using any convenient numerical method. The normal (noninverted) neuron output will be bounded by $[0, Vmax]$ while the inverted output will be bounded by $[0, -Vmax]$ where $Vmax$ is the saturation voltage of the op-amp. It was shown in [1] that the T_{ij} conductances are of equal value and form inhibitory connections. For simplicity, the $1/p_i$ conductance will be made equal to the T_{ij} values. Therefore, the input resistance of the neuron when connected in the network is

$$R_i = 1/nT. \tag{3}$$

Hopfield and Tank have showed numerous examples of obtaining network energy functions [12], [13] using the general Cohen–Grossberg energy function [14],

$$E = -0.5\sum_{i\neq j}\sum T_{ij}V_iV_j - \sum_i I_iV_i + \sum_i U_iV_i. \tag{4}$$

Manuscript received January 30, 1991; revised manuscript received December 21, 1991, and November 17, 1992. This work was supported in part by NASA. This paper was recommended by Associate Editor A. Kuh.

G. L. Dempsey is with the Department of Electrical Engineering and Technology, Bradley University, Peoria, IL 61625.

E. S. McVey is with the School of Engineering and Applied Science, University of Virginia, Charlottesville, VA 22901.

IEEE Log Number 9210029.