14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

ON THE EQUIVALENCE OF A REDUCED-COMPLEXITY RECURSIVE POWER NORMALIZATION ALGORITHM AND THE EXPONENTIAL WINDOW POWER ESTIMATION

Kutluyıl Doğançay

School of Electrical and Information Engineering, University of South Australia
Mawson Lakes, SA 5095, Australia
phone: + (61) 8 8302 3984, fax: + (61) 8 8302 3384, email: kutluyil.dogancay@unisa.edu.au
web: http://people.unisa.edu.au/Kutluyil.Dogancay

ABSTRACT

The transform-domain least-mean-square (TD-LMS) algorithm provides significantly faster convergence than the LMS algorithm for coloured input signals. However, a major disadvantage of the TD-LMS algorithm is the large computational complexity arising from the unitary transform and power normalization operations. In this paper we establish the equivalence of a recently proposed recursive power normalization algorithm and the traditional exponential window power estimation algorithm. The proposed algorithm is based on the matrix inversion lemma and is optimized for implementation on a digital signal processor (DSP). It reduces the number of divisions from N to one for a TD-LMS adaptive filter with N coefficients. This provides a significant reduction in computational complexity for DSP implementations. The equivalence of the reduced-complexity algorithm and the exponential window power estimation algorithm is demonstrated in simulation examples.

1. INTRODUCTION

Computational complexity and convergence speed are two important considerations for adaptive filtering algorithms [1]. The computational complexity of an adaptive filtering algorithm is proportional to the number of filter coefficients. For example, the computational complexity of the least-mean-square (LMS) algorithm is $O(N)$ for an adaptive finite impulse response (FIR) filter with N coefficients. The convergence speed of stochastic gradient descent adaptive filtering algorithms such as the LMS is dependent on the eigenspread of the autocorrelation matrix of the input signal. In general, the larger the eigenspread, the slower the convergence. For LMS-type adaptive filters, the most challenging adaptive filtering problems are those that require long adaptive filters and have input signals with large eigenspread. For such problems, the adaptive filtering algorithm would require large computational complexity and exhibit rather slow convergence.

A remedy for slow convergence is to whiten the adaptive filter input signal by using an orthogonal transform followed by power normalization. This is also known as *transform-domain adaptive filtering* [2]. While the use of an orthogonal transform and power normalization speeds up the algorithm convergence by effectively reducing the eigenspread of the adaptive filter input, it does not readily lead to a reduction in computational complexity or the required number of filter coefficients. The complexity of the orthogonal transform and power normalization can be reduced to a certain extent by employing *generalized subband decomposition* (GSD) [3]. GSD essentially enables the use of smaller-size transforms resulting in less division operations. The computational complexity of long adaptive filters can also be reduced by employing block LMS algorithms in the frequency domain at the expense of an end-to-end delay [4]. A particularly attractive method for complexity reduction is *selective partial updating* [5], which was applied to transform domain adaptive filters in [6]. To allow complexity reduction by selective partial updating, a new power normalization algorithm was also developed.

In this paper we provide a detailed analysis of this recently proposed recursive power normalization algorithm [6], which requires only one division rather than N divisions for a TD-LMS adaptive filter with N coefficients. We study the equivalence of the proposed recursive power estimation algorithm and the exponential window power estimation algorithm, which is traditionally used in TD-LMS implementations. A comparison is also provided between the two power estimation algorithms in terms of their respective cycle counts when implemented on a current digital signal processor (DSP), revealing the significant complexity reduction that is achieved by employing the reduced-complexity recursive power estimation algorithm proposed in [6]. The recursive power estimation algorithm is optimized for DSP implementation by taking advantage of uncorrelated transform outputs and inexpensive multiply operations. Compared with the recursive least squares (RLS) algorithm [1] which has a complexity of $O(N^2)$, the proposed recursive algorithm only requires a complexity of $O(N)$.

The paper is organized as follows. Section 2 summarizes the TD-LMS algorithm and its key components. Section 3 concentrates on the power normalization component of the TD-LMS algorithm. A complexity analysis is provided for the traditional exponential window power estimation algorithm. A detailed derivation of the reduced-complexity recursive algorithm is presented based on the matrix inversion lemma. In Section 4, computer simulations are presented to demonstrate the equivalence of the proposed algorithm and the exponential window power estimation algorithm. The conclusions are drawn in Section 5.

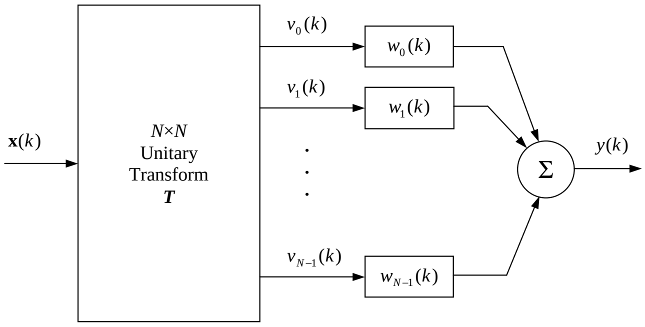

Figure 1: TD-LMS structure.

2. OVERVIEW OF TD-LMS

For coloured input signals with large eigenspread, the LMS and normalized LMS (NLMS) algorithms exhibit rather slow convergence. For such signals, the TD-LMS algorithm converges significantly faster. This improvement is brought about by approximate decorrelation (whitening) of the input regressor vector to the adaptive filter. Decorrelation is accompanied by power normalization. The TD-LMS algorithm is defined by the recursion [2]

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \mu\, e(k)\boldsymbol{\Lambda}^{-2}\boldsymbol{v}^*(k) \qquad (1)$$

where

$$\boldsymbol{v}(k) = [v_0(k), \ldots, v_{N-1}(k)]^T = \boldsymbol{T}\boldsymbol{x}(k)$$

is the transformed regressor vector (the superscript * denotes complex conjugate), $\boldsymbol{\Lambda}^2 = \mathrm{diag}\{\sigma_0^2, \sigma_1^2, \ldots, \sigma_{N-1}^2\}$ is the power matrix of transform outputs, and $e(k) = d(k) - \boldsymbol{w}^T(k)\boldsymbol{v}(k)$ is the error signal. The TD-LMS structure is illustrated in Fig. 1. The power matrix can be estimated online using a sliding exponential window:

$$\sigma_i^2(k) = \lambda\sigma_i^2(k-1) + |v_i(k)|^2, \quad i = 0, \ldots, N-1 \qquad (2)$$

where λ is a forgetting factor for the exponential window ($0 < \lambda < 1$). Using (2), the power matrix $\boldsymbol{\Lambda}^2$ in (3) is replaced by $\boldsymbol{\Lambda}^2(k) = \mathrm{diag}\{\sigma_0^2(k), \ldots, \sigma_{N-1}^2(k)\}$, resulting in

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \mu\, e(k)\boldsymbol{\Lambda}^{-2}(k)\boldsymbol{v}^*(k). \qquad (3)$$

The transform $\boldsymbol{T}$ is a fixed $N \times N$ orthogonal or unitary matrix obtained from a discrete-time transform such as the discrete Fourier transform (DFT), the discrete cosine transform (DCT), the discrete Wavelet transform (DWT) or the discrete Hartley transform (DHT), to name but a few. The optimal transform is derived from the autocorrelation matrix of the input signal, and is known as the Karhunen-Loève transform (KLT). However, because the KLT is signal-dependent and has a large computational complexity, it is not employed in practical applications.

The computational complexity associated with obtaining the transformed signal vector $\boldsymbol{v}(k)$ from the input signal $x(k)$ has been studied in the open literature, and several low-complexity implementations are available. For example, most of the orthogonal transforms can be implemented as IIR filter banks [7, 8] or as sliding window transforms [9] with computational complexity of $O(N)$.

3. POWER NORMALIZATION

Power normalization is an important component of transform-domain adaptive filters. Without power normalization, no improvement can be achieved in the convergence speed. The computational complexity of power normalization is also very significant because of the large number of division operations for which the DSP processors are not optimized.

For the exponential window power estimation in (2), the computational effort required for each transform output $v_i(k)$ is 2 multiplications (one for squaring modulus of $v_i(k)$ and another to compute $\lambda\sigma_i^2(k-1)$). Referring to (3), power normalization requires one division for each $v_i^*(k)$:

$$\frac{v_i^*(k)}{\sigma_i^2(k)}, \quad i = 0, \ldots, N-1. \qquad (4)$$

Thus, the total computational complexity for power normalization is $2N$ multiplications and N divisions.

The large number of divisions in power normalization is a major obstacle to the adoption of the TD-LMS for practical applications [10]. An alternative method for exponential window power estimation followed by division is to do both in one step by resorting to the *matrix inversion lemma* [1]:

$$(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}\boldsymbol{D})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{B} + \boldsymbol{C}^{-1})^{-1}\boldsymbol{D}\boldsymbol{A}^{-1}. \qquad (5)$$

The objective of power normalization is ultimately to estimate the reciprocal of power for each transform output to be used in (3):

$$\boldsymbol{\Lambda}^{-2}(k) = \begin{bmatrix} 1/\sigma_0^2(k) & & & \boldsymbol{0} \\ & 1/\sigma_1^2(k) & & \\ & & \ddots & \\ \boldsymbol{0} & & & 1/\sigma_{N-1}^2(k) \end{bmatrix} = \left(\lambda\boldsymbol{\Lambda}^2(k-1) + \begin{bmatrix} |v_0(k)|^2 & & \boldsymbol{0} \\ & \ddots & \\ \boldsymbol{0} & & |v_{N-1}(k)|^2 \end{bmatrix} \right)^{-1}. \qquad (6)$$

In order to ensure that the matrix inversion lemma results in complexity reduction, the term $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{B} + \boldsymbol{C}^{-1}$ in the right-hand side of (5) must be scalar so that matrix inversion is avoided. It turns out that this can only be achieved if we set $\boldsymbol{A} = \lambda\boldsymbol{\Lambda}^2(k-1)$, $\boldsymbol{B} = \boldsymbol{D}^H = \boldsymbol{v}(k)$ and $\boldsymbol{C} = 1$, leading to an approximation to (6):

$$(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{C}\boldsymbol{D})^{-1} = \left(\lambda\boldsymbol{\Lambda}^2(k-1) + \boldsymbol{v}(k)\boldsymbol{v}^H(k)\right)^{-1} \approx \boldsymbol{\Lambda}^{-2}(k). \qquad (7)$$

In the averaged sense this is a good approximation because the transform $\boldsymbol{T}$ is assumed to whiten the input signal. Thus, $E\{\boldsymbol{v}(k)\boldsymbol{v}^H(k)\}$ would ideally be a diagonal matrix. We will replace the above approximation with equality in the following derivations. The application of the matrix inversion lemma to (6) dispenses with the need for implicit inverse power calculation for the transform outputs:

$$\begin{aligned}\boldsymbol{\Lambda}^{-2}(k) &= \frac{1}{\lambda}\boldsymbol{\Lambda}^{-2}(k-1) \\ &\quad - \frac{1}{\lambda^2}\boldsymbol{\Lambda}^{-2}(k-1)\boldsymbol{v}(k)\boldsymbol{v}^H(k)\boldsymbol{\Lambda}^{-2}(k-1) \\ &\quad \times \Big(\frac{1}{\lambda}\boldsymbol{v}^H(k)\boldsymbol{\Lambda}^{-2}(k-1)\boldsymbol{v}(k)+1\Big)^{-1} \\ &= \frac{1}{\lambda}\boldsymbol{\Lambda}^{-2}(k-1) \\ &\quad - \frac{\boldsymbol{\Lambda}^{-2}(k-1)\boldsymbol{v}(k)\boldsymbol{v}^H(k)\boldsymbol{\Lambda}^{-2}(k-1)}{\lambda^2+\lambda\boldsymbol{v}^H(k)\boldsymbol{\Lambda}^{-2}(k-1)\boldsymbol{v}(k)}.\end{aligned} \tag{8}$$

The second term in the right-hand side of (8) is not diagonal. In order to calculate only the diagonal entries, we modify (8) to

$$\begin{aligned}\boldsymbol{\Lambda}^{-2}(k) &= \frac{1}{\lambda}\boldsymbol{\Lambda}^{-2}(k-1) \\ &\quad - \frac{1}{\lambda^2+\lambda\boldsymbol{v}^H(k)\boldsymbol{\Lambda}^{-2}(k-1)\boldsymbol{v}(k)} \\ &\quad \times \boldsymbol{\Lambda}^{-2}(k-1)\begin{bmatrix}|v_0(k)|^2 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & |v_{N-1}(k)|^2\end{bmatrix} \\ &\quad \times \boldsymbol{\Lambda}^{-2}(k-1).\end{aligned} \tag{9}$$

Letting $\phi_i(k) = 1/\sigma_i^2(k)$, (9) can be conveniently rewritten as

$$\begin{bmatrix}\phi_0(k) \\ \vdots \\ \phi_{N-1}(k)\end{bmatrix} = \frac{1}{\lambda}\begin{bmatrix}\phi_0(k-1) \\ \vdots \\ \phi_{N-1}(k-1)\end{bmatrix} - \frac{1}{1+\frac{1}{\lambda}\sum_{i=0}^{N-1}|v_i(k)|^2\phi_i(k-1)} \times \begin{bmatrix}|v_0(k)|^2\Big(\frac{\phi_0(k-1)}{\lambda}\Big)^2 \\ \vdots \\ |v_{N-1}(k)|^2\Big(\frac{\phi_{N-1}(k-1)}{\lambda}\Big)^2\end{bmatrix}. \tag{10}$$

The computational steps of the above recursion can be performed as detailed below:

$$\frac{1}{\lambda}\phi_i(k-1), \quad (N \text{ muls}) \tag{11a}$$

$$\Big(\frac{1}{\lambda}\phi_i(k-1)\Big)v_i(k)v_i^*(k), \quad (2N \text{ muls}) \tag{11b}$$

$$\Big(\frac{1}{\lambda}\phi_i(k-1)|v_i(k)|^2\Big)\Big(\frac{1}{\lambda}\phi_i(k-1)\Big), \quad (N \text{ muls}) \tag{11c}$$

$$\frac{1}{1+\frac{1}{\lambda}\sum\cdots}\Big[\quad\Big] \quad (N \text{ adds, 1 div, } N \text{ muls}) \tag{11d}$$

Power Norm. Algorithm	Complexity		Min. Cycle Count (C62x)
	Muls	Divs	
Exp. window	$2N$	N	$17N$
Rec. algorithm	$5N$	1	$\frac{5}{2}N+16$

Table 1: Complexity comparison for power normalization algorithms [6].

The above procedure generates the reciprocal of the transform output power values recursively, as desired. However, the reciprocal power values must be multiplied with the transform outputs to achieve the required power normalization. If this is done explicitly, it would increase the number of multiplications by N. In order to save N multiplications, we use (11b) in the recursive power estimation algorithm, which produces delayed power normalized transform outputs:

$$\frac{1}{\lambda}\phi_i(k-1)v_i^*(k), \quad i = 0, \ldots, N-1. \tag{12}$$

Compared with (4), which would result from the application of (2) and direct divisions, the outputs of the recursive power normalization algorithm in (12) are scaled by a constant $1/\lambda$ and use one-sample-delayed power estimates $\sigma_i^2(k-1)$. The constant scaling factor can easily be absorbed into the stepsize parameter μ, and therefore has no effect on power normalization. The one-sample delay for the power estimate used in normalizing the $v_i^*(k)$ will have no noticeable impact on the performance of TD-LMS since the tracking performance of the TD-LMS algorithm will hardly be influenced by this delay. Thus, we can use (12) instead of (4) as the power normalized transform output signal with no adverse effect on the TD-LMS performance whatsoever.

As shown in (11), the total computational complexity of (10) is $5N$ multiplications and one division. Compared with the power normalization method based on (2) and (4), the recursive power normalization algorithm in (10) reduces the number of divisions from N to one at the expense of a 2.5-fold increase in the number of multiplications (see Table 1).

On DSP processors, the actual complexity of (10) would be significantly smaller than that of (2) and (4) despite an increased number of multiply operations. The reason for this lies in the way the DSP architectures have evolved over the years. Modern DSP processors have highly optimized architectures for the multiply-and-accumulate (MAC) operation as it is one of the most common operations performed in signal processing applications [11]. This means that while DSP processors will perform multiply and MAC operations very fast, the less common operations such as division will require longer processing times. For example, on Texas Instruments DSP TMS320C62x, which is one of the high performance fixed-point DSP processors currently available on the market, a MAC operation can be performed in half cycle, while a division operation takes between 16–41 cycles [12]. This implies that on the C62x the complexity of a division operation is at least 32 times larger

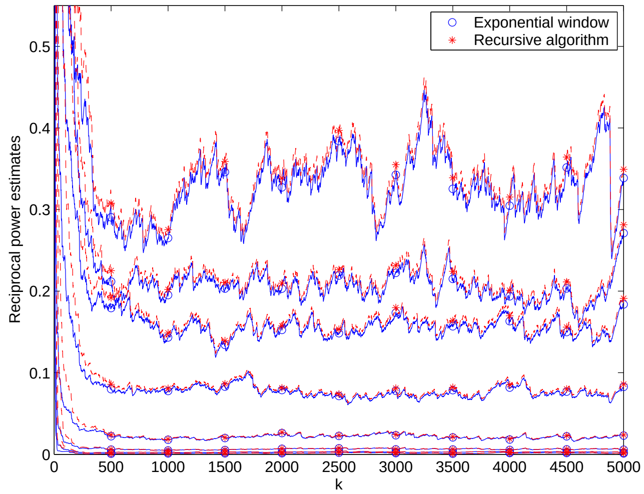

Figure 2: Plot of $1/\sigma_i^2(k)$ and $\phi_i(k)$, $i = 0, \ldots, 7$.

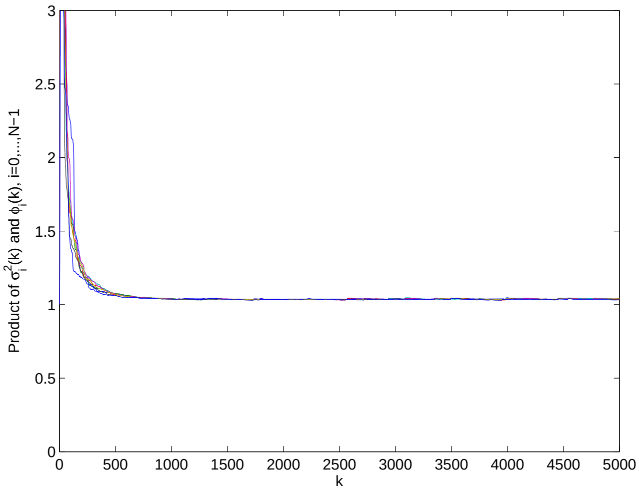

Figure 3: Plot of $\sigma_i^2(k)\phi_i(k)$, $i = 0, \ldots, 7$, versus k ($\lambda = 0.995$). Note that this product should ideally yield a horizontal line at the same level for all i.

than that of a multiply operation. Table 1 includes a complexity comparison between the two power normalization algorithms when implemented on the C62x. For $N = 512$, which is the typical adaptive filter length for a network echo canceller, the range of C62x cycle counts for the exponential window and recursive power normalization algorithms would be 8704–21504 and 1296–1321, respectively. The significant difference between the cycle counts provides a clear indication of the complexity reduction offered by the recursive power normalization algorithm.

4. EQUIVALENCE OF THE POWER NORMALIZATION ALGORITHMS

In this section we compare the reciprocal power estimates produced by the traditional exponential window power estimation algorithm (4) and the reduced-complexity recursive power estimation algorithm derived from the matrix inversion lemma (12). The approximate equivalence of the two power normalization algorithms is shown by way of computer simulations.

In the simulations we use an input signal $x(k)$ with a speech-like spectrum as recommended by USASI (USA Standards Institute) for acoustic echo cancellation applications. An 8-point DCT is used for transforming the time-domain input signal $x(k)$ to the transformed signal $v(k)$ (i.e., $N = 8$). We use a small N in order to facilitate graphical illustration of the power estimates produced by the two algorithms. The exponential window power estimation algorithm was initialized to $\sigma_i^2(0) = 0.02$, $i = 0, \ldots, 7$, and the recursive power estimation algorithm to $\phi_i(0) = 1/\sigma_i^2(0) = 50$, $i = 0, \ldots, 7$. The exponential forgetting factor was set to $\lambda = 0.995$. The resulting reciprocal power estimates, $1/\sigma_i^2(k)$ and $\phi_i(k)$, $i = 0, \ldots, 7$, produced by the two algorithms are shown in Fig. 2. Following an initial transient period, the two estimates settle on approximately the same values. There is a slight difference between the results of the two algorithms because of the assumption made in obtaining (9) from the matrix inversion lemma. To ascertain whether this slight difference is uniform for different transform outputs, we have also plotted the product $\sigma_i^2(k)\phi_i(k)$, $i = 0, \ldots, 7$, versus k in Fig. 3.

In the case of exact equality between the outputs of the two power estimation algorithms, we would have

$$\sigma_i^2(k)\phi_i(k) = 1, \quad 0 \leq i \leq N-1. \tag{13}$$

If the power estimation outputs were simply related to each other by $\phi_i(k) = \alpha/\sigma_i^2(k)$, $i = 0, \ldots, N-1$, where α is a constant, then we would have

$$\sigma_i^2(k)\phi_i(k) = \alpha, \quad 0 \leq i \leq N-1. \tag{14}$$

Equation (14) represents an exact match between the outputs of the two power estimation algorithms up to a scaling factor. Note that for the scaling α to be uniform for each transform output as prescribed by (14), we require the products $\sigma_i^2(k)\phi_i(k)$, $i = 0, \ldots, N-1$ to form identical horizontal lines when plotted against k. Referring to Fig. 3, we observe that the products $\sigma_i^2(k)\phi_i(k)$ form approximately identical horizontal lines with $\alpha \approx 1.05$ as k increases, thereby confirming the approximate equivalence between the two algorithm outputs. We have repeated the previous simulations with the adaptive filter length increased to $N = 32$ and all other parameters remaining unchanged. This has resulted in the product plot shown in Fig. 4. As can be seen from Fig. 4, (14) roughly holds with $\alpha \approx 1.20$ for large k, again confirming the approximate equivalence between the two algorithms. The scaling factor α arises from the approximations made in (7) and (9) in order to reduce the computational complexity. As λ approaches one, the two algorithms produce closer results as can be seen in Figs. 5 and 6.

5. CONCLUSION

The approximate equivalence of the traditional exponential window power estimation algorithm and a recently proposed recursive power normalization algorithm with a single division operation has been demonstrated by way of computer simulations. The cycle count

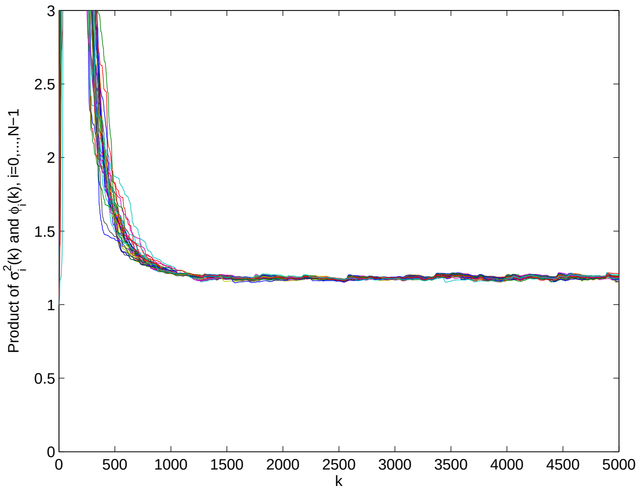

Figure 4: Plot of $\sigma_i^2(k)\phi_i(k)$, $i = 0,\ldots,31$, versus k ($N = 32$, $\lambda = 0.995$).

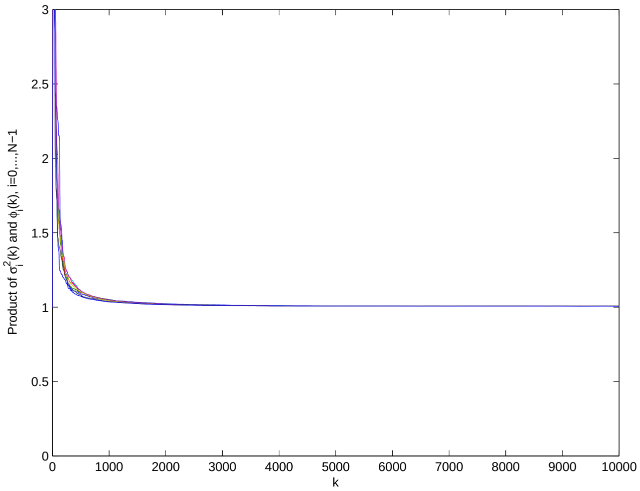

Figure 5: Plot of $\sigma_i^2(k)\phi_i(k)$ ($N = 8$, $\lambda = 0.999$).

saving achieved by the proposed algorithm is significant for DSP implementations [6].

In addition to saving on the computational complexity, the proposed power estimation algorithm can also be used to facilitate further complexity reduction in selective partial updating of the TD-LMS algorithm [6]. It was shown in [6] that selective partial updating of the adaptive filter coefficients leads to a complexity reduction only if the proposed recursive power normalization algorithm is employed. The traditional exponential window power estimator does not permit any complexity reduction for selective partial updating. It results in higher complexity than the full-update TD-LMS algorithm, thereby beating the purpose of partial updating.

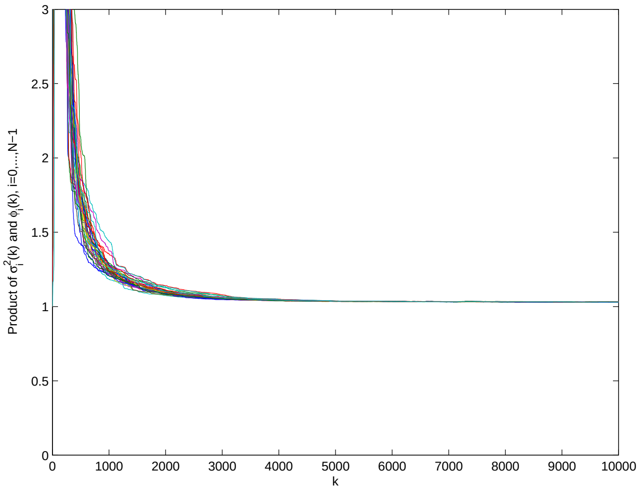

Figure 6: Plot of $\sigma_i^2(k)\phi_i(k)$ ($N = 32$, $\lambda = 0.999$).

REFERENCES

[1] S. Haykin, *Adaptive Filter Theory*, 3rd ed. New Jersey: Prentice Hall, 1996.

[2] W. K. Jenkins, A. W. Hull, J. C. Strait, B. A. Schnaufer, and X. Li, *Advanced Concepts in Adaptive Signal Processing*. Boston: Kluwer, 1996.

[3] M. R. Petraglia and S. K. Mitra, "Adaptive FIR filter structure based on the generalized subband decomposition of FIR filters," *IEEE Trans. on Circuits and Systems II*, vol. 40, no. 6, pp. 354–362, June 1993.

[4] J. J. Shynk, "Frequency-domain and multirate adaptive filtering," *IEEE Signal Processing Magazine*, vol. 9, no. 1, pp. 14–37, January 1992.

[5] K. Doğançay and O. Tanrıkulu, "Adaptive filtering algorithms with selective partial updates," *IEEE Trans. on Circuits and Systems II*, vol. 48, no. 8, pp. 762–769, August 2001.

[6] K. Doğançay, "Complexity considerations for transform-domain adaptive filters," *Signal Processing*, vol. 83, no. 6, pp. 1177–1192, June 2003.

[7] S. S. Narayan, A. M. Peterson, and M. J. Narasimha, "Transform domain LMS algorithm," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 31, no. 3, pp. 609–615, June 1983.

[8] P. K. Bondyopadhyay, "Application of running Hartley transform in adaptive digital filtering," *Proc. IEEE*, vol. 76, no. 10, pp. 1370–1372, October 1988.

[9] B. Farhang-Boroujeny, "Order of N complexity transform domain adaptive filters," *IEEE Trans. on Circuits and Systems II*, vol. 42, no. 7, pp. 478–480, July 1995.

[10] R. Storn, "Echo cancellation techniques for multimedia applications—a survey," International Computer Science Institute, Tech. Rep. TR-96-046, November 1996, ftp.icsi.berkeley.edu.

[11] J. Eyre and J. Bier, "The evolution of DSP processors," *IEEE Signal Processing Magazine*, vol. 17, no. 2, pp. 43–51, March 2000.

[12] Texas Instruments DSP Village website: `http://dspvillage.ti.com`.