Example:

$$T(s) = \cfrac{1}{1 + \cfrac{1}{\frac{1}{3}s + \cfrac{1}{3 + \cfrac{1}{\frac{1}{6}s + \cfrac{1}{12 + \cfrac{1}{H_6 s}}}}}}. \tag{5}$$

In this case

$$\begin{aligned}\begin{bmatrix} B \\ D \end{bmatrix} &= \begin{bmatrix} 1 & \frac{1}{3}s \\ 1 & 1+\frac{1}{3}s \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{6}s \\ 3 & 1+\frac{1}{2}s \end{bmatrix} \begin{bmatrix} H_6 s \\ 1 + 12H_6 s \end{bmatrix} \\ &= \begin{bmatrix} 1 & \frac{1}{3}s \\ 1 & 1+\frac{1}{3}s \end{bmatrix} \begin{bmatrix} (H_6 + \frac{1}{6})s + 2H_6 s^2 \\ 1 + (15H_6 + \frac{1}{2})s + 6H_6 s^2 \end{bmatrix} \\ &= \begin{bmatrix} (H_6 + \frac{1}{2})s + (7H_6 + \frac{1}{6})s^2 + 2H_6 s^3 \\ 1 + (16H_6 + 1)s + (13H_6 + \frac{1}{6})s^2 + 2H_6 s^3 \end{bmatrix}. \end{aligned} \tag{6}$$

Hence, the corresponding rational form is

$$T(s) = \frac{(H_6 + \frac{1}{2})s + (7H_6 + \frac{1}{6})s^2 + 2H_6 s^3}{1 + (16H_6 + 1)s + (13H_6 + \frac{1}{6})s^2 + 2H_6 s^3}. \tag{7}$$

If

$$H_6 = \frac{1}{30} + \frac{G_1(s)}{s} + \frac{1}{sG_2(s)}$$

(7) becomes [1]

$$T(s) = \frac{(8s + 6s^2 + s^3)G_2(s) + 15(1 + 7s + 2s^2)\{1 + G_1(s)G_2(s)\}}{(15 + 23s + 9s^2 + s^3)G_2(s) + 15(16 + 13s + 2s^2)\{1 + G_1(s)G_2(s)\}}. \tag{8}$$

An extension of the above approach to [2] can easily be conceived. A detailed analysis will be presented elsewhere [3].

REFERENCES

[1] L. C. Agarwal, "A matrix method for a general continued fraction inversion," *Proc. IEEE* (Lett.), vol. 64, pp. 1433-1435, Sept. 1976.

[2] C. F. Chen and W. T. Chang, "A matrix method for continued fraction inversion," *Proc. IEEE* (Lett.), vol. 62, pp. 636-637, May 1974.

[3] Alok Kumar, V. Singh, and C. P. Gupta, "On continued fraction inversion," scheduled for presentation at the National Systems Conf. (NSC-78), Ludhiana, India, Sept. 4-6, 1978.

Adaptive Filtering in the Frequency Domain

MAURO DENTINO, JOHN McCOOL, AND BERNARD WIDROW

Abstract—**Adaptive filtering in the frequency domain can be accomplished by Fourier transformation of the input signal and independent weighting of the contents of each frequency bin. The frequency-domain filter performs similarly to a conventional adaptive transversal filter but promises a significant reduction in computation when the number of weights equals or exceeds 16.**

M. Dentino is with the Marine Systems Division, Rockwell International Corporation, Anaheim, CA 92803.

J. McCool is with the Fleet Engineering Department, Naval Ocean Systems Center, San Diego, CA 92152.

B. Widrow is with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305.

Adaptive filters are used in a wide variety of applications, including statistical prediction, interference canceling, array phasing, and channel equalization in communication systems [1]-[4]. In the normal implementation the outputs of a tapped delay line are weighted and summed under control of a recursive algorithm to form the filter output. The most commonly used algorithm at the present time is the Widrow-Hoff least-mean-square (LMS) algorithm [5]. The purpose of this note is to describe a new method of adaptive filtering, based on a complex form of the LMS algorithm and performed in the frequency rather than the time domain. This method promises great improvements in computational efficiency when the number of adaptive weights is large.

A conventional time-domain adaptive filter is shown in Fig. 1. The discrete input signal is represented by x_j, the output signal by y_j, and the "desired response" by d_j. The latter is a training signal necessary to effect the adaptive process and whose derivation generally requires some ingenuity. The LMS algorithm is represented by

$$W_{j+1} = W_j + 2\mu\epsilon_j X_j \tag{1}$$

where W_j and X_j are vectors representing, respectively, the filter weights and the outputs of the tapped delay line at time j:

$$W_j^T = [w_{1j}, w_{2j}, \cdots, w_{nj}] \tag{2}$$

$$X_j^T = [x_j, x_{j-1}, \cdots, x_{j-n+1}]. \tag{3}$$

The error ϵ_j is the difference between the desired response and the filter output

$$\epsilon_j = d_j - y_j \tag{4}$$

where the filter output is given by

$$y_j = X_j^T W_j. \tag{5}$$

The term μ in (1) is a constant that governs rate of convergence and whose proper choice ensures stability of the adaptive process. The weights are updated at the input sampling rate.

The frequency-domain LMS adaptive filter, illustrated in Fig. 2, is similar in general configuration to the conventional time-domain filter. The input signal x_j and desired response d_j, however, are accumulated in buffer memories (not shown in Fig. 2) to form n-point data blocks. They are then transformed by n-point FFT's. Each of the FFT outputs comprises a set of n complex numbers. The desired response transform values are subtracted from the input transform values at corresponding frequencies to form n complex error signals. There are n complex weights, one corresponding to each spectral bin. Each weight is independently updated once for each data block. The weighted outputs are not summed but are fed to an inverse FFT operator to produce the output signal y_{j-n}, delayed by the number of samples n in the input data block.

The complex LMS algorithm is given in [6] as

$$W_{j+1} = W_j + 2\mu\epsilon_j \overline{X}_j \tag{6}$$

where the overbar denotes complex conjugate. This algorithm was devised for the filtering of complex signals in the time domain but can be applied in the frequency domain by providing each complex weight with an individual error signal. The lth complex weight for the lth frequency is then updated according to

$$w_l(k+1) = w_l(k) + 2\mu\epsilon_l(k)\,\overline{x}_l(k) \tag{7}$$

where $w_l(k+1)$ is the lth complex weight after adaptation with the kth n-point input data block. Since each weight is adapted only once for each n-point data block, the number of adaptations required to obtain output data similar to those obtained with the conventional time-domain filter is reduced by a factor of n. The value of the adaptive constant μ may accordingly be increased by a factor of n. The larger value of μ, corresponding to less frequent adaptation, permits a lowering of the weight resolution requirements for the frequency-domain filter by a factor of n, so that the number of bits required to store each weight can be reduced by $\log_2 n$, simplifying the weight update arithmetic.

The difference in computational requirements of the conventional and frequency-domain LMS adaptive filters can perhaps best be demonstrated in terms of number of multiply operations required to produce a given amount of output data. To produce n output data points with

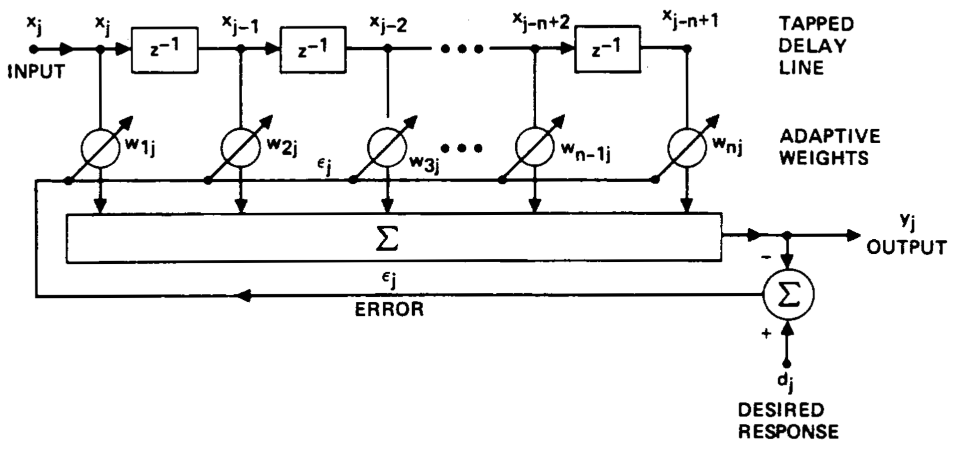

Fig. 1. Conventional LMS adaptive filtering.

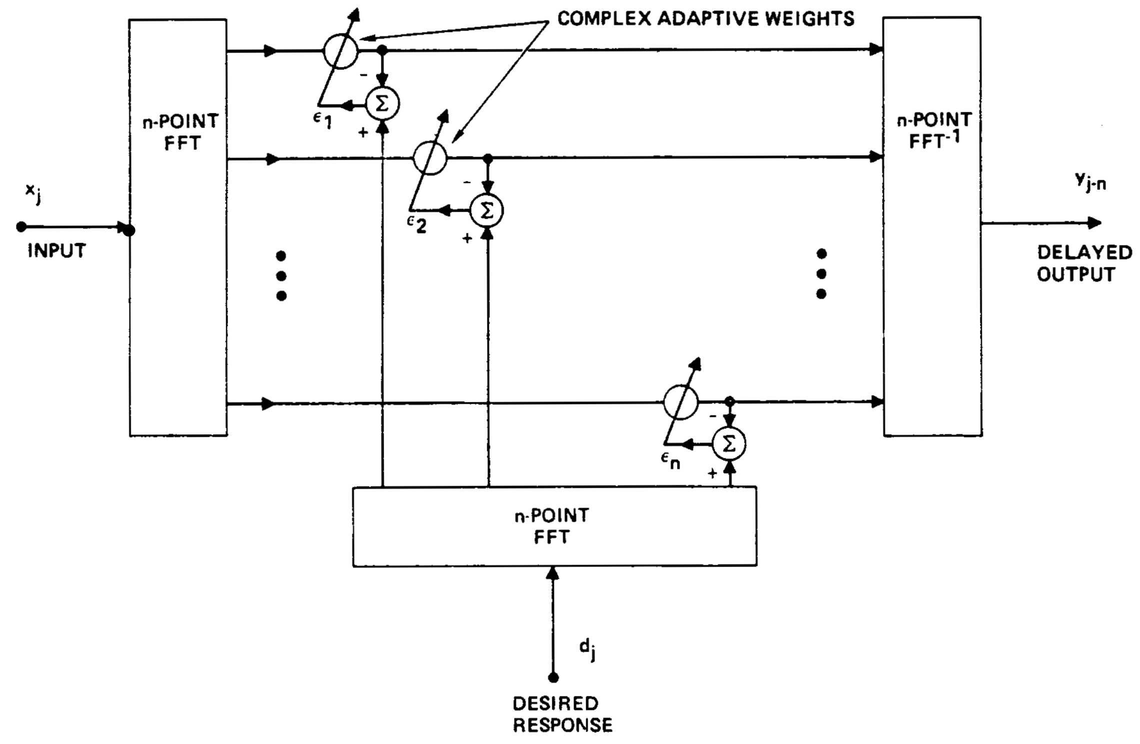

Fig. 2. LMS adaptive filtering in the frequency domain.

the conventional filter requires n^2 adaptations and $2n^2$ real multiplies. To produce the same output with the frequency-domain filter requires $(3n/2) \log_2 n$ complex multiplies for the three n-point FFT's and $2n$ complex multiplies for the complex weighting and weight updating. The ratio of complex multiplies required by the frequency-domain filter to real multiplies required by the conventional filter is thus

$$\frac{\text{complex multiplies}}{\text{real multiplies}} = \frac{(3n/2)\log_2 n + 2n}{2n^2} = \frac{3\log_2 n + 4}{4n}. \qquad (8)$$

With $n = 4$, this ratio is 0.833; with $n = 16$ it is 0.250; with $n = 128$ it is 0.0469; and with $n = 1024$ it is 0.0083. With $n = 4$ there is thus no reduction in computational requirements, while with $n = 16$ the potential reduction factor is approximately 4, and with $n = 1024$ it is greater than two orders of magnitude.

It is apparent that in many practical cases the savings resulting from use of the frequency-domain technique are substantial, so much so that even when one takes into account the additional computing of three FFT's per data block, it pays to use this technique and to recover the filtered signal from its transform. Using conventional digital techniques, the output can be accumulated in a buffer memory and "stitched together," creating a continuous though delayed flow of output data. In this way any large adaptive digital filter can be efficiently realized by a combination of the complex LMS and FFT algorithms. If one requires only the transform of the output or its power spectrum, the third FFT computation can be eliminated.

References

[1] J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, pp. 561-580, Apr. 1975.
[2] B. Widrow *et al.*, "Adaptive noise cancelling: Principles and applications," *Proc. IEEE*, vol. 63, pp. 1692-1716, Dec. 1975.
[3] *IEEE Trans. Antennas Propagat.*, vol. AP-24, Sept. 1976, Special Issue on Adaptive Antennas.
[4] R. Lucky *et al.*, *Principles of Data Communication*. New York: McGraw-Hill, 1968.
[5] B. Widrow *et al.*, "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proc. IEEE*, vol. 64, pp. 1151-1162, Aug. 1976.
[6] B. Widrow, J. McCool, and M. Ball, "The complex LMS algorithm," *Proc. IEEE (Lett.)*, vol. 63, pp. 719-720, Apr. 1975.