A POLYPHASE MODEL FOR FAST AFFINE PROJECTION WITH PARTIAL FILTER UPDATE

Edward Chau, Hamid Sheikhzadeh, Robert L. Brennan

Dspfactory Ltd., 611 Kumpf Drive, Unit 200, Waterloo Ontario, Canada N2V 1K8
E-mail addresses: {echau, hsheikh, rbrennan}@dspfactory.com

ABSTRACT

In this paper, we propose a framework for analysing the convergence behavior of an affine projection based algorithm, using a polyphase filterbank model of the adaptive filter. We have recently introduced a low computation-cost version of the Fast Affine Projection Algorithm (FAPA) that combines FAPA with Partial Filter Update (PFU-FAPA). It is shown in this paper that the polyphase model of PFU-FAPA does not represent a perfect reconstruction system. This is in contrast to the sequential update LMS, which is based on a similar partial filter update method, and can be modeled by a perfect reconstruction polyphase filterbank. Hence, in PFU-FAPA, as the decimation rate of the partial update algorithm increases, distortion of the output signal increases. However, as the polyphase model shows, the distortion is only due to the effect of partial update on the autocorrelation estimation, and is negligible at low or modest decimation rates.

1. INTRODUCTION

The popular Normalized Least Square algorithm (NLMS) offers a simple and effective method for adaptive noise and echo cancellation. However, because of the large eigenvalue spread for colored input signals (such as the subband signals in oversampled subband adaptive filters), the convergence of the NLMS algorithm can be very slow. Thus, the Affine Projection Algorithm (APA) is employed in [1] as a superior adaptation technique for providing better convergence behavior than the NLMS algorithm, while at the same time avoiding the high computation cost and instability associated with the Recursive Least Squares (RLS) algorithm. Nevertheless, for real-time implementation on very low resource platforms, only very low affine projection orders (2 or 3) are generally practical.

Recently, fast versions of APA (FAPA) have been introduced [2, 3] that approximate the original APA without considerable performance loss. To be able to employ FAPA on low-resource platforms, we recently proposed to further reduce the complexity of FAPA by employing partial filter updates in FAPA [4]. The new algorithm was evaluated in a subband acoustic echo canceller application.

In this paper, motivated by the research presented in [5], a polyphase filterbank representation is formulated for the Partial Filter Update FAPA (PFU-FAPA) initially proposed in [4]. In [6], polyphase models based on delay-chain perfect reconstruction (PR) filterbanks are presented for different classes of partial update LMS algorithm. In this research we expand the models to include PFU-FAPA. We show that the polyphase filterbank model of the "transformed" version of the adaptive filter in PFU-FAPA (the "fast" adaptive coefficient vector in [2]) constitutes a PR filterbank. However, we also show that the polyphase filterbank model of the autocorrelation estimation part in PFU-FAPA is not a PR system. This justifies the results already reported in [4].

This paper is organized as follows. Section 2 describes the PFU-FAPA. In Section 3 a polyphase model for the PFU-FAPA is proposed, and discussions are presented in Section 4. Throughout this paper, L denotes the adaptive filter length, N denotes the affine order, and D denotes the decimation rate of the partial filter update. Mathematical notations and definitions are based heavily on [2].

2. COMBINATION OF FAPA AND PARTIAL UPDATE ALGORITHM

The time-domain FAPA and PFU-FAPA are summarized as follows. Generalization for complex subbands (e.g. in oversampled subband adaptive filters) is straight-forward, and is omitted here for simplicity.

2.1 FAPA

The following description of FAPA is based on [2] and [3], where the inversion of the autocorrelation matrix is formulated as solving a system of equations (see Eq. (5) below).

Initialization: assume, for $n<0$, $x_n=y_n=0$, $\underline{\hat{h}}_n=\underline{0}^t$, $\underline{E}_n=\underline{0}^t$, $\underline{\hat{\alpha}}_n=\underline{0}^t$, $\mathbf{R}_n=\mathbf{X}_n^t\mathbf{X}_n=\mathbf{0}+\delta\mathbf{I}$, $\underline{\hat{r}}_{xx,n}=[\delta,\underline{0}]^t$, $0.7<\mu<1$, and $\underline{P}_n=\underline{b}/\delta$.

Then, at each sample $n\geq 0$:

$$\underline{\hat{r}}_{xx,n} = \underline{\hat{r}}_{xx,n-1}+x_n\underline{\hat{\alpha}}_n-x_{n-L}\underline{\hat{\alpha}}_{n-L} \quad (1)$$

$$\hat{e}_n = y_n-\underline{x}_n^t\underline{\hat{h}}_{n-1} \quad (2)$$

$$e_n = \hat{e}_n-\mu\underline{\tilde{r}}_{xx,n}^t\overline{\underline{E}}_{n-1} \quad (3)$$

$$\text{update } \mathbf{R}_n \text{ using } \underline{\hat{r}}_{xx,n} \quad (4)$$

$$\text{solve } \mathbf{R}_n\underline{P}_n = \underline{b} \text{ for } \underline{P}_n \quad (5)$$

$$\underline{\varepsilon}_n = e_n\underline{P}_n \quad (6)$$

$$\underline{E}_n = \begin{bmatrix} 0 \\ \overline{\underline{E}}_{n-1} \end{bmatrix}+\underline{\varepsilon}_n \quad (7)$$

$$\underline{\hat{h}}_n = \underline{\hat{h}}_{n-1}+\mu\underline{x}_{n-(N-1)}E_{N-1,n} \quad (8)$$

In the above, $\underline{\hat{h}}_n$ is the $L\times 1$ "transformed" adaptive filter coefficients, $\underline{E}_n$ is a $N\times 1$ vector consisting of a sum of the fast normalized residual echo $\underline{\varepsilon}_n$ [2], $E_{N-1,n}$ is the last element of $\underline{E}_n$, $\overline{\underline{E}}_n$ is a vector consisting of the uppermost $N-1$ elements of $\underline{E}_n$, $\underline{x}_n$ is the $L\times 1$ excitation signal vector, y_n is the reference signal, e_n is the error signal, $\underline{\hat{\alpha}}_n=[x_n,\ldots,x_{n-N+1}]^t$, $\underline{b}=[1,\underline{0}]^t$, $\mathbf{X}_n$ is the $L\times N$ excitation signal matrix, $\mathbf{I}$ is the identity matrix, and δ is the regularization factor. Note, $\underline{\tilde{r}}_{xx,n}$ in Eq. (3) has the same definition as in [2], and is simply the $N-1$ lower elements of $\underline{\hat{r}}_{xx,n}$. Also, in Eq. (4), $\mathbf{R}_n$ is updated simply by replacing the first row and column with the elements of $\underline{\hat{r}}_{xx,n}$, and the bottom $(N-1)\times(N-1)$ submatrix is replaced with the top $(N-1)\times(N-1)$ submatrix of $\mathbf{R}_{n-1}$. The system of equations in Eq. (5) can be solved efficiently by performing one Gauss-Seidel (GS) iteration per each iteration of the FAPA as suggested in [3].

2.2 The PFU-FAPA Algorithm

The Partial Filter Update (PFU) method [4] aims to reduce the computation requirement of FAPA without sacrificing too much of the system performance. It is similar in concept to the Sequential Least Mean Square (S-LMS) algorithm in [7], and is implemented as follows. Let D be a positive integer, $D\geq 1$ and $L \bmod D=0$. Then, the updating of transformed filter coefficients $\underline{\hat{h}}_n$ (Eq. (8)) is modified

to

$$\hat{\underline{h}}_n \quad = \quad \hat{\underline{h}}_{n-1} + \mu \tilde{\underline{x}}_{n-(N-1)} E_{N-1,n} \tag{9}$$

where

$$\tilde{\underline{x}}_n = \left[\tilde{x}_n, \tilde{x}_{n-1}, \ldots, \tilde{x}_{n-L+1}\right]^t$$
$$\tilde{x}_n = \begin{cases} x_n & \text{if } n \bmod D = 0 \\ 0 & \text{otherwise} \end{cases}$$

Also, $\hat{\underline{\alpha}}_n$ in Eq. (1) is replaced with $\breve{\underline{\alpha}}_n = \left[\tilde{x}_n, \ldots, \tilde{x}_{n-N+1}\right]^t$. As a result, when $D > 1$, only every D-th element of $\hat{\underline{h}}_n$ and $\hat{\underline{r}}_{xx,n}$ are updated at a time. $\mathbf{R}_n$ is then updated at every D-th frame (i.e. executing Eq. (4) only when $n \bmod D = 0$), which is also when $\hat{\underline{r}}_{xx,n}$ is fully updated. When $D = 1$, the algorithm reverts to the original FAPA method.

3. POLYPHASE MODEL FOR PFU-FAPA USING A DELAY CHAIN FILTERBANK

In [6], it is shown that the PFU method in the S-LMS algorithm (described in [7]) can be seen as a polyphase implementation of LMS where only one polyphase component is updated at a time. It can be seen clearly that a similar polyphase analysis will also apply here for PFU-FAPA. However, one significant difference here is that the partial update is applied separately to two different parts of the adaptive filter – i.e. the transformed filter coefficients $\hat{\underline{h}}_n$ and the estimated autocorrelation vector $\tilde{\underline{r}}_{xx,n}$. In contrast, the partial update in S-LMS is applied directly to the adaptive filter ($\underline{h}_n$) itself. Hence, we intend to show in the following that the partial update method in PFU-FAPA is not modelled by a Perfect Reconstruction (PR) filterbank as noted in [6] for S-LMS. Also, note that the model derived here is general enough not to rely on the particular method used in Eq. (5) or finding the regularization factor δ, so long as the method does not alter the algorithm as described in Section 2.

To derive the polyphase model, let

$$\mathbf{A}_n = \begin{bmatrix} \tilde{\underline{\alpha}}_n^t \\ \tilde{\underline{\alpha}}_{n-1}^t \\ \vdots \\ \tilde{\underline{\alpha}}_{n-L+2}^t \end{bmatrix}$$

where $\tilde{\underline{\alpha}}_n = \left[\tilde{x}_{n-1}, \ldots, \tilde{x}_{n-N+1}\right]^t$ (the lower $N-1$ elements of $\breve{\underline{\alpha}}_n$), i.e.

$$\mathbf{A}_n = \begin{bmatrix} \tilde{x}_{n-1} & \tilde{x}_{n-2} & \cdots & \tilde{x}_{n-N+1} \\ \tilde{x}_{n-2} & \tilde{x}_{n-3} & \cdots & \tilde{x}_{n-N} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{x}_{n-L} & \tilde{x}_{n-L-1} & \cdots & \tilde{x}_{n-L-N+2} \end{bmatrix}$$

Then, it can be easily seen that

$$\tilde{\underline{r}}_{xx,n} = \underline{x}_n^t \mathbf{A}_n$$

and, assuming $\mu = 1$ from now on for simplicity,

$$e_n = y_n - \underline{x}_n^t \left(\hat{\underline{h}}_{n-1} + \mathbf{A}_n \overline{\underline{E}}_{n-1}\right) \tag{10}$$

The adaptive filter as described by Eq. (10) is illustrated in Figure 1. Note that the matrix $\mathbf{A}_n$ is "adaptive" (as shown with the dashed arrow in the figure) only in the sense that it contains the *time-varying* elements of the input signal x_n.

Next, we separate $\underline{x}_n^t \mathbf{A}_n \overline{\underline{E}}_{n-1}$ into $N-1$ "filters" (i.e. "filtering" of $\underline{x}_n^t$ with $\mathbf{A}_n$) and a sum of $N-1$ multiplications (i.e. vector multiplication of $(\underline{x}_n^t \mathbf{A}_n)$ with $\overline{\underline{E}}_{n-1}$). In other words, we treat each column of $\mathbf{A}_n$ independently, so that

$$\underline{x}_n^t \mathbf{A}_n = \underline{x}_n^t \left[\underline{A}_{n,0}, \underline{A}_{n,1}, \ldots, \underline{A}_{n,N-2} \right]$$

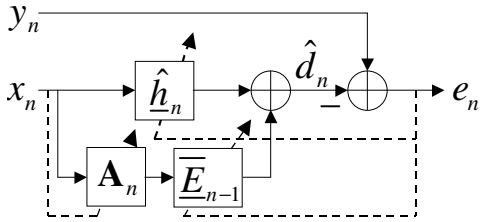

Figure 1: Adaptive Filtering in PFU-FAPA

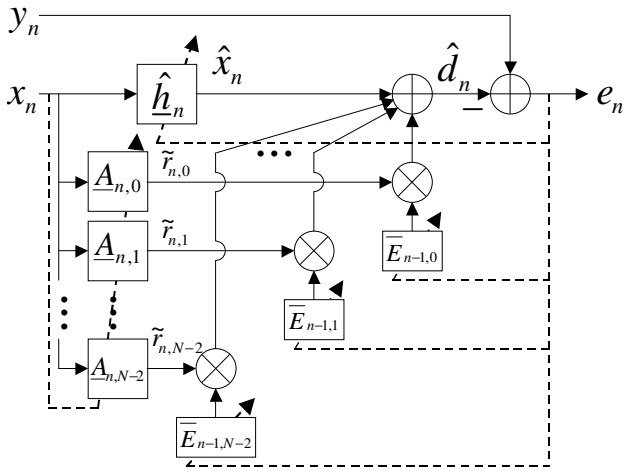

Figure 2: Separating $\mathbf{A}_n$ into $N-1$ filters

Figure 2 shows the adaptive filter with $\mathbf{A}_n$ treated as $N-1$ separate filters. The output of each of the $N-1$ filters, $\underline{A}_{n,i}$, is an estimation of the autocorrelation at time n, $\tilde{r}_{n,i}$, with $(i+1)$ lag and $0 \le i \le N-2$. Note, $\overline{E}_{n-1,i}$ denotes the i-th element of $\overline{\underline{E}}_{n-1}$.

For $D = 1$, Figure 1 or Figure 2 will be sufficient in describing the adaptive filter. However, for $D > 1$, we will take the polyphase approach to describe the partial filter update method in PFU-FAPA. The polyphase model derived for the S-LMS method in [6] can be applied equivalently to the transformed adaptive filter $\hat{\underline{h}}_n$ in PFU-FAPA. This is described later in this section. First, for PFU-FAPA we need to additionally derive the polyphase model for the autocorrelation vector $\tilde{\underline{r}}_{xx,n}$. In the descriptions to follow, we assume $D = 2$. Extension to $D > 2$ is straight-forward.

For example, assume $L = N = 4$ for simplicity (in practice $L >> N$). At $n = 0$, we have

$$\mathbf{A}_0 = \begin{bmatrix} \tilde{x}_{-1} & \tilde{x}_{-2} & \tilde{x}_{-3} \\ \tilde{x}_{-2} & \tilde{x}_{-3} & \tilde{x}_{-4} \\ \tilde{x}_{-3} & \tilde{x}_{-4} & \tilde{x}_{-5} \\ \tilde{x}_{-4} & \tilde{x}_{-5} & \tilde{x}_{-6} \end{bmatrix} = \begin{bmatrix} 0 & x_{-2} & 0 \\ x_{-2} & 0 & x_{-4} \\ 0 & x_{-4} & 0 \\ x_{-4} & 0 & x_{-6} \end{bmatrix}$$

Then for the autocorrelation vector we have $\tilde{\underline{r}}_{xx,0} = \underline{x}_0^t \mathbf{A}_0$, thus,

$$\begin{aligned} \tilde{\underline{r}}_{xx,0} \quad &= \quad \left[\begin{array}{cccc} x_0 & x_{-1} & x_{-2} & x_{-3} \end{array} \right] \mathbf{A}_0 \\ &= \quad \left[\begin{array}{ccc} (x_{-1}x_{-2} + x_{-3}x_{-4}) & (x_0 x_{-2} + x_{-2}x_{-4}) & \cdots \\ & (x_{-1}x_{-4} + x_{-3}x_{-6}) & \end{array} \right] \end{aligned}$$

Next, at $n = 1$, we have $\tilde{\underline{r}}_{xx,1} = \underline{x}_1^t \mathbf{A}_1$, where

$$\mathbf{A}_1 = \begin{bmatrix} x_0 & 0 & x_{-2} \\ 0 & x_{-2} & 0 \\ x_{-2} & 0 & x_{-4} \\ 0 & x_{-4} & 0 \end{bmatrix}$$

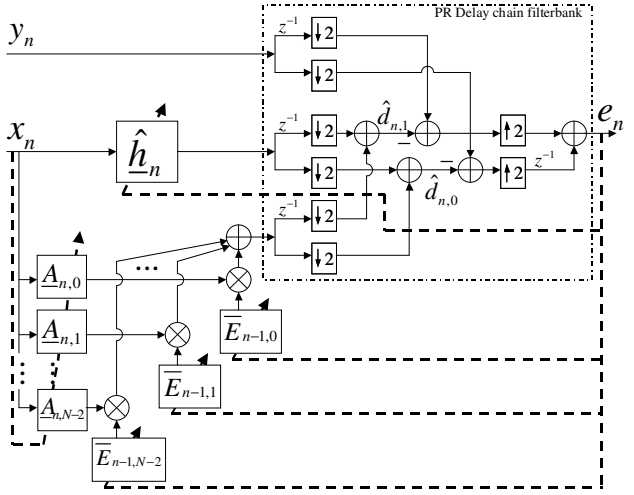

Figure 3: Cascading the Adaptive Filter with a PR Delay Chain Filterbank - Step I

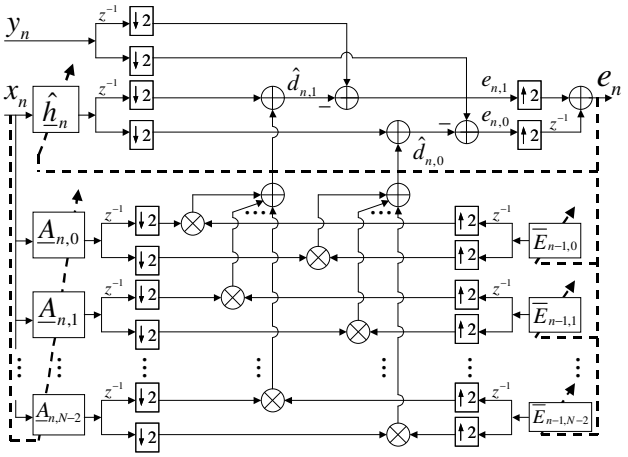

Figure 4: Cascading the Adaptive Filter with a PR Delay Chain Filterbank - Step II

Thus,

$$\tilde{\underline{r}}_{xx,1} = \left[\; (x_1x_0 + x_{-1}x_{-2}) \quad (x_0x_{-2} + x_{-2}x_{-4}) \quad \ldots \right.$$
$$\left. (x_1x_{-2} + x_{-1}x_{-4}) \; \right]$$

Similarly, at $n = 2$, $\tilde{\underline{r}}_{xx,2} = \underline{x}_2^t \mathbf{A}_2$, so we have

$$\tilde{\underline{r}}_{xx,2} = \left[\; (x_1x_0 + x_{-1}x_{-2}) \quad (x_2x_0 + x_0x_{-2}) \quad \ldots \right.$$
$$\left. (x_1x_{-2} + x_{-1}x_{-4}) \; \right]$$

As shown in Figure 2, we can treat each element of $\tilde{\underline{r}}_{xx,n}$ independently (i.e. $\tilde{r}_{n,i}$, $0 \le i \le N-2$) as the output of the filters $\underline{A}_{n,i}$. Then, it can be seen from the above example that the effect of the PFU algorithm on the autocorrelation vector is determined by the modification to the individual filters $\underline{A}_{n,i}$. In particular, there are two modifications to $\underline{A}_{n,i}$ due to partial update – first the use of a decimated version of x_n, and second the update of the "filter taps" in each $\underline{A}_{n,i}$. These two modifications are key to the construction of the polyphase model as described in the following.

For the polyphase model, we first cascade the adaptive filter with a PR delay chain filterbank as similarly done in [6]. This is illustrated in Figure 3. Next, we move the multiply-add operations with $\overline{E}_{n-1,i}$ to the subband by swapping the decimation-by-two operation with the multiply-add operation, and adding the appropriate expansion-by-two operation, as shown in Figure 4.

Then, the equivalent polyphase model of the adaptive filter is shown in Figure 5. The figure depicts the adaptive filter without PFU, and all the polyphase components are being jointly updated. Since $D = 2$ in our example, there are only two polyphase components shown in the figure.

When the PFU method as described in Section 2 is employed, the modifications to the polyphase model are shown in Figure 6 and Figure 7. Figure 6 shows the polyphase model when the time index, n, is even, and Figure 7 shows the corresponding model when n is odd. Essentially, the transformed filter coefficients $\hat{\underline{h}}_n$ are partially updated in a similar way as in S-LMS – i.e. while all the polyphase components of $\hat{\underline{h}}_n$ are used for filtering, only one of them is updated at a time (see Eq. (9)), as shown with the dashed arrows in the figures.

On the other hand, the modification to $\underline{A}_{n,i}$ due to partial update has two effects on the polyphase model, as shown in the above example. The first is the decimation effect by a factor of D, which means only one polyphase component of $\underline{A}_{n,i}$ is used at a time for filtering. In the figures, the polyphase components that are not used at time n (n even or odd) are shaded out. The second effect is due to the updating of the elements of $\underline{A}_{n,i}$, where the zero coefficients are not confined to specific "filter taps" of $\underline{A}_{n,i}$, but change positions depending on the value of $\tilde{x}_n$. This means that different polyphase components of $\underline{A}_{n,i}$ are updated and used for filtering at different times. Specifically, as shown in Figure 6, when n is even, only the first polyphase component of $\underline{A}_{n,i}$ (i.e. $\underline{A}_{n,i,0}$) is updated and used for filtering in the even polyphase branch, while in the odd polyphase branch, only the second polyphase component (i.e. $\underline{A}_{n,i,1}$) is used for filtering (but not updated). Conversely, when n is odd, only the second polyphase component is updated and used for filtering in the even polyphase branch, while only the first polyphase component (which has the same values as in $n-1$) is used for filtering in the odd polyphase branch, as shown in Figure 7. Similarly, for $D > 2$, in the even polyphase branch, the polyphase components of $\underline{A}_{n,i}$ are used for filtering one at a time in ascending order (i.e. starting with the first polyphase component at $n = 0$, then the second component at $n = 1$, the third component at $n = 2$ ... etc. until $n \bmod D = 0$, when the first component is used again, and so on), while in the odd polyphase branch, the polyphase components of $\underline{A}_{n,i}$ are used for filtering one at a time in descending order. At each time n, only the polyphase component that is being used for filtering in the even polyphase branch is updated. Thus this completes the polyphase model for PFU-FAPA.

4. DISCUSSION

In this work, a polyphase filterbank model of PFU-FAPA is derived and presented. As shown clearly in Figure 6 and Figure 7, a major difference between PFU-FAPA and S-LMS is the effect of the partial update method on part of the adaptive filter – in particular, the estimated autocorrelation vector $\tilde{\underline{r}}_{xx,n}$. Unlike the transformed filter $\hat{\underline{h}}_n$, the partial update method does not have a PR effect on $\tilde{\underline{r}}_{xx,n}$ because only one polyphase component of $\underline{A}_{n,i}$ is used for filtering at a time. However, it is also different from a simple decimation of $\underline{A}_{n,i}$ because, over time, none of its polyphase component is ignored (i.e. for $n > D$, every component will have been updated and used for filtering at least once). Nevertheless, the estimation of the autocorrelation vector will be worse as the value of D increases – because of first the decimation effect at each n (i.e. only one polyphase component is used for filtering at a time), and second the slower update rate for $\underline{A}_{n,i}$ (i.e. only one polyphase component is updated at a time), especially when non-stationary signals are used.

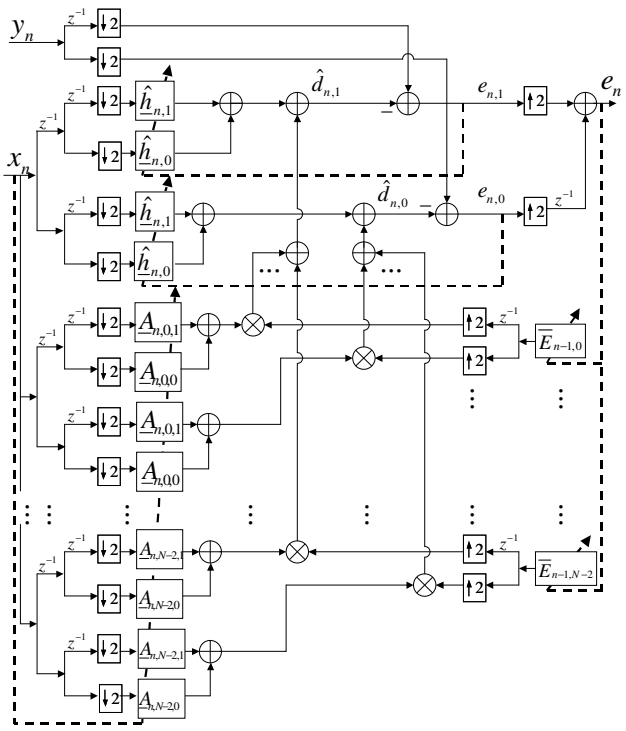

Figure 5: Polyphase Representation of Adaptive Filter using a PR Delay Chain Filterbank - Without Partial Filter Update

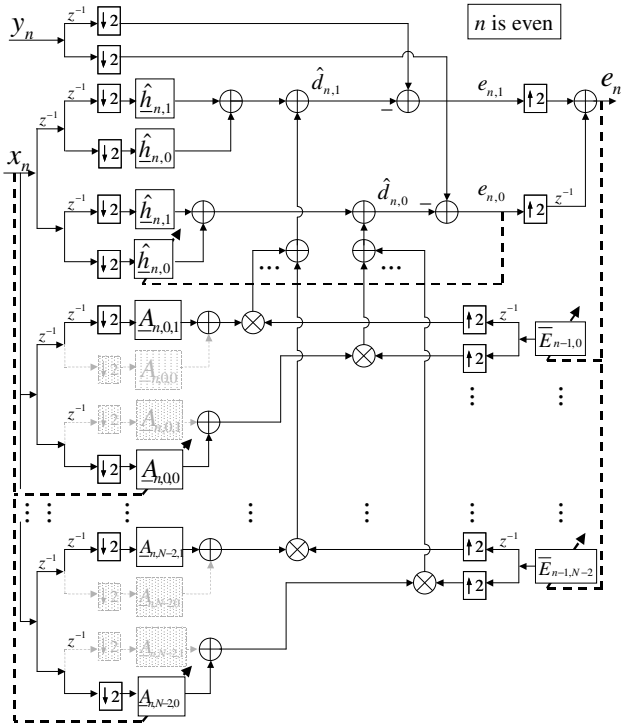

Figure 6: Polyphase Representation of Adaptive Filter using a PR Delay Chain Filterbank - With Partial Filter Update and n is even (shaded components are not used for filtering)

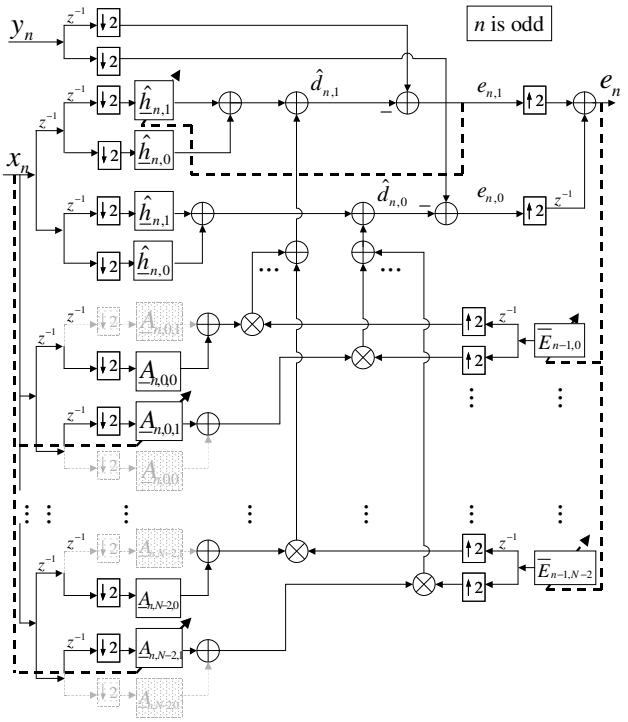

Figure 7: Polyphase Representation of Adaptive Filter using a PR Delay Chain Filterbank - With Partial Filter Update and n is odd (shaded components are not used for filtering)

This is consistent with the results presented in [4] for PFU-FAPA. We expect this polyphase approach should provide the groundwork for further analysis of the relationship between D and the filter performance in PFU-FAPA or other similar partial update methods for adaptive filters.

REFERENCES

[1] H. R. Abutabeli, H. Sheikhzadeh, R. L. Brennan, and G. H. Freeman, "Affine projection algorithm for oversampled subband adaptive filters," *Proc. IEEE Int. Conf. Acous., Speech, and Sig. Proc.*, pp. VI 209–212, 2003.

[2] S. L. Gay and S. Tavathia, "The fast affine projection algorithm," *Proc. IEEE Int. Conf. Acous., Speech, and Sig. Proc.*, pp. 3023–3026, 1995.

[3] F. Albu, J. Kadlec, N. Coleman, and A. Fagan, "The gauss-seidel fast affine projection algorithm," *IEEE Workshop SIPS*, pp. 109–114, 2002.

[4] E. Chau, H. Sheikhzadeh, and R. L. Brennan, "Complexity reduction and regularization of a fast affine projection algorithm for oversampled subband adaptive filters," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 2004.

[5] S. S. Pradhan and V. U. Reddy, "A new approach to subband adaptive filtering," *IEEE Trans. Signal Processing*, vol. 47, no. 3, pp. 655–664, 1999.

[6] H. Sheikhzadeh, H. R. Abutabeli, R. L. Brennan, K. R. L. Whyte, and E. Chau, "Sequential LMS for low-resource subband adaptive filtering: Oversampled implementation and polyphase analysis," *submitted to EUSIPCO 2004*.

[7] S. C. Douglas, "Adaptive filters employing partial updates," *IEEE Trans. Circuits and Systems - II*, vol. 44, pp. 209–216, 1997.