Convergence Enhancement of the GMDFα Algorithm

Karim Maouche† and Dirk T.M. Slock ††

Institut EURECOM, 2229 route des Crêtes
B.P. 193, 06904, Sophia Antipolis Cedex, FRANCE
† *Tel/Fax: +33 493002632/ 493002627, E-Mail: maouche@eurecom.fr*
†† *Tel/Fax: +33 493002606/ 493002627, E-Mail: slock@eurecom.fr*

Abstract. In this paper, we derive a new normalisation technique for Frequency Domain Adaptive Filtering (FDAF) algorithms. FDAF algorithms are well-known for their low computational complexity and for their decorrelating property that allows the use of different step sizes for each adaptive weight, yielding a uniform convergence of all the modes of the input signal. In these algorithms, normalisation is done by recursively estimating the power of each frequency bin. By introducing a normalisation based on an orthogonal projection (as is the case for the Affine Projection algorithm), we derive new frequency domain adaptive filtering algorithms and show by means of simulation that the convergence speed is improved.

1 Introduction

Because of their important reduction in computational complexity, Frequency Domain Adaptive Filter (FDAF) algorithms [Shy92] are very attractive. This fact may be of great importance for real-time applications where long Finite Impulse Response (FIR) filters are needed as in the case of Acoustic Echo Cancellation (AEC). FDAF algorithms also have the advantage of working in the spectral domain where the signals are approximately decorrelated. In fact, this property allows the use of different step sizes for each adaptive weight yielding an approximately uniform convergence of all the modes of the input signal. In basic FDAF algorithms, the adaptive filter of length N is updated every N input signal samples and FFTs of order $2N$ (N in the unconstrained case) are performed. Hence, the processing delay which is equal to the size block and the length of FFTs can be prohibitive in AEC applications. In order to reduce the delay inherent to the block strategy updating and to work with FFTs of reasonable lengths, the impulse response is segmented in small blocks. This was the key ingredient in the derivation of the Multidelay Filter (MDF) algorithms [SP90]. Moreover, by using Weighted Overlap and Add (WOLA) reconstruction technique, the GMDFα algorithm [MtAG95] was derived and showed enhanced initial convergence and tracking behaviour.

In these algorithms, normalisation is done by recursively estimating the power of each frequency bin. The aim of our work is to show that the convergence speed of the MDF and the GMDFα algorithms can be enhanced when a convenient normalisation of the gradient is introduced. The main idea is based on the introduction of a pseudo-inverse matrix in the adaptive algorithm which leads to a projection at each update of the deviation filter onto a particular spectral subspace. We have applied this idea to the well-known FDAF algorithms based on Overlap-Add method (FDAF-OLA) and Overlap-Save (OLS) method. In the two algorithms, our normalisation technique leads to two new frequency domain adaptive algorithms that are the FDAF-Projection OLA (FDAF-POLA) and FDAF-POLS algorithms. After the derivation of these two algorithms, we show how to apply the new normalisation scheme to the MDF algorithms and then to the GMDFα algorithm. Simulations done in various situations prove that the new normalisation scheme improves the convergence. Moreover, this normalisation allows an easier tuning of the adaptive algorithm parameters when compared with the classical normalisation scheme where convergence speed is very sensitive to the choice of the forgetting factor of the exponential window (or the length of the rectangular window) and the initial value of the power. In fact, the new algorithms have only one parameter to fix which is the step-size.

In what follows, we will begin our discussion by the derivation of the the new normalisation scheme for the FDAF-OLA and FDAF-OLS algorithms.

2 The FDAF Projection Algorithms

2.1 The FDAF Projection OLA Algorithm

Consider an adaptive transversal filter, $W_{N,k}$ whose zero-padded DFT of order $2N$ is $W^f_{2N,k} = F_{2N}\Phi^T_N W_{N,kN}$, $\Phi_N = [\, I_N \; 0_{N\times N} \,]$ (F_M is the DFT matrix of order M: $(F_M)_{lm} = \exp(-j2\pi(l-1)(m-1)/M)$) and consider the diagonal matrix whose main diagonal is the DFT of the current N-block input data padded with N zeros: $\overline{X}^f_{2N}(k) = \mathrm{diag}\{F_{2N}\Phi^T_N[x_{kN}\cdots x_{kN+N-1}]^T\}$. The OLA technique leads to the use of the following input transformed matrix: $X^f_{2N}(k) = \overline{X}^f_{2N}(k) + \Omega\overline{X}^f_{2N}(k-1) = \mathrm{diag}\{X_0(k)\cdots X_{2N-1}(k)\}$, ($\Omega$ diagonal with $(\Omega)_{ll} = (-1)^{l+1}$). With these conventions, the output filter may be expressed in the frequency domain as: $Y^f_{2N}(k) = X^f_{2N}(k)W^f_{2N,k}$ and the corresponding inverse DFT (IDFT) is the time domain output filter where only the first N components correspond to the linear convolution of the input data and the adaptive filter (the other components correspond to

circular convolution). Hence, the FDAF-OLA algorithm is given by [Shy92]:

$$\begin{aligned} Y_{2N}^f(k) &= X_{2N}^f(k)W_{2N,k}^f \\ y_N(k) &= \Phi_N F_{2N}^{-1} Y_{2N}^f(k) \\ e_N(k) &= d_N(k) - y_N(k) \\ E_{2N}(k) &= F_{2N}\Phi_N^T e_N(k) \\ & m = 0,\ldots,2N\text{–}1 \\ & P_m(k) = \lambda P_m(k\text{–}1) + (1\text{–}\lambda)|X_m(k)|^2 \\ \Upsilon_k &= \mu\,(\mathrm{diag}\{P_0(k),\cdots,P_{2N-1}(k)\})^{-1} \\ W_{2N,k+1}^f &= W_{2N,k}^f + \Pi X_{2N}^{fH}(k)\Upsilon_k E_{2N}(k) \end{aligned} \quad (1)$$

where $\Pi = F_{2N}\Phi_N^T\Phi_N F_{2N}^{-1}$ is the constraint matrix that forces the last N components of the frequency domain adaptive filter to be zero ($\Pi = I_N$ in the unconstrained case), $d_N(k) = [\,d_{kN}\cdots d_{kN+N-1}\,]^T$ is the N-block desired signal and $0 < \lambda < 1$ is the forgetting factor. In general, $W_{2N,0} = 0_{2N}$ and $P_m(0) = \delta$, $m = 0\ldots 2N\text{–}1$ ($\delta > 0$).
In the noiseless case and when the unknown filter W_{2N}^{of} is time-invariant, the frequency deviation filter is $V_{2N,k+1}^f = W_{2N,k+1}^f - W_{2N}^{of} = \Pi\left(I - X_{2N}^{fH}(k)\Upsilon_k\Pi X_{2N}^f(k)\right)V_{2N}^f(k)$. In order to have a recursive projection scheme for the deviation vector, we introduce a normalisation matrix $\Upsilon_k = \mu\Pi T_k$. The deviation filter becomes: $V_{2N,k+1}^f = \Pi\left(I - \mu X_{2N}^{fH}(k)\Pi T_k\Pi X_{2N}^f(k)\right)V_{2N,k}^f$. Now, if we want to have a projection onto the orthogonal subspace ($\mu = 1$) to the column space of $X_{2N}^{fH}(k)\Pi$, we clearly have to choose T_k to be the inverse of $\left(\Pi X_{2N}^f(k)X_{2N}^{fH}(k)\Pi\right)$ but this last matrix is singular (Π is not full-rank). Hence, we take its pseudo-inverse:

$$\begin{aligned} T_k &= \left(\Pi X_{2N}^f(k)X_{2N}^{fH}(k)\Pi\right)^\dagger \\ &= F_N^1\left(F_N^{1H}X_{2N}^f(k)X_{2N}^{fH}(k)F_N^1\right)^{-1}F_N^{1H}\,, \end{aligned} \quad (2)$$

where $F_{2N} = \left[F_N^1\ F_N^2\right] = \left[F_N^1\ \Omega F_N^1\right]$. With the introduction of the pseudo-inverse, the filter update equation becomes:

$$\begin{aligned} W_{2N,k+1}^f &= W_{2N,k}^f + \\ &\mu\Pi X_{2N}^{fH}(k)\Pi F_N^1\left(F_N^{1H}X_{2N}^f(k)X_{2N}^{fH}(k)F_N^1\right)^{-1}F_N^{1H}E_{2N}(k)\,. \end{aligned} \quad (3)$$

Now, notice that $\Pi F_N^1 = F_N^1$, $\frac{1}{2N}F_N^{1H}E_{2N}(k) = e_N(k)$ and $\frac{1}{2N}F_N^{1H}X_{2N}^f(k)X_{2N}^{fH}(k)F_N^1$ is a real Toeplitz symmetric matrix with the first row being the first N components of the IDFT of the main diagonal of $X_{2N}^f(k)X_{2N}^{fH}(k)$. Therefore, the inversion of this matrix is performed efficiently by using the Levinson algorithm whose computational complexity is $2N$ operations per sample. It could also be done using the doubling Levinson algorithm whose computational complexity is $\mathcal{O}((\log(N))^2)$ operations per sample. Note that the N correlation lags obtained through this operation could be directly computed in the time domain by performing the circular correlation of the input data block of length $2N$. Hence the FDAF-Projection OLA (FDAF-POLA) algorithm is given by the following set of equations:

$$\begin{aligned} Y_{2N}^f(k) &= X_{2N}^f(k)W_{2N,k}^f \\ y_N(k) &= \Phi_N F_{2N}^{-1}Y_{2N}^f(k) \\ e_N(k) &= d_N(k) - y_N(k) \\ r_N(k) &= \Phi_N F_{2N}^{-1}X_{2N}^f(k)X_{2N}^{fH}(k)1_{2N} \\ &\text{Levinson Procedure :} \\ &\text{Input: } \{r_N(k), e_N(k)\}\ ,\ \text{Output: } \{z_N(k)\} \\ Z_{2N}(k) &= F_{2N}\Phi_N^T z_N(k) \\ W_{2N,k+1}^f &= W_{2N,k}^f + \mu\Pi X_{2N}^{fH}(k)Z_{2N}(k)\ , \end{aligned} \quad (4)$$

where $1_M = [\,1\cdots 1\,]^T$.

2.2 The FDAF Projection OLS Algorithm

When using OLS sectioning, the input transformed data matrix is: $X_{2N}^f(k) = \mathrm{diag}\{F_{2N}[x_{kN-N}\cdots x_{kN+N-1}\,]^T\}$. In the OLS sectioning method, only the last $N+1$ components of the IDFT of $X_{2N}^f(k)W_{2N,k}^f$ correspond to a linear convolution. In order to avoid circular convolution for the gradient, the error vector must be padded with N zeros in a different way than in the FDAF-OLA algorithm. Denoting $\widetilde{\Phi}_N = [0_{N\times N}\ I_N]$, the FDAF-OLS algorithm is given by the same set of equations as the one of the FDAF-OLA algorithm (1), except the fact that Φ_N is replaced by $\widetilde{\Phi}_N$.
In this case, the deviation filter (in the noiseless case) is $V_{2N,k+1}^f = \Pi\left(I - X_{2N}^{fH}(k)\Upsilon_k\widetilde{\Pi}X_{2N}^f(k)\right)V_{2N,k}^f$ where $\widetilde{\Pi} = F_{2N}\widetilde{\Phi}_N^T\widetilde{\Phi}_N F_{2N}$. Let us introduce the normalisation matrix $\Upsilon_k = \mu\widetilde{\Pi}T_k$. In order to have a recursive projection scheme, we choose:

$$\begin{aligned} T_k &= \left(\widetilde{\Pi}X_{2N}^f(k)X_{2N}^{fH}(k)\widetilde{\Pi}\right)^\dagger \\ &= F_N^2\left(F_N^{2H}X_{2N}^f(k)X_{2N}^{fH}(k)F_N^2\right)^{-1}F_N^{2H}\,, \end{aligned} \quad (5)$$

so that the deviation vector is projected onto the orthogonal subspace ($\mu = 1$) to the column space of $X_{2N,k}^{fH}\widetilde{\Pi}$. Note that in the case of the FDAF-POLA algorithm, the projection was done onto the orthogonal subspace to the column space of $X_{2N,k}^{fH}\Pi$. Note also that because $X_{2N}^f(k)$ is diagonal, we have:

$$\begin{aligned} F_N^{2H}X_{2N}^f(k)X_{2N}^{fH}(k)F_N^2 &= F_N^{1H}\Omega X_{2N}^f(k)X_{2N}^{fH}(k)\Omega F_N^2 \\ &= F_N^{1H}X_{2N}^f(k)X_{2N}^{fH}(k)F_N^1\,, \end{aligned} \quad (6)$$

hence, the matrices to invert are the same for the OLA and OLS methods. With $\widetilde{\Pi}F_N^2 = F_N^2$ and $\frac{1}{2N}F_N^{2H} = e_N(k)$, the FDAF-Projection OLS (FDAF-POLS) algorithm is finally given by the same set of equation as the one of the FDAF-OLA algorithm (4) except that Φ_N is replaced by $\widetilde{\Phi}_N$. By comparing the numerical complexities of the FDAF-P algorithms and the corresponding FDAF algorithms, we see that FDAF-P algorithms use an additional FFT of length $2N$ to compute the N correlation lags $r_N(k)$ and perform a generalized Levinson algorithm which costs $2N$ operations per sample. We will see that this additional complexity will be smaller with the MDF and GMDFα algorithms. Note also the similarity of the new algorithms with the so-called

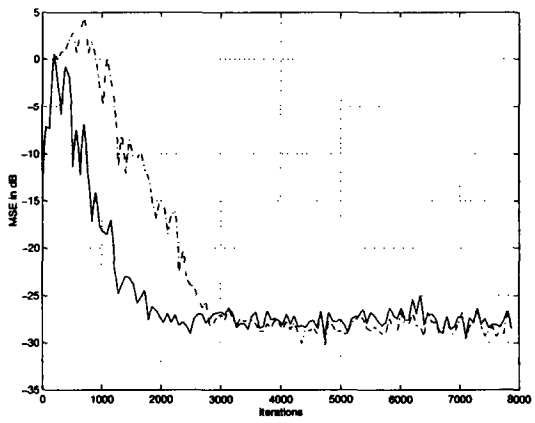

Figure 1: Comparison of FDAF-OLA ('-.') and FDAF-POLA ('–') algorithms ($N = 256, \mu = 1$).

Self-Orthogonalizing Block Adaptive Filter (SOBAF) algorithm [PMCG86] where the correlation lags are computed recursively in the time-domain. The new algorithms could be seen as the SOBAF algorithm where the correlation lags are estimated through the DFT in place of classical recursive schemes (exponential or sliding window).
We have simulated the FDAF-POLA and FDAF-POLS algorithms in various situations. Our conclusion is that the projection scheme yields to faster convergence speed.
In Fig.(1), we give the learning curve (averaged over 128 samples) for the FDAF-OLA and the FDAF-POLA algorithms. The input is a correlated noise, and a white noise is added to the desired signal so that the SNR is 30 dB. $N = 256$, $\mu = .7$ for the two algorithms and $\lambda = .7$, $\delta = 100$ for the FDAF-OLA algorithm. As one can see, the FDAF-POLA algorithm converges faster than the FDAF-OLA algorithm.
However, FDAF algorithms are not used in practice when the filter length is relatively large as is the case for AEC applications. This is due to the large delay which is inherent to these algorithms since adaptation is done every N samples. In what follows, we will apply the projection scheme to the MDF algorithm which overcomes the delay drawback.

3 The MDF Projection Algorithm

In [SP90], the MDF algorithm is introduced. The main feature of this algorithm is to allow a flexible delay. This is done by segmenting the adaptive filter in M blocks of length L ($M = \frac{N}{L}$). This technique reduces the delay and allows a more frequent updating of the adaptive filter (M times more than in the FDAF algorithms). In the MDF algorithms, FFTs of length L are computed. This fact leads to a smaller quantization error if compared with FFTs of length N and permits a more efficient use of the hardware since DSP chips are in general optimized for relatively small size FFTs (FFTs are typically of order 256). In what follows, we shall only concern ourselves with the OLS sectioning method. The extension to the OLA method or to the unconstrained methods being straightforward.
In the MDF-OLS algorithms, we consider the transformed data matrix: $X_{2L}^{f\,1}(k) = \mathrm{diag}\{F_{2L}[x_{kL-L}\cdots x_{kL+L-1}]^T\}$. Define: $W_{2L,k}^{f\,m} = F_{2L}\Phi[W_{kL}(mL)\cdots W_{kL}(mL+L-1)]^T$, the frequency domain output filter of length L is computed as:

$$Y_{2L}^{f}(k) = \sum_{m=1}^{M} X_{2L}^{f\,m}(k) W_{2L,k}^{f\,m} = \mathbf{X}_{2N}^{f}(k) W_{2N,k}^{f}\,, \qquad (7)$$

where the $(2L \times 2N)$ data matrix is: $\mathbf{X}_{2N}^{f}(k) = [X_{2L}^{f\,1}(k)\cdots X_{2L}^{f\,M}(k)]$ and $W_{2N,k}^{f} = [W_{2L,k}^{f\,1}\cdots W_{2L,k}^{f\,M}]^T$. Note that $X_{2L}^{f\,(m+1)}(k) = X_{2L}^{f\,m}(k-1)$, $1 \le m < M$, so only one FFT of length $2L$ need to be computed at time kL, the $M-1$ FFTs corresponding to the previous blocks of length L being stored in memory. The MDF algorithm is given by the same set of equations as those of the FDAF algorithm, except for the normalization which is done by averaging over every M blocks (see [SP90]):

$$\begin{aligned}
X_{2L}^{f\,(m+1)}(k) &= X_{2L}^{f\,m}(k-1)\,,\ 1 < m \le M \\
Y_{2L}^{f}(k) &= \sum_{l=1}^{M} X_{2L}^{f\,l}(k) W_{2L,k}^{f\,l} \\
y_L(k) &= \widetilde{\Phi}_L F_{2L}^{-1} Y_{2L}^{f}(k) \\
e_L(k) &= d_L(k) - y_L(k) \\
E_{2L}(k) &= F_{2L}\widetilde{\Phi}_L^T e_L(k) \\
P_{2L}^{m+1}(k) &= P_{2L}^{m}(k-1)\,,\ m = 1\ldots M-1 \\
P_{2L}^{M}(k) &= X_{2L}^{f\,1}(k) X_{2L}^{f\,1\,H}(k) 1_{2L} \\
\beta_{2L}(k) &= \lambda\beta_{2L}(k-1) + (1-\lambda)\sum_{l=1}^{M} P_{2L}^{l}(k) \\
\Upsilon_k &= \mu\left(\mathrm{diag}\{\beta^1(k),\cdots,\beta^{2L-1}(k)\}\right)^{-1} \\
W_{2L,k+1}^{f\,l} &= W_{2L,k}^{f} + \Pi X_{2L}^{f\,H}(k)\Upsilon_k E_{2L}(k)\,,\ l = 1\ldots M.
\end{aligned} \qquad (8)$$

The computational complexity of the MDF-OLS algorithm is $4\frac{N}{L} + \left(3 + 2\frac{N}{L}\right)\frac{FFT(2L)}{L}$ operations per sample where $FFT(L)$ is the number of operations needed for computing an FFT of order L. The adaptive filter update can be rewritten in the following form:

$$W_{2N,k+1}^{f} = W_{2N,k}^{f} + \Psi\mathbf{X}_{2N}^{f\,H}(k)\Upsilon_k E_{2L}(k)\,, \qquad (9)$$

where

$$\Psi = \begin{bmatrix} \Pi & 0 & \cdots & 0 \\ 0 & \Pi & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Pi \end{bmatrix}, \qquad (10)$$

is the $(2N \times 2N)$ constraint matrix. The deviation filter is: $V_{2N,k+1}^{f} = \Psi\left(I - \mathbf{X}_{2N}^{f\,H}(k)\Upsilon_k\widetilde{\Pi}\mathbf{X}_{2N}^{f}(k)\right)V_{2N,k}^{f}$. In order to realize the projection scheme, we take $\Upsilon_k = \mu\widetilde{\Pi}T_k$ with:

$$T_k = \left(\widetilde{\Pi}D_{2L}(k)\widetilde{\Pi}\right)^{\dagger} = F_{2L}^2\left(F_{2L}^{2H}D_{2L}(k)F_{2L}^2\right)^{-1}F_{2L}^{2H}\,, \qquad (11)$$

where:

$$\begin{aligned}
D_{2L}(k) &= \mathbf{X}_{2N}^{f}(k)\mathbf{X}_{2N}^{f\,H}(k) = D_{2L}(k-1) + \\
&\quad X_{2L}^{f\,1}(k)X_{2L}^{f\,1\,H}(k) - X_{2L}^{f\,M}(k-1)X_{2L}^{f\,M\,H}(k-1)\,,
\end{aligned} \qquad (12)$$

and $D_{2L}(k)$ is simply initialized by $D_{2L}(0)1_{2L} = 0_{2L}$. Hence the MDF Projection OLS (MDF-POLS) algorithm is given

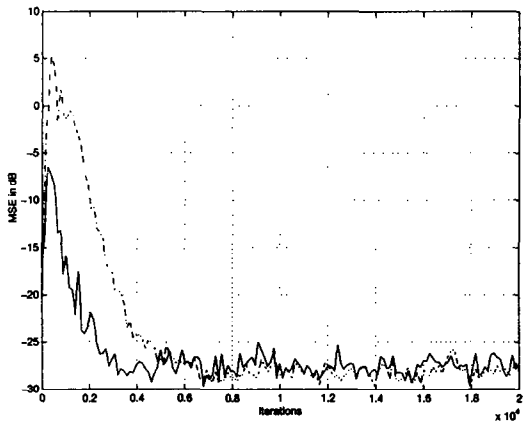

Figure 2: Comparison of the MDF-OLS ('-.') and MDF-POLS ('–') algorithms ($N = 256, \mu = 1$).

by:

$$
\begin{aligned}
X_{2L}^{f\,(m+1)}(k) &= X_{2L}^{f\,m}(k\text{–}1)\ ,\ 1 < m \leq M \\
Y_{2L}^{f}(k) &= \sum_{l=1}^{M} X_{2L}^{f\,l}(k) W_{2L,k}^{f\,l} \\
y_L(k) &= \widetilde{\Phi}_L F_{2L}^{-1} Y_{2L}^{f}(k) \\
e_L(k) &= d_L(k) - y_L(k) \\
D_{2L}(k) &= D_{2L}(k\text{–}1) + X_{2L}^{f\,1}(k) X_{2L}^{f\,1\,H}(k) \\
&\quad - X_{2L}^{f\,M}(k\text{–}1) X_{2L}^{f\,M\,H}(k\text{–}1) \\
r_L(k) &= \Phi_L F_{2L}^{-1} D_{2L}(k) 1_{2L} \\
&\text{Levinson Procedure :} \\
&\text{Input: } \{r_L(k), e_L(k)\}\ ,\ \text{Output: } \{z_L(k)\} \\
Z_{2L}(k) &= F_{2L} \widetilde{\Phi}_L^T z_L(k) \\
W_{2L,k+1}^{f\,l} &= W_{2L,k}^{f\,l} + \mu \Pi X_{2L}^{f\,l\,H}(k) \Upsilon_k E_{2L}(k)\ ,\ l = 1 \ldots M.
\end{aligned}
\tag{13}
$$

The complexity of the MDF-POLS algorithm is $4\frac{N}{L} + \left(4 + 2\frac{N}{L}\right)\frac{FFT(2L)}{L} + 2L$ operations per sample, *i.e.*, $2L + \frac{FFT(2L)}{L}$ operations more than the MDF-OLS algorithm. Fig.(2) gives the MSE evolution for the MDF-OLS and MDF-POLS algorithms, in this simulation the input was a highly correlated signal (AR(20) model) and a white noise was added to the desired signal with SNR=30 dB. $N = 256$, $L = 64$, $\mu = 1$ for the two algorithms and $\lambda = .9$, $\delta = 100$ for the MDF-OLS algorithm. This situation shows clearly the convergence improvement of the projection algorithm.

4 The GMDF Projection Algorithm

The MDF algorithm can be seen as a particular case of a more general scheme, namely the GMDFα algorithm. The main idea of this algorithm is the introduction of a parameter α that controls the overlap between successive input blocks of length L. In fact, for fast convolution, there is no need to overlap successive input blocks but within an adaptive filtering framework, this overlapping factor allows the adaptive filter to be updated every $R = L/\alpha$ samples,*i.e.*, α times more than in the MDF algorithm. For the GMDFα algorithm, the new normalisation technique is done in the same way as for the MDF algorithm and leads to the GMDF Projection α (GMDFPα) algorithm. Simulations in stationary environments showed us the convergence enhancement

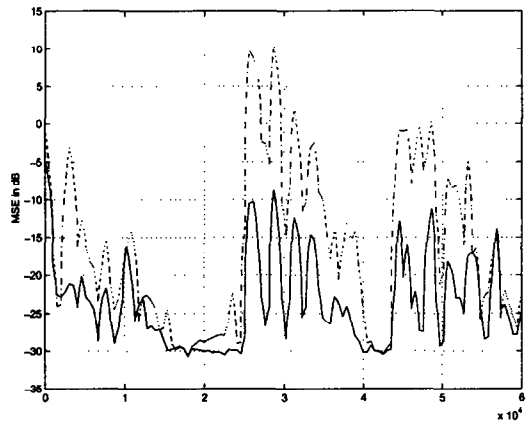

Figure 3: Comparison of the GMDF4 ('-.') and the GMDFP4 ('–') algorithms ($N = 512, \mu = .7$).

of the GMDFPα algorithm. However, with an input speech signal, we were confronted with the noise amplification phenomenon that is due to the ill-conditioning of the covariance matrix. In this case, we have regularized the matrix by simply replacing $r_L(1)$ by $r_L(1) + \rho$ where ρ is a small real positive number. Fig.(3), gives the MSE evolution in the case of speech signal input. $N = 512, L = 128$, $\mu = .5$, $\rho = .1$ for the GMDFP4 algorithm and $\lambda = .95$, $\delta = 10$ for the GMDF4 algorithm. It is clearly shown that the GMDFP4 algorithm performs better.

5 Concluding remarks

We have derived new frequency domain adaptive algorithms by introducing a new normalisation scheme based on a projection approach. This normalisation improves the convergence speed of the classical frequency domain algorithms. Unfortunately, this kind of projection suffers from noise amplification. One possible approach to combat the noise is to regularize the covariance matrix by simply adding a constant diagonal matrix. Another possibility is to use an exponential window for the computation of the correlation lags. Noise amplification and analysis of the new algorithms are the subjects of our ongoing research.

REFERENCES

[MtAG95] E. Moulines, O. Aït Amrane, and Y. Grenier. "The Generalized Multidelay Adaptive Filter: Structure and Convergence Analysis". *IEEE Trans. SP*, SP-43(1):2184–2193, Jan. 1995.

[PMCG86] G. Panda, B. Mulgrew, Colin F. N. Cowan, and P. M. Grant. "A Self-Orthogonalizing Efficient Block Adaptive Filter". *IEEE Trans. ASSP*, ASSP-34(6):1573–1582, Dec. 1986.

[Shy92] J.J. Shynk. "Frequency-Domain and Multirate Adaptive Filtering". *IEEE ASSP Magazine*, 9(1):15–37, January 1992.

[SP90] J. Soo and K. Pang. "Multidelay block Frequency Domain Adaptive Filters". *IEEE Trans. ASSP*, ASSP-38(2):373–376, Feb. 1990.