JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

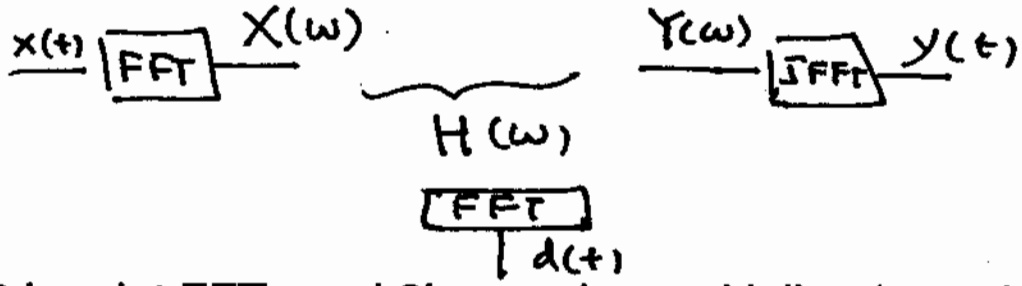
FREQUENCY DOMAIN ADAPTIVE FILTERS

Why FDAF?

1. The amount of computation required can be greatly reduced.
2. Convergence properties can be improved.

An L lag filter requires (convolution) $2L^2$ multiplies. $L = M$

In the frequency domain this operation may require fewer operations.



Require 3 L point FFTs and 2L complex multiplies (complex W). For real input data the L point FFT can be implemented with L/2 point FFT, and L/2 complex mul.

Each FFT takes (radix 2) $\frac{M}{4} \log \frac{M}{2} - \frac{M}{2}$

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

If 4 real multiplies for each complex mul.

$$3M \log_2 \frac{M}{2} + 4M$$

Compared with $2M^2$. (5% for 256, 1.5% for 1024).

The 2 problems with the frequency domain approach are:

1. It produces circular convolution
2. Implies block computation.

What is important is how much use the algorithm makes of the available information containing in the points. We can achieve the same solution adapting fewer times, but more accurately.

We will show that in the FDAF the adaptation is as fast as the LMS, when the signal spectrum has a small dynamic range. The problem of the eigenvalue spread is worse in the FDAF,

BUT IT CAN BE EASILY CONTROLLED WITH THE NORMALIZED LMS IDEA.

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

DEFINITIONS

Let us define F as a symmetric MxM matrix where i,kth coefficients is

$$F_{i,k} = \exp\left(-j\frac{2\pi ik}{M}\right)$$

When F operates on a vector the result is the DFT of the vector. Similarly F^{-1} is the IDFT, and

$$\overline{F}^{*} = MF^{-1}$$

If $\vec{c}$ is a circulant matrix (all rows of the square matrix are obtained by successive right circular shifts of the first row). Then

$$F\,c\,F^{-1}$$

is a diagonal matrix, whose elements are the DFT of the first column of $\vec{c}$.

Remember also that the complex LMS is

$$W_{k+1} = W_k + 2\mu\,\varepsilon_k X_k^{*}$$

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

FREQ. DOMAIN LMS

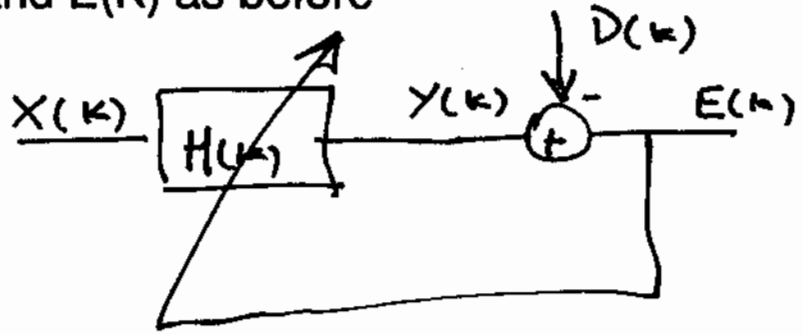
Define the freq. domain weight vector for the kth block of data as

$$H^T(k) = [H_1(k), \ldots, H_M(k)]$$

and the diagonal matrix of the input $\overset{DFT}{\lambda}$ as

$$X(k) = \begin{bmatrix} X_1(k) & & 0 \\ & X_2(k) & \\ 0 & & \ddots \\ & & X_M(k) \end{bmatrix}$$

Y(k), D(k) and E(K) as before



$$\begin{cases} Y(k) = H(k)X(k) \\ E(k) = D(k) - Y(k) \end{cases}$$

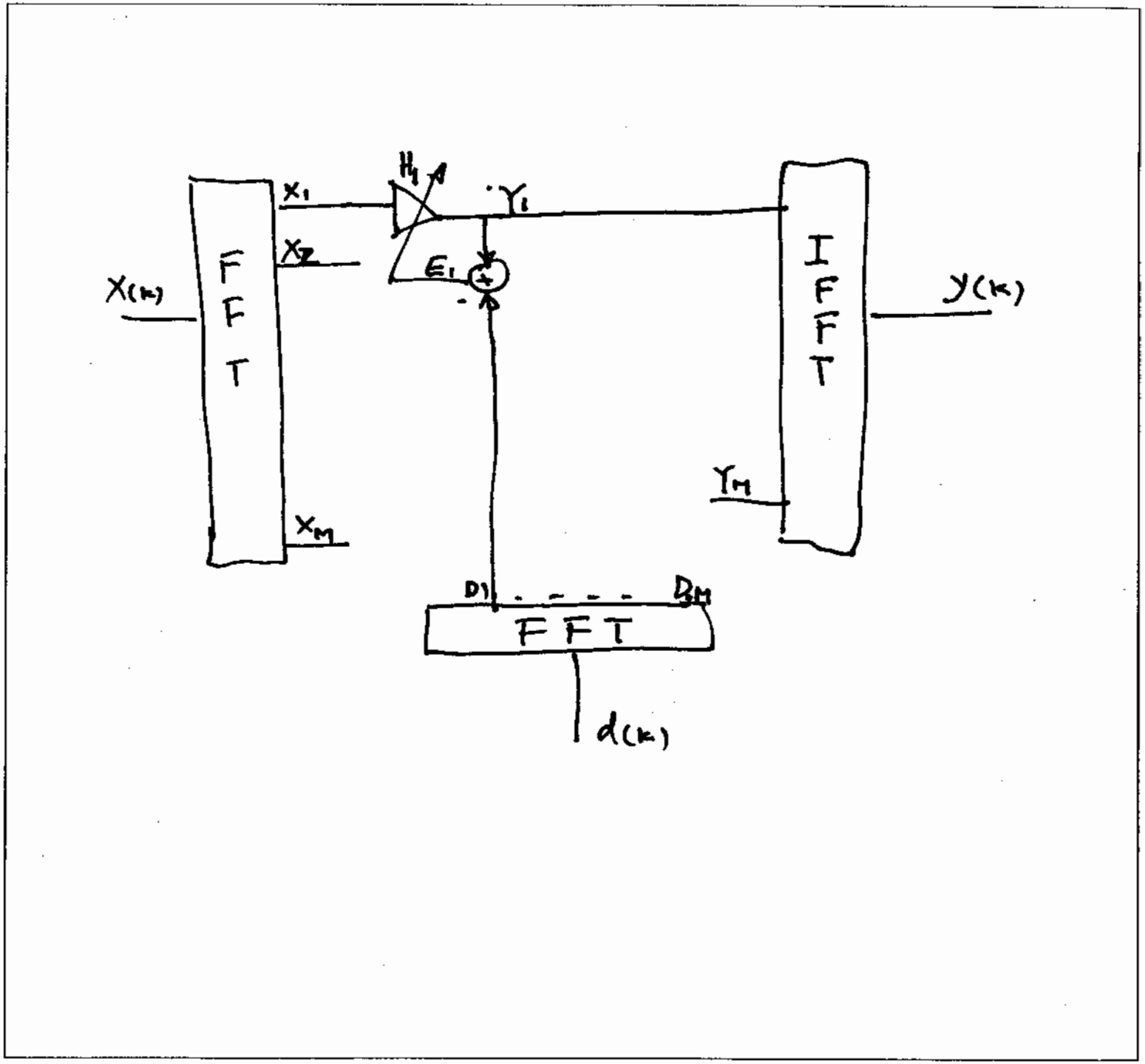
JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu



JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

Each weight is adapted independently. The freq. domain weight update can be expressed as

$$H(k+1) = H(k) + 2\mu X^*(k) E(k)$$

$$= H(k) + 2\mu \{ X^*(k) D(k) - X^*(k) X(k) H(k) \}$$

What is the time domain equivalent of this?

$$h(k+1) = h(k) + 2\mu \{ \alpha_k^T d_k - \alpha_k^T \alpha_k h_k \}$$

d_k and h_k are IDFT and a

$$\alpha_k = F^{-1} X_k F$$

is a circulant matrix. The first column is the input vector

$$\alpha_k = \begin{bmatrix} x(k) & x(k+M-1) & \cdots & x(k+1) \\ x(k+1) & x(k) & \cdots & x(k+2) \\ \vdots & & & \\ x(k+M-1) & x(k+M-2) & \cdots & x(k) \end{bmatrix}$$

Denoting the ith row of a(k) by x^T_i then we can write

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

$$h(k+1) = h(k) + 2\mu \sum_{i=1}^{M} \{d_i(k) x_i(k) - y_i(k) x_i(k)\}$$

where $y_i(k)$ is the ith element of the output vector

$$y(k) = \alpha(k)\, h(k)$$

The vector y(k) contains the elements of the kth output block. The elements are obtained by circularly convolving the impulse response h(k) with rotated versions of the input vector.

So the equivalent weight update in the time domain is

$$h(k+1) = h(k) + 2\mu \sum_{i=1}^{M} e_i(k) x_i(k)$$

Comparing this with the LMS, we see that the gradient estimates are added over the entire block of data before updating the weights.

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

OPTIMUM WEIGHT VECTOR

Can derive it by noting that (assume stationarity)

$$E[d_k - y_k]^2 = E[(d_k - y_k)^*(d_k - y_k)]$$

$$= E\{[D(k) - Y(k)]^*[D(k) - Y(k)]\}$$

$$= E[D^*(k)D(k)] = R_{xD}^* H - H^* R_{xD} + H^* R_{xx} H$$

R_{xx} is diagonal and its ith diagonal element is

$$E[X_i^*(k) X_i(k)] \qquad (X_i(k) \rightarrow \text{DFT of } x_i(t))$$

Now making $\frac{\partial E}{\partial H} = 0$

$$-R_{xD}^* - R_{xD}^* + R_{xx} H + R_{xx} H = 0$$

$$\Longrightarrow \boxed{H_{op} = R_{xx}^{-1} R_{xD}}$$

Taking the IDFT we obtain

$$h_{op} = r_{xx}^{-1} r_{xd}$$

$$\begin{cases} r_{xx} = F^{-1} R_{xx} F \\ r_{xd} = F^{-1} R_{xd} F \end{cases}$$

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

The matrix r_{xx} is circulant since R_{xx} is diagonal. The 1st row is given by lags 0 to N-1 of the circular autocorrelation function of $(\phi_c(i))$ x(n). Re member that

$$\phi_c(i) = \frac{N-i}{N}\phi(i) + \frac{i}{N}\phi(i-N)$$

where $\phi(i)$ is the true autocorrelation function.

This analysis shows that the output of the LMS in the frequency domain produces an aliased version of the output, and in no way can be compared with the output of the LMS in the time domain.

In some applications (noise cancellation) aliasing is no problem, but in system identification aplications and signal enhancement, aliasing is not acceptable, and this straight implementation is not used.

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

CONVERGENCE PROPERTIES

Assuming stationary inputs and taking E of (pp40

$$E[H(k+1)] = E[H(k)] + \mu \{ R_{xd} - R_{xx} E[H(k)] \}$$

and making the assumption of uncorrelated R and H, for sufficient small μ

$$\lim_{k \to \infty} E[H(k)] = R_{xx}^{-1} R_{xd} = H_{OPT}$$

Under these assumptions, the weight values are unbiased.

The condition for convergence is

$$\mu < \frac{2}{\lambda_{MAX}}$$

$$\left(R_{xx} = \begin{bmatrix} E[x_1(k) x_1(k)^*] & 0 \\ 0 & E[x_2 \cdot x_2^*] \\ & \ddots \end{bmatrix} \right)$$

where λ_{max} is the largest eigenvalue of R. Since R is diagonal, the eigenvalues are the elements of the diagonal, so largest eigenvalue is the largest value of the power in the DFT bins.

The time constant associated with the p mode is,

$$\tau_p = \frac{1}{\mu \lambda_p} \text{ BLOCKS OR } \tau_p = \frac{1}{M \mu \lambda_p} \text{ SAMPLES}$$

WEIGHTS CONVERGE INDEPENDENTLY

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

MISADJUSTEMENT

Same definition,

$$\text{MISADJUSTMENT} = \frac{\mu}{2M} tr[R_{xx}] = \frac{\mu P M^2}{2M}$$

The stability condition implies

$$Mis < \frac{\lambda_{avg}}{\lambda_{MAX}} \Rightarrow \tau_p = \frac{M}{2 Mis.} \frac{\lambda_{ave}}{\lambda_p}$$

The only difference is that both τ and M are given as a function of the power of FFT bins. These results also show that the speed of adaptation is the same for the same misadjustment between the LMS and this algorithm.

NORMALIZED FDAF

$$h_i(k+1) = h_i(k) + \frac{2\mu}{\frac{1}{N}\sum_{j=k-N+1}^{k} |x_i(j)|^2} X_i^*(k)\, e_i(k)$$

THIS SOLVES THE EIGENSPREAD PROBLEM !

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

FREQUENCY DOMAIN LMS

This algorithm does display aliasing. How can we avoid it?

Easiest way is to add zeros to the coefficient vector. From the two block processing methods (overlap and save, overlap and add), the overlap and save is superior for adaptive signal processing (50% block overlap).

ALGORITHM

N data points are collected, and filter coefficients are kept constant during the block. During the kth block

$$h(k+1) = h(k) + \mu \sum_{i=0}^{M-1} e(kM+i)\, x(kM+i)$$

$$= h(k) + \mu \nabla(k)$$

$$y(kM+i) = h^T(k)\, x(kM+i) \qquad i = 0, \ldots, M-1$$

$$h_k^T = \begin{bmatrix} h_0(k) \\ \vdots \\ h_{M-1}(k) \end{bmatrix} \qquad x^T(n) = \begin{bmatrix} x(n) \\ \vdots \\ x(n-M+1) \end{bmatrix} \quad ; \; e(n) = d(n) - y(n)$$

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

Compute this in the Fre. domain without aliasing. So define

$$H^T(k) = \mathcal{F}\left[h^T(k), \underbrace{0, \ldots 0}_{M \text{ ZEROS}} \right]$$

$$X(k) = \text{diag}\left\{ \mathcal{F}\left[\underbrace{X(kM-M), \ldots X(kM-1)}_{k-1 \text{ block}}, X(kM) \ldots X(kM+M-1) \right] \right.$$

$$Y(k) = \text{last } M \text{ terms of } \mathcal{F}^{-1}\{ X_k H_k \}$$

To implement the update equation, notice that the jth element ∇_k

$$\nabla_j(k) = \sum_{i=0}^{M-1} e(kM+i)\, x(kM+i-j) \qquad j = 0, 1, \ldots M-1$$

so the elements are the cross correlation of the error sequence with the filter input. So $\nabla(k)$ can be computed if we first take the FFT of the error preceeded by M zeros.

$$E(k) = \mathcal{F}\left[0, \ldots 0, \underbrace{[d(kM) - y(kM)] \ldots, [d(kM+M-1) - y(kM+M-1)]}_{k^{th} \text{ error block}} \right]$$

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

$$\nabla(k) = 1^{st}\ M \text{ TERMS OF } \mathcal{F}^{-1}\{X^*_{(k)} E(x)\}$$

AND

$$H(k+1) = H(k) + \mu\, \mathcal{F}\begin{bmatrix} \nabla(k) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \left.\vphantom{\begin{bmatrix} \nabla(k) \\ 0 \\ \vdots \\ 0 \end{bmatrix}}\right\} M \text{ ZEROS}$$

This is an exact computation, i.e. we are computing the LMS filter output. Updating is done in blocks. This algorithm is called the FAST LMS.

We are paying a penalty to compute the filter output without aliasing

1. FFT of order 2N
2. Compute several FFTs to compute the update,

5-2N FFTs and 2N complex multiplies for updating.

So

$$\frac{5M}{2} \log_2 M + 4M$$

JOSE C. PRINCIPE

UNIVERSITY OF FLORIDA

EEL 6935- SPRING 90

904-392-2662

principe@brain.ee.ufl.edu

The ratioFLMS/LMS is

$$\frac{5\log_2 M + 4}{M}$$

M=16	1.5
256	0.17
1024	0.053

CONVERGENCE PROPERTIES

$$E[h(k+1)] = E[h(k)] + \mu\{M(r_{xd} - r_{xx}E[h(k)]\}$$

$$r_{xx} = E[x(n)x^T(n)]$$

$$r_{xd} = E[d(n)x(n)]$$

$$\mu < \frac{2}{M\lambda_{MAX}}$$

Notice that now λ_{max} is the eigenvalue of the autocorrelation matrix, so we can see that the algorithm is more restrictive for convergence, since now we have to set μ M times smaller (slower convergence).

The misadjustment is $M_{rs} = \frac{\mu}{2} tr[r_{xx}]$

which is the same as the time domain LMS.