

频域 LMS 算法在语音消噪中的应用

王布宏, 郭英

(空军工程大学 电讯工程学院, 陕西 西安 710077)

摘要: 推导了一种替代时域 LMS 算法的快速频域算法 (FLMS), 计算机仿真的结果表明: 它在自适应滤波器权数超过 64 时, 运算量较时域 LMS 算法有大幅度的下降, 但保持了与时域 LMS 算法相同的收敛速度。同时对算法的局限性和应用范围进行了讨论。

关键词: 自适应滤波器; 算法; 频域; LMS

中图分类号: TN713.7 **文献标识码:** A **文章编号:** 1009-3516(2000)03-0064-04

在自适应技术中, 参考信号通过自适应滤波器改变权系数, 由此产生的误差信号对期望信号中相关分量进行最佳估计。LMS 算法是一种误差信号对横向自适应滤波器权系数进行更新控制的递归算法, 它以其算法简单、易行的特点被广泛应用于诸如系统识别、干扰抵消、信道均衡、谱线增强、自适应天线等各种自适应系统中。实际应用中为了更好地逼近任何所希望的频率响应特性, N 应取得较大, 但运算量却以 N 的平方的比例迅速增加, 妨碍了系统的实时实现, 因此 LMS 算法的快速实现有很大的实际意义。本文利用快速 FFT 技术实现了 LMS 频域快速算法 (FLMS)。并保持了与传统 LMS 算法相近的收敛速度。

1 自适应滤波器 LMS 频域算法 (FLMS) 的推导

1.1 LMS 时域算法

Windrow-Hoff 于 1959 年提出的 LMS (Least Mean Square) 算法是自适应横向滤波器中求最佳权系数的简单有效的递归算法, 它不要求相关矩阵, 也不涉及矩阵求逆, 而是运用了最优化的数学算法——最陡下降法 (steepest descent method)。

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

$$W_j^T = [w_{0,j}, w_{1,j}, w_{2,j}, \dots, w_{N-1,j}]$$

$$X_j^T = [x_j, x_{j-1}, x_{j-2}, \dots, x_{j-N+1}]$$

$$e_j = d_j - y_j$$

$$y_j = X_j W_j^T$$

其中 X_j 为输入矢量, W_j 为权矢量, d_j 为参考信号, μ 为收敛步长, e_j 为误差信号。

1.2 FLMS 频域算法的推导

在 LMS 频域实现中, 将参考信号分割成 N 长 (自适应滤波器的权数) 的块, 权系数的更新每 N 个样点进行一次, 而每次的更新是由 N 个误差信号样点累加结果控制的, 这既保证了与时域 LMS 算法有相同的收敛速度, 同时可以利用快速 FFT 技术, 用序列的循环卷积来计算线性卷积, 从而大幅度地减少运算量。

在频域实现中输入序列分割成 N 长的块, 相应的输出也为 N 长的块:

$$y_{kN+i} = X_{kN+i}^T W_k \quad 0 \leq i \leq N-1$$

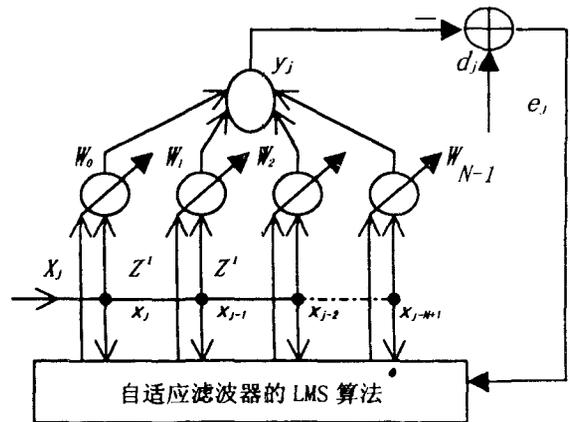


图1 自适应横向滤波器结构

输入 $y(n)$ 的第 k 块实际是输入 $x(n)$ 的第 k 块与第 k 块对应 W_k (在第 k 块内保持不变) 的线性卷积:

$$y_{kN+i} = \sum_{j=0}^{N-1} w_{j,k} x_{kN+i-j} \quad 0 \leq i \leq N-1$$

运用快速 FFT 技术,用循环卷积来计算线性卷积(重叠保留法):

$$\omega_k(K) = \text{FFT} \left[\underbrace{W_k^T, 0, 0, \dots, 0}_{N \uparrow 0} \right] \quad 0 \leq K \leq 2N-1 \tag{1}$$

$$X_k(K) = \text{FFT} \left[\underbrace{x_{(k-1)N}, x_{(k-1)N+1}, x_{(k-1)N+2}, \dots, x_{kN-1}}_{\text{第 } k-1 \text{ 块}}, \underbrace{x_{kN}, x_{kN+1}, x_{kN+2}, \dots, x_{kN+N-1}}_{\text{第 } k \text{ 块}} \right] \tag{2}$$
$$0 \leq K \leq 2N-1$$

$$y'_k(n) = \text{IFFT} [\omega_k(K) \cdot x_k(K)] \quad 0 \leq n \leq 2N-1 \tag{3}$$

$$[y_{kN}, y_{kN+1}, y_{kN+2}, \dots, y_{kN+N-1}] = y'(n) \quad N \leq n \leq 2N-1 \tag{4}$$

另外,权系数是在每 N 个样点经过滤波器后进行一次修正,而且更新量为 N 个输入样点更新量的累加和:

$$W_{k+1} = W_k + 2\mu \sum_{i=0}^{N-1} e_{kN+i} X_{kN+i} = W_k + 2\mu \nabla_k$$

又因为: $\nabla_{k,i} = \sum_{j=0}^{N-1} e_{kN+j} x_{kN+j-i} \quad 0 \leq i \leq N-1$, 它可以看作误差序列与输入序列的互相关函数在 i 处的函数值,其计算可以用线性卷积完成,仍采用快速 FFT 用循环卷积来计算线性卷积:

$$E_k(K) = \text{FFT} \left[\underbrace{0, 0, \dots, 0}_{N \uparrow 0}, d_{kN} - y_{kN}, d_{kN+1} - y_{kN+1}, \dots, d_{kN+N-1} - y_{kN+N-1} \right]$$
$$= \text{FFT} \left[\underbrace{0, 0, \dots, 0}_{N \uparrow 0}, e_{kN}, e_{kN+1}, \dots, e_{kN+N-1} \right] \quad 0 \leq K \leq N-1 \tag{5}$$

$$\nabla'_k(n) = \text{IFFT} (E_k(K) \cdot \text{conj}(x_k(K))) \quad 0 \leq n \leq 2N-1 \tag{6}$$

$$\nabla_k = \nabla'_k(n) \quad 0 \leq n \leq N-1 \tag{7}$$

(1)~(7)为 LMS 算法的频域实现(FILMS)的计算公式。

1.3 FILMS 与 LMS 运算量的理论比较(以实数乘法为标准)

对每 N 点的输入序列, FILMS 算法需要 5 个 $2N$ 点的 FFT (包含 IFFT) 计算和 2 个 $2N$ 点的复数相乘。每个 N 点 FFT 需要 $\frac{N}{2} \log_2 N$ 个复数乘法,对于实输入序列由于 DFT 的对称特性,又可以将运算量减半。对于时域的 LMS 算法,共需要 $2N^2$ 个实数乘法。每个复数乘法按 4 个实数乘法计算,则 FILMS 的运算量与 LMS 的运算量的比值可近似为 $f(N) = (5 \log_2 N + 13) / N$ 。

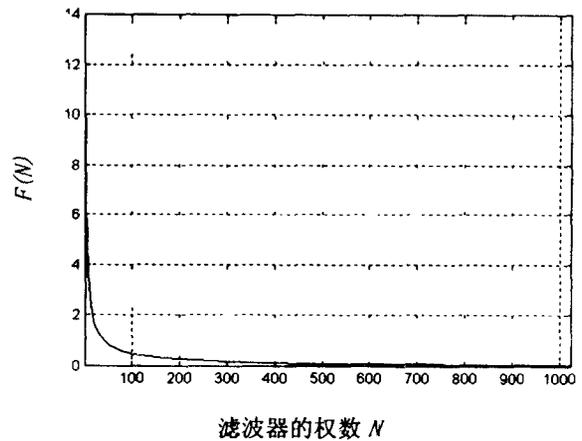


图 2 FILMS 算法与 LMS 算法运算量的比较

2 LMS 算法和 FILMS 算法的 MATLAB 语言实现

2.1 LMS 算法 MATLAB 语言实现

```
function[h,y,e]=lms(x,d,u,N)           M=length(x);
% * * * * * * * * * * * * * * * * * * Y=zeros(1,M);
% * [h,y,e]=lms(x,d,u,N)             * h=zeros(1,N);
% * 自适应滤波器的 LMS 算法子函数    * e=zeros(1,M);
% * h 为估计的 FIR 滤波器            * for n=N:M
% * y 为滤波器的输出序列             *     xb=x(n:-1:n-N+1);
% * e 为误差序列                     *     y(n)=h * xb';
```

```

% * x 为输入的参考序列          *   e(n)=d(n)-y(n);
% * d 为滤波器输出的期望序列    *   h=h+2*u*e(n)*xb;
% * u 为步长                      *   end
% * N 为滤波器的阶数            *   M=length(x);
% * * * * *                      *   y=zeros(1,M);
2.2 FLMS 算法 MATLAB 语言实现    *   h=zeros(1,N);
function[h,y,e]=flms(x,d,u,N)      *   e=zeros(1,M);
% * * * * *                      *   for i=1:fix(M/N)-1
% * [h,y,e]=flms(x,d,u,N)        *   x=fft(x((i-1)*N+1:(i+1)*N));
% * 自适应滤波器的 FLMS 算法子函数 *   H=fft([h,zeros(1,N)]);
% * h 为估计的 FIR 滤波器        *   O1=real(ifft(X.*H));
% * y 为滤波器的输出序列        *   y(i*N+1:(i+1)*N)=O1(N+1:2*N);
% * e 为误差序列                *   e(i*N+1:(i+1)*N)=d(i*N+1:(i+1)*N)-y(i*
% * x 为输入的参考序列          *   N+1:(i+1)*N);
% * d 为滤波器输出的期望序列    *   E=fft([zeros(1,N),e(i*N+1:(i+1)*N)]);
% * u 为步长                      *   O2=real(ifft(E.*conj(X)));
% * N 为滤波器的阶数            *   V=O2(1:N);
% * * * * *                      *   h=h+2*u*V;
% * * * * *                      *
% * * * * *                      *   end

```

3 计算机仿真结果

3.1 将 FLMS 和 LMS 算法用在自适应干扰抵消器中比较计算速度和收敛速度

3.1.1 通过微机的运行时间对运算量进行的估计 (在奔腾 200MHZ 主频上运行)

用 MATLAB 内置的秒表粗略计算运行时间:

```

>>tic;[h,y,e]=flms(noise,nspeech,0.0006,64);toc;
elapsed_time=
    12.5200

```

```

>>tic;[h,y,e]=lms(noise,nspeech,0.0006,64);toc;
elapsed_time=
    34.8200

```

3.1.2 计算机仿真图形显示

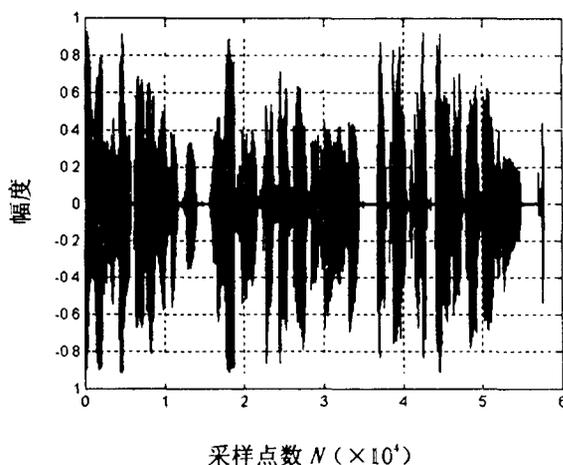


图3 纯净语音信号波形

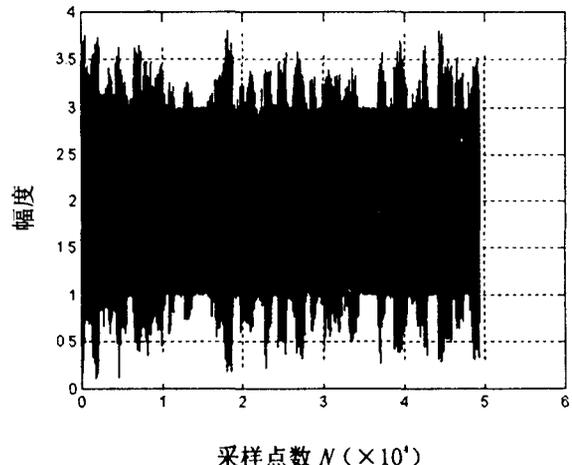


图4 带噪声语音信号波形

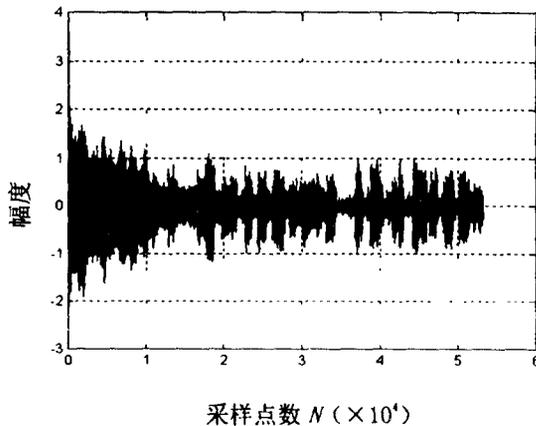


图5 FLMS算法去噪后的语音信号波形

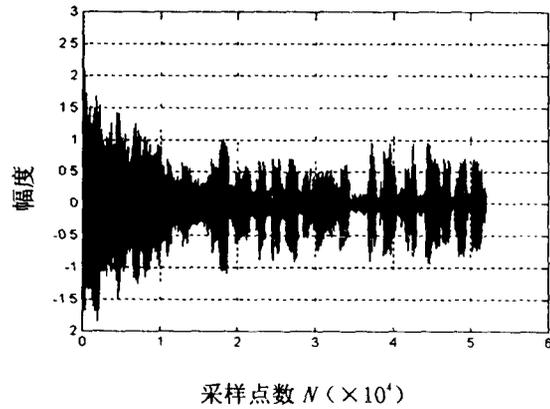


图6 LMS算法去噪后的语音信号波形

3.2 实验的结论

实验中采用近7s的纯净语音混入高斯白噪声,两种算法大约在一秒左右同时收敛,只是LMS算法用时较长。通过实验可得如下结论:

- FLMS算法较LMS算法运算量权数在大于64时有大幅度的减少。
- FLMS算法在收敛的情况下,与LMS算法的收敛速度相近,但在起始阶段的误差较大。
- FLMS算法的局限性和可能的应用方案:

FLMS算法在权数较大时显示出了其运算量小的优点,但是由于其对权数的调整是 N 个样点累加的结果,每次的调整量很大,尤其在权向量的初始值为全0的情况下,若步长选取过大,特别容易导致算法的发散。在权数选取较大的实际应用中应综合LMS算法收敛平稳和FLMS算法计算量小的特点,在收敛算法的起始阶段考虑使用LMS算法,而在算法收敛平稳时,考虑用FLMS算法来减小运算量。可以设置一个误差平均能量的阈值,当误差平均能量超过该阈值时用LMS算法;而在低于该阈值时,可以考虑选用FLMS算法用较少的运算量来进行维持平稳状态。

参 考 文 献

- [1] Ferrara E R. Fast implementation of LMS adaptive filter[J]. IEEE Trans Acoust, Speech, Signal Processing, 1980, 28(4): 474 - 475.
- [2] [美]Widrow B, Stearns S D. Adaptive Signal Processing[M]. Prentice-Hall, Inc, 1985.

The Appliance of LMS Algorithm in Frequency Domain for Eliminating Noise

WANG Bu-hong, GUO Ying

(The Telecommunication Engineering Institute, AFEU., Xi'an 710077, China)

Abstract: A replacement algorithm of LMS by the algorithm in frequency domain (FLMS) was derived in detail. Computer simulation of adaptive noise canceller with FLMS shows that it requires less computation than conventional LMS algorithm when the filter length equals or exceeds 64, while the convergence rate of FLMS is nearly the same as that of LMS algorithm. The limitation and possible application in practice of FLMS are also discussed.

Key words: adaptive filter; algorithm; frequency domain; LMS