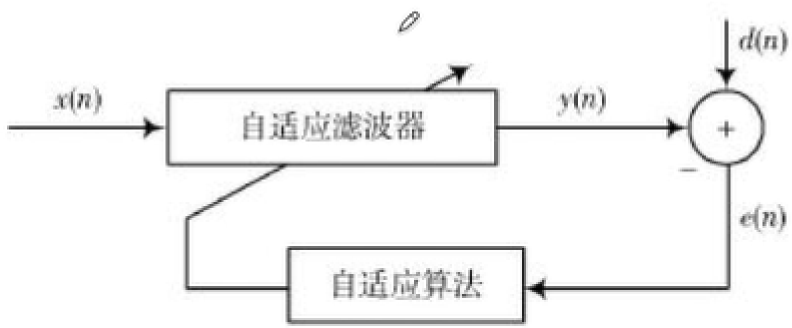
自适应滤波方法

LMS 算法

许多相关的最小均方算法（MMSE）均可以视为LMS算法族中的一种。

自适应滤波器：可以根据输入信号自动调整自身参数的数字滤波器，对于一些应用（噪声消除），我们没办法事先知道需要操作的参数，必须使用自适应系数进行处理。在处理语音信号时，我们不需要事先知道输入语音信号和噪声的统计特性，但是滤波器自身可以在工作过程中学习或估计信号的统计特性，并以此为依据调整自身参数，以达到某个代价函数下最优滤波的效果。若信号统计特性发生变化，还可以跟踪这种变化，重新调节参数，使滤波性能重新达到最优。



自适应滤波器的一般结构

现在来介绍LMS算法，先从一个N阶线性系统出发：

假设有一个N阶滤波器，其参数为$w(n)$，那么滤波器输出为：

$$y(n)=\sum_{i=0}^{N-1} w_i(n)x(n-i)=\mathbf{w}^T(n)\mathbf{x}(n)=\mathbf{x}^T(n)\mathbf{w}(n)$$

其中：

$$\mathbf{x}(n)=[x(n),\quad x(n-1),\quad \ldots\quad x(n-N+1)]^T$$
$$\mathbf{w}(n)=[w_0(n),\quad w_1(n),\quad \ldots\quad w_{N-1}(n)]^T$$

定义期望输出为$d(n)$，那么误差信号就为：

$$e(n)=d(n)-y(n)=d(n)-\mathbf{w}^T(n)\mathbf{x}(n)$$

那么自然而然我们期望经过这个滤波器后的输出信号尽可能地接近我们的期望输出，那么也就是要让误差信号的均方误差最小（MMSE），所以我们定义最小化目标函数为：$J(w)$

$$J(\mathbf{w})=E\left\{|e(n)|^2\right\}=E\left\{\left|d(n)-\mathbf{w}^T(n)\mathbf{x}(n)\right|^2\right\}$$

然后对目标函数关于w进行求导有，并令为0，有：（第一项包含了输入信号自相关矩阵的信息，第二个是输入信号x和输出信号d的相关向量）

$$E\left\{\mathbf{x}(n)\mathbf{x}^T(n)\right\}\mathbf{w}(n) - E\left\{\mathbf{x}(n)d(n)\right\} = \mathbf{R}\mathbf{w} - \mathbf{r} = \mathbf{0}$$

那么之后就可以得到:

$$\boxed{\mathbf{w}_{opt} = \mathbf{R}^{-1}\mathbf{r}}$$

那么通过这个式子定义滤波器系数的滤波器就称为Wiener滤波器，那么也就可以说Wiener滤波器时均方误差最小意义上的统计最优滤波器。（其实它不适用于非平稳信号，语音信号不适用），这里可以看到我们对此时最优滤波器的估计是基于全部的历史数据，如果有一个新数据要放入的话，就有重新计算全部的历史数据。所以实际应用中，我们一般得不到这两个统计参数，所以不显式的使用这个公式，反而我们采用自适应的方式，逐步更新滤波器系数w，梯度下降，逐步接近最优滤波器系数。

我们采用瞬时梯度，将目标函数的数学期望换成瞬时误差平方来作为目标函数。

$$\hat{\nabla}(n) = -2e(n)\mathbf{x}(n)$$

所以我们就可以得到标准时域下的LMS算法的更新公式：（基于采样点更新）

$$\boxed{\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu\mathbf{x}(n)e(n)}$$

标准LMS算法的执行流程：

1) 滤波器初始化：$\mathbf{w}(0)$ 、$\mathbf{x}(0)$

2) 对每一个新的输入采样 $x(n)$ ，计算输出信号 $y(n)$

3) 利用期望输出 $d(n)$ ，根据等式(3.4)计算误差信号 $e(n)$ ，得到梯度(3.8)式。

4) 利用等式(3.9)更新滤波器系数：$\mathbf{w}(n)$

5) 返回步骤 2)，直至结束，可以得到输出序列和误差序列。

LMS算法优点：算法简单，容易实现。缺点：收敛速度慢

LMS算法的改进思路：Block LMS， 也就是不对每个采样点进行逐点更新，而是累计一些采样点作为一个block，然后对每个block进行更新。梯度噪声会变小。

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu\mathbf{x}(n)e(n)$$

$$\mathbf{w}(n+2) = \mathbf{w}(n+1) + 2\mu\mathbf{x}(n+1)e(n+1)$$

...

$$\mathbf{w}(n+L) = \mathbf{w}(n+L-1) + 2\mu\mathbf{x}(n+L-1)e(n+L-1)$$

最后可以得到：时域LMS的分块更新公式，每L个点更新一次

$$\boxed{\mathbf{w}(n+L) = \mathbf{w}(n) + 2\mu\sum_{m=0}^{L-1}\mathbf{x}(n+m)e(n+m)}$$

注意，(3.10)中的误差信号是通过同一个滤波器 $\mathbf{w}(n)$ 得到的：

$$e(n+m) = d(n+m) - \mathbf{x}^T(n+m)\mathbf{w}(n)$$

令$kL = n$，可以得到:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu \sum_{m=0}^{L-1} \mathbf{x}(kL+m)e(kL+m)$$

那么block LMS 的方法的两个核心运算:

- 输入向量与滤波器系数向量的线性卷积 $y(n+m) = \mathbf{x}^T(n+m)\mathbf{w}(n)$
    - 计算线性卷积，我们采用FFT计算，其中有overlap-save和overlap-add的两种快速计算卷积的方法
        - 为了得到 N 点的（线性卷积后的）输出信号：

          $$y(n+m) = \mathbf{x}^T(n+m)\mathbf{w}(n) \qquad m = 0, 1, ..., N-1$$

          要保证至少有 N 个点的线性卷积和圆周卷积的结果相同：

          $$N_1 - N_2 + 1 = N$$

          由于 $N_1 \geqslant N_2$（也就是输入信号长度通常大于滤波器的阶数），且 $N_2 = N$ (滤波器阶数为N)，那么要求每次参与运算的输入信号长度 N_1 至少为 $2N-1$

          为了计算FFT方便，我们令输入信号的长度：

          $$N_1 = 2N$$

          显然，FFT的长度为2N

        那么怎么构造长度为2N的数据，把输入信号分为两个长度为N的blocks,然后将滤波器系数向量长度扩展为长度为2N的向量

        之后分别计算输入信号向量和滤波器系数向量的傅里叶变换

        $$\mathbf{X}(k) = diag\left\{\mathbf{F}\left[x(kN-N),\ ...,x(kN-1),\ x(kN),\ ...,x(kN+N-1))\right]\right\}$$

        $$\mathbf{W}(k) = \mathbf{F}\left[\mathbf{w}^T(k),\ 0,\ ...,0\right]^T$$

        之后直接在频率相乘: $\mathbf{Y}(k) = \mathbf{X}(k)\mathbf{W}(k)$ （此时输出信号频域表示长度也就是2N）

        我们对$Y(k)$ 做傅里叶逆变换后，取后N个点，那么就得到了最终想要得到的两个信号的线性卷积的结果。

下面便是Overlap Save方法的框架图：（这个方法不仅仅适用于频域自适应滤波，对于滤波器系数固定不变的情况依然适用，如语音加混响，与固定系数的房间冲激响应之间的线性卷积，我们也可以使用overlap-save方法实现）

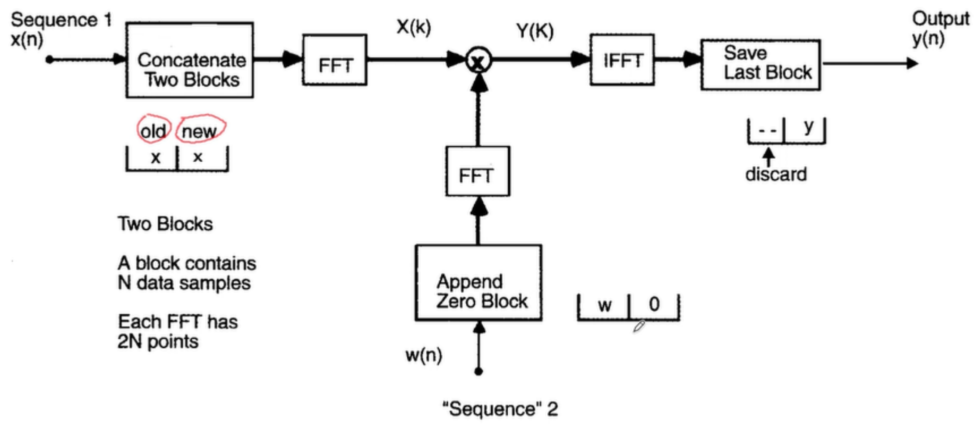


Figure: overlap-save sectioning method

- 误差信号与输入向量的线性相关： $$\hat{\bigtriangledown}(k) = -2\sum_{m=0}^{L-1}\mathbf{x}(kL+m)e(kL+m)$$
  - 其实思路与之前的卷积操作一致，我们考虑想办法把他们变换到频域，然后计算输入信号的共轭谱与误差信号谱的乘积，而输入信号的频域表示在第一步中已经被计算出来，所以我们只需要先将误差向量扩展到2N长度。（计算卷积的时候，我们在滤波器系数向量后面补上N个零，而对于计算相关的时候，我们在误差向量前面补N个零）

    补零后：对误差向量做傅里叶变换：

$$\mathbf{E}(k) = \mathbf{F}\left[0,\ 0,\ \ldots,0,\ \mathbf{e}^T(k)\right]^T$$

然后在频域内相乘： $$\boldsymbol{\Delta}(k) = \mathbf{X}^H(k)\mathbf{E}(k)$$

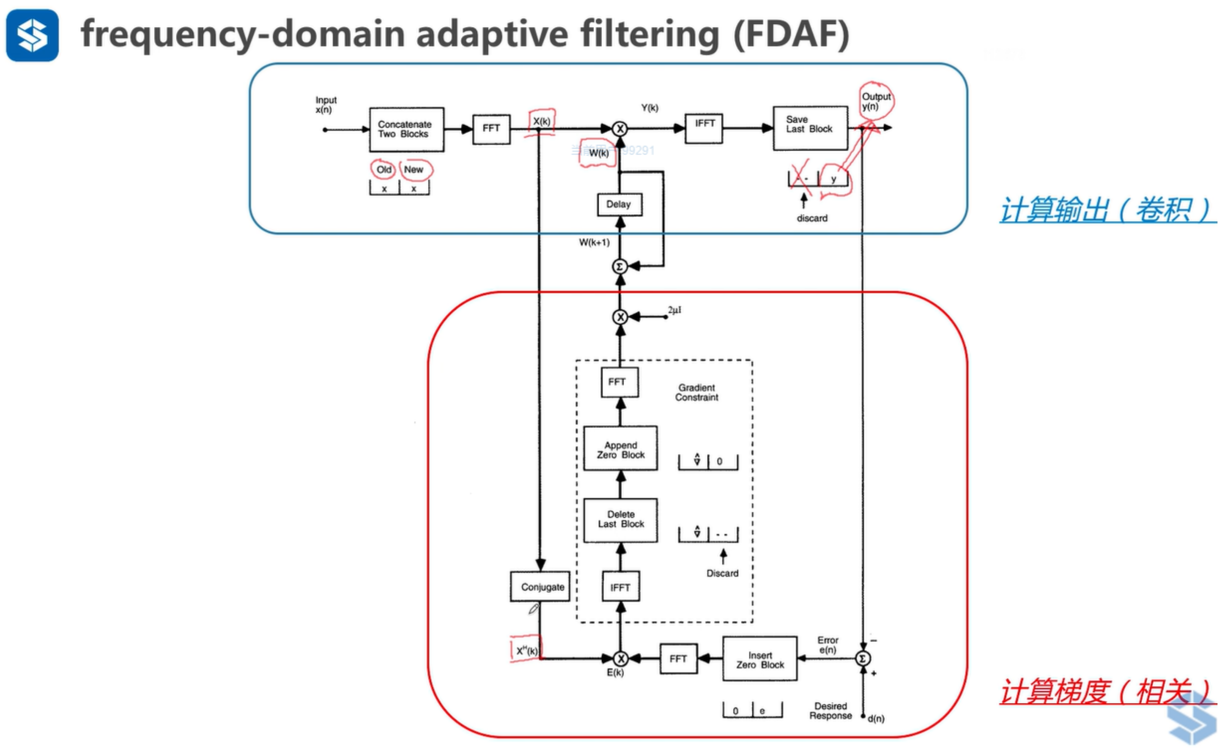
现在梯度向量长度也是2N，所以我们只需要$\triangle(k)$ 的傅里叶逆变换的前N个点。

- 滤波器系数更新：

$$\boxed{\mathbf{W}(k+1) = \mathbf{W}(k)+2\mu\mathbf{F}\left[\vec{\bigtriangledown}^T(k),\ 0,\ \ 0,\ \ldots,0\right]^T}$$

注意：滤波器系数直接在频域更新，所以要将刚才得到时域上长度为N的梯度向量再次变换到频域内，由于滤波器系数向量在后面补了N个零，所以梯度向量也要在后面补N个零。

那么整体框架就是：



LMS算法对输入信号的要求：要求不同时刻的输入向量$x(n)$线性无关——LMS的独立性假设，如果输入信号之间存在相关性，会导致前一次迭代产生的噪声梯度传播至下一次迭代，造成误差的反复传播，收敛速度变慢，跟踪性能变差。LMS算法对白噪声的效果最好。

NLMS:归一化LMS：

$$\mathbf{w}(n+1)=\mathbf{w}(n)+2\mu\frac{\mathbf{x}(n)}{\mathbf{x}^T(n)\mathbf{x}(n)}e(n)$$

与标准的LMS相比，在更新步长mu上（梯度项）增加了一个归一化项，也就是输入信号的功率。（归一化是为了：平滑掉较大的输入从而导致的噪声梯度的放大）

那么学习速率如何选取：对m个频率bin，利用输入信号在这个频点处的功率$P_m(k)$对学习速率进行归一化：

$$\mu_m(k)=\frac{\mu}{P_m(k)}$$

而频点的功率$P_m(k)$通常由迭代的方法求得：

$$P_m(k)=\lambda P_m(k-1)+(1-\lambda)\left|X_m(k)\right|^2$$