

## 自适应滤波

### 5.1 自适应滤波基本原理

所谓自适应,从通俗的意义上讲,就是这种滤波器能够根据输入信号统计特性的变化自动调整其结构参数,以满足某种最佳准则的要求。自适应滤波采用的最佳准则有最小均方误差(LMS)准则、最小二乘(LS)准则、最大信噪比准则等,其中应用最广泛的是 LMS 准则。

自适应滤波在通信、控制、语音分析和合成、地震信号处理、雷达和声呐波束形成以及医学诊断等诸多科学领域有着广泛的应用,正是这些应用推动了自适应滤波理论和技术的发展。本章将对一些典型的自适应滤波器应用实例进行介绍。

自适应滤波器一般由参数可调的数字滤波器和自适应算法两部分构成,如图 5.1 所示。

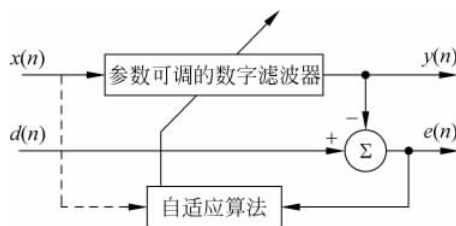


图 5.1 自适应滤波器原理图(闭环系统)

输入信号  $x(n)$  通过参数可调的数字滤波器后产生输出信号  $y(n)$ , 将其与参考信号(或者期望响应)  $d(n)$  进行比较, 形成误差信号  $e(n)$ 。通过预设的自适应算法对滤波器参数进行调整, 以满足某种最佳准则为目的(如使  $e(n)$  的均方值最小)。由于图 5.1 的自适应滤波器存在反馈机制, 因此这是一种闭环系统。

自适应滤波器实际上是一种可以自动调整自身参数的特殊维纳滤波器, 在设计时不需要事先知道关于输入信号和噪声统计特性的先验知识, 它能够在自己的工作过程中逐渐“了解”或估计出所需的统计特性, 并以此为依据自动调整自己的参数, 以达到最佳滤波的目的。一旦输入信号的统计特性发生变化, 它又能自动跟踪这种变化, 自动调整自身参数, 使滤波器的性能重新达到最佳。所以, 自适应滤波器在输入过程的统计特性未知时, 自适应滤波器

调整自己参数的过程称作“学习”过程；当输入过程的统计特性变化时，自适应滤波器调整自己参数的过程称作“跟踪”过程。

最小均方误差(LMS)自适应滤波器与递推最小二乘(RLS)自适应滤波器是两种最常用的自适应滤波器，由于它们采用的最佳准则不一样，因此这两种自适应滤波器在原理、性能等方面均有较大差别，本章接下来将主要介绍这两种自适应滤波器。

## 5.2 最小均方误差自适应滤波

最小均方误差(Least Mean Square, LMS)滤波的准则在于，使得滤波器的输出与期望信号误差的平方统计平均值最小<sup>[1,2]</sup>。LMS 自适应横向滤波器原理图如图 5.2 所示。

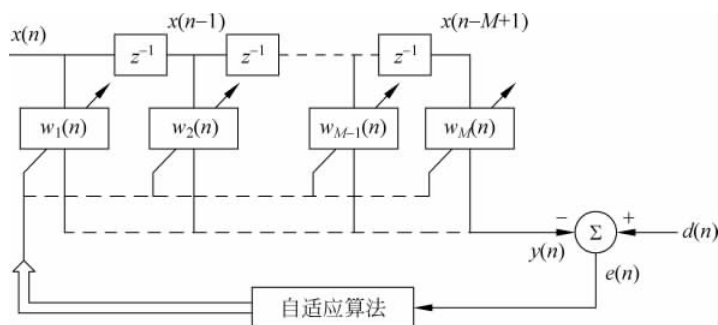


图 5.2 LMS 自适应横向滤波器原理图

该自适应滤波器的输入矢量为

$$\mathbf{X}(n) = [x(n), x(n-1), \dots, x(n-M+1)]^T$$

加权矢量(即滤波器参数矢量)为

$$\mathbf{W}(n) = [w_1(n), w_2(n), \dots, w_M(n)]^T$$

滤波器的输出为

$$y(n) = \sum_{i=1}^M w_i(n)x(n-i+1) = \mathbf{W}^T(n)\mathbf{X}(n) = \mathbf{X}^T(n)\mathbf{W}(n)$$

$y(n)$  相对于滤波器期望输出  $d(n)$  的误差为

$$e(n) = d(n) - y(n) = d(n) - \mathbf{W}^T(n)\mathbf{X}(n)$$

根据最小均方误差准则，最佳的滤波器参量应使得均方误差最小，即

$$\min\{E[e^2(n)]\}$$

最小均方误差算法是一种简单实用的方法，其突出特点就是计算量小，易于实现，而且不要求离线计算。其关键技术在于按照  $e(n)$  及各  $x(n)$  值，通过某种算法，确定  $E[e^2(n)]$  最小时各个  $w^*(n)$  的取值，从而自动调节各  $w(n)$  值至  $w^*(n)$ 。

LMS 自适应横向滤波器具有 FIR 滤波器的结构，虽然有简单和容易实现的优点，但也存在着计算量大的缺点，在实际应用中常需要采用阶数很高的 FIR 滤波器。如果此时改用 IIR 递推结构，滤波器阶数可以明显降低，而且在某些应用中必须采用 IIR 滤波器。

IIR 自适应滤波器的主要优点是可以大幅降低运算量，并且具有谐振和锐截止特性，但它也有两个主要不足：第一是由于递归结构存在着反馈支路，故在自适应过程中有可能使

极点移到单位圆外从而使滤波器失去稳定性；第二是它的性能曲面一般高于二次的曲面，有可能存在一些局部最小值，使得搜索全局极小值的工作变得复杂和困难。为了克服第一个缺点，必须采取措施限制滤波器参数的取值范围；对于第二个缺点，则必须寻找更好的自适应算法，以便在复杂的性能曲面上能够正确地搜索全局最低点<sup>[1]</sup>。

图 5.3 所示为 LMS 自适应 IIR 滤波器原理图，图中输入信号矢量  $\mathbf{x}(n)$  可以是单输入的，也可以是多输入的。

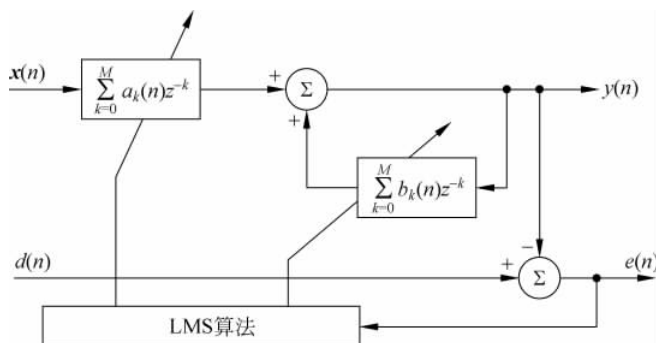


图 5.3 LMS 自适应 IIR 滤波器原理图

单输入情况下，IIR 滤波器的差分方程为

$$y(n) = \sum_{i=0}^M a_i(n)x(n-i) + \sum_{i=1}^M b_i(n)y(n-i)$$

误差为

$$e(n) = d(n) - y(n) = d(n) - \mathbf{W}^T(n)\mathbf{U}(n) = d(n) - \mathbf{U}^T(n)\mathbf{W}(n)$$

其中，复合权矢量  $\mathbf{W}(n)$  和复合数据矢量  $\mathbf{U}(n)$  分别定义为

$$\mathbf{W}(n) = [a_0(n), a_1(n), \dots, a_M(n), b_1(n), \dots, b_M(n)]^T$$

$$\mathbf{U}(n) = [x(n), x(n-1), \dots, x(n-M), y(n-1), \dots, y(n-M)]^T$$

除此之外，还存在 LMS 自适应格型滤波器。这种格型结构中的反射系数可以根据输入数据直接进行计算，有迭代形式和非迭代形式两种算法，有兴趣的读者可以参阅有关文献<sup>[1]</sup>。

## 5.3 递归最小二乘自适应滤波

LMS 自适应横向滤波器和 LMS 自适应 IIR 滤波器等，根据的最佳准则都是最小均方误差(LMS)准则，即使得滤波器输出与期望信号误差的平方的统计平均值最小。LMS 准则根据输入数据的长期统计特性寻求最佳滤波。但是，通常已知的仅为一组数据，只能对长期统计特性进行估计或近似，LMS 算法、格型梯度算法等均是如此。在此介绍的最小二乘(Least Square, LS)算法，直接根据一组数据寻求最佳，主要包括 RLS 自适应横向滤波器、RLS 自适应格型滤波器和快速横向滤波自适应滤波器等。本章主要介绍 LMS 自适应横向滤波器。

假设输入信号为  $x(1), x(2), \dots, x(N)$ ，将其加到一个  $M$  阶横向滤波器的输入端，其输出  $y(n)$  满足  $y(n) = x(n) * w(n)$ ，假如滤波器的期望响应为  $d(n)$ ，则可得误差分量为

$$e(n) = d(n) - x(n) * w(n)$$

最小二乘算法是在满足误差测度函数  $\epsilon(\mathbf{W})$  取最小值条件下, 求解出最佳的滤波器权系数  $\mathbf{W}_{\text{opt}}$ , 这就是最小二乘的基本原理。

$$\min[\epsilon(\mathbf{W})] = \min\left(\sum_{n=i_1}^{i_2} |e(n)|^2\right)$$

这里所讨论的最小二乘滤波器实际上是一个确定性的最小二乘滤波器, 求和的上下限  $i_1$  和  $i_2$  取决于不同的加窗形式。一般来说, 输入数据的加窗方式有协方差法、自相关法、前加窗法和后加窗法等<sup>[1]</sup>。

RLS 滤波器算法是一种递推的最小二乘算法,  $n$  时刻的滤波器参数可以在  $n-1$  时刻滤波器参数的基础上, 根据  $n$  时刻到了的输入数据  $x(n)$  进行更新, 因此观察到的数据长度是可变的, 为此将误差测度函数写成数据长度  $n$  的函数  $\epsilon(n)$ 。另外, 习惯加入一个加权因子 (又称遗忘因子) 到误差测度函数的定义中, 使之成为

$$\epsilon(n) = \sum_{i=1}^n \lambda^{n-i} |e(i)|^2$$

其中,  $0 < \lambda \leq 1$ 。引入加权因子  $\lambda^{n-i}$  的目的是赋予老数据与新数据不同的权值, 以使自适应滤波器具有对输入过程特性变化的快速反应能力。RLS 自适应横向滤波器原理的具体推导过程和算法流程, 有兴趣的读者可以参阅有关文献<sup>[1]</sup>。

LMS 滤波器与 RLS 滤波器是两种不同工作准则的滤波器。LMS 算法是一种有效而简便的方法, 其优点是结构简单, 算法复杂度低, 易于实现, 稳定性高, 然而 LMS 算法对于快速变化的信号并不适用, 因为它的收敛速度较慢。RLS 算法是另外一种基于最小二乘准则的精确方法, 它具有快速收敛和稳定的回波抵消器特性, 因而被广泛地应用于实时系统识别和快速启动的信道均衡等领域。

## 5.4 自适应滤波器的应用

许多实际问题无法采用固定的数字滤波器很好地解决, 这是因为在设计固定系数滤波器时缺乏足够的信息, 或者是在运算过程中输入数据特性发生改变导致预先设计的滤波器准则欠妥。这些问题都可以用自适应滤波器来较好地解决, 这是因为自适应滤波器的显著特点就是在运算过程中, 在无须人工干预的情况下也能修改自身的响应, 从而快速适应外界的变化, 提高滤波器的性能。自适应滤波器解决的实际应用问题非常广泛, 包括通信中的回波抵消、数据通信中的信道均衡、线性预测编码、噪声抵消等<sup>[3]</sup>。下面通过几个例子介绍自适应滤波器的应用。

### 5.4.1 自适应干扰抵消

图 5.4 所示为自适应干扰抵消器的基本原理图, 期望信号  $d(n)$  是有用信号  $s(n)$  与干扰  $N_1(n)$  之和, 即  $d(n) = s(n) + N_1(n)$ ,  $N_2(n)$  是与  $N_1(n)$  相关的另一个干扰, 自适应滤波器将调整自己的参数, 以使其输出  $y(n)$  成为  $N_1(n)$  的最佳估计  $\hat{N}_1(n)$ , 误差  $e(n)$  即为对有用信号的最佳估计, 干扰  $N_1(n)$  就得到了一定

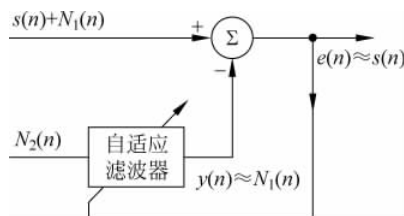


图 5.4 自适应干扰抵消器的基本原理框图

程度的抵消。

在具体应用中,还有两个实际问题需要考虑:

(1) 由于  $N_2(n)$  与  $N_1(n)$  相关,因而  $N_1(n)$  能被很好地抵消。但若另外还有与  $N_1(n)$  不相关的干扰  $N_3(n)$  叠加在  $s(n)$  上,则  $N_3(n)$  无法被抵消。

(2) 若由于信号  $s(n)$  输入自适应滤波器的输入端,则有用信号也将有一部分被抵消。因此,应尽可能地避免信号漏入自适应滤波器的输入端。

自适应干扰抵消有着广泛应用,例如在胎心监测中用于抵消胎儿心电图中的母亲的心音。将从母亲腹部取得的信号加在参考输入端,它是胎儿心音与母亲心音的叠加。将从母亲腹部取得的信号加在自适应滤波器输入端,系统输出的将是胎儿心音的最佳估计。下面通过一个类似的例子进行演示。

**例 5.1** 在鸟叫声 chirp 中混入高斯白噪声,请采用自适应的 LMS 算法提取纯净的鸟叫声信号。

**解:** chirp.mat 一般存放于 \toolbox\compiler\mcr\matlab\audiovideo 路径下,是 MATLAB 自带的一个数据文件,例题的 MATLAB 源代码如下(见 chp5sec5\_1.m),实验结果如图 5.5 所示。

```
% 采用 LMS 自适应滤波器滤除 chirp 信号中的高斯白噪声 chp5sec5_1.m
clear;
clc;
load('chirp','Fs','y');
s = y;
N = length(s);
var = 1;
n0 = sqrt(var) * randn(N,1); % 高斯白噪声方差
                                % 零均值, 高斯白噪声
nfilt = fir1(3,0.5);
n1 = filter(nfilt,1,n0); % 接收到的噪声 n1 与 n0 相关
d = s + n1; % 接收到的语音信号,s 被 n1 污染

% LMS 自适应滤波
M = 32;
step = 0.01; % LMS step size.
lk = 1;
W0 = zeros(M,1);
Zi = zeros(M-1,1);
Hadapt = adaptfilt.lms(M, step, lk, W0, Zi);
[y,e] = filter(Hadapt,n0,d); % y 是对 n1 的估计, 误差 e = d - y 是对原始信号 s 的估计

subplot(3,1,1);plot(s);axis([1,N,-2,2]);title('原始语音信号');grid on;
subplot(3,1,2);plot(d);axis([1,N,-3,3]);title('观测到的信号');grid on;
subplot(3,1,3);plot(e);axis([1,N,-2,2]);title('恢复后的信号');grid on;
% sound(s,Fs); % 接上外接设备就可以听到声音效果
```

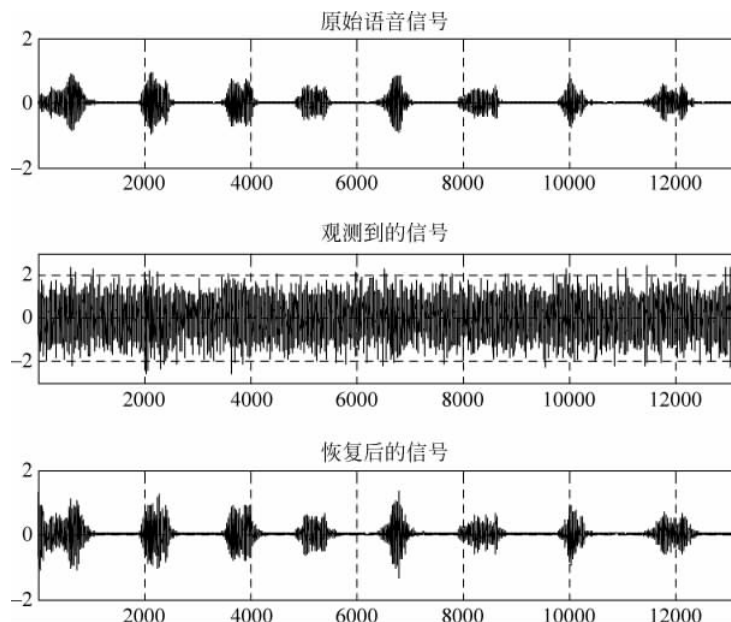


图 5.5 利用自适应 LMS 算法恢复被噪声污染的鸟叫声(chirp)

**例 5.2** 与例 5.1 类似,请采用自适应的 RLS 算法提取纯净的鸟叫声信号。

**解:** 将自适应滤波器由 `adaptfilt.lms` 改为 `adaptfilt.rls`,并对输入参数进行必要的修改,例题的 MATLAB 源代码如下(见 `chp5sec5_2.m`),实验结果如图 5.6 所示。

```
% 采用 RLS 自适应滤波器滤除 chirp 信号中的高斯白噪声 chp5sec5_2.m
clear;
clc;
load('chirp','Fs','y');
s = y;
N = length(s);
var = 1;
n0 = sqrt(var) * randn(N,1); % 高斯白噪声方差
nfilt = fir1(3,0.5); % 零均值,高斯白噪声
n1 = filter(nfilt,1,n0); % 接收到的噪声 n1 与 n0 相关
d = s + n1; % 接收到的语音信号, s 被 n1 污染

% RLS 自适应滤波
M = 32;
lambda = 0.99; % RLS 算法遗忘因子
P0 = 10 * eye(M); % 输入信号协方差的逆矩阵
W0 = zeros(M,1);
Zi = zeros(M-1,1);
Hadapt = adaptfilt.rls(M,lambda,P0,W0,Zi);
[y,e] = filter(Hadapt,n0,d); % y 是对 n1 的估计,误差 e = d - y 是对原始信号 s 的估计

subplot(3,1,1);plot(s);axis([1,N,-2,2]);title('原始语音信号');grid on;
subplot(3,1,2);plot(d);axis([1,N,-3,3]);title('观测到的信号');grid on;
subplot(3,1,3);plot(e);axis([1,N,-2,2]);title('恢复后的信号');grid on;
% sound(s,Fs); % 接上外接设备就可以听到声音效果
```

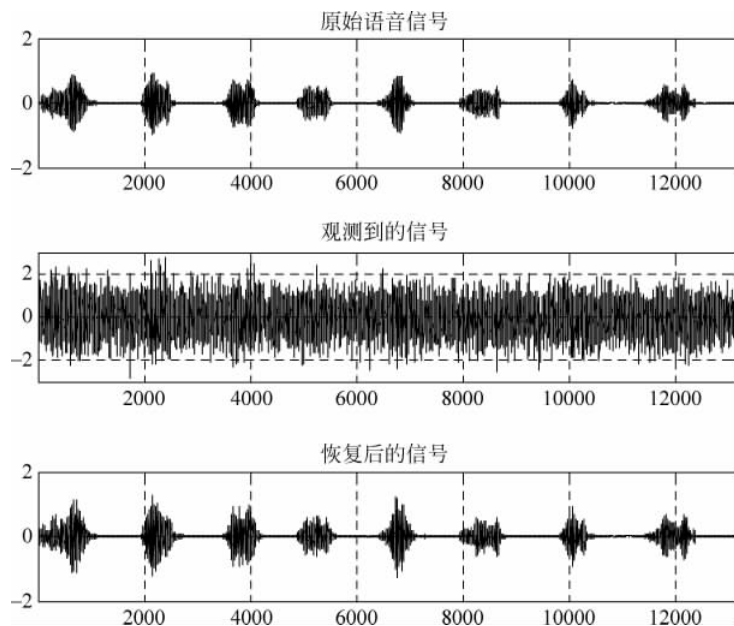


图 5.6 利用自适应 RLS 算法恢复被噪声污染的鸟叫声(chirp)

### 5.4.2 自适应预测

若将自适应干扰抵消器中的输入信号替换为有用信号的时延,则构成自适应预测器,其原理如图 5.7 所示。当完成自适应调整后,将自适应滤波器的参数复制移植到预测滤波器上,那么后者的输出便是对有用信号的预测,预测时间与时延时间相等。

自适应预测的应用之一是分离窄带信号和宽带信号。在图 5.7 所示的自适应预测器中,若在 A 端加入的是一个窄带信号  $s_N(n)$  和一个宽带信号  $s_B(n)$  的混合,窄带信号的自相关函数  $R_N(k)$  比宽带信号的自相关函数  $R_B(k)$  的有效带宽要短。当延迟时间选择为  $k_B < \Delta < k_N$  时,信号  $s_B(n)$  与  $s_B(n-\Delta)$  将不再相关,而  $s_N(n)$  与  $s_N(n-\Delta)$  仍然相关,因而自适应滤波器的输出将只是  $s_N(n)$  的最佳估计  $\hat{s}_N(n)$ ,  $s_B(n) + \hat{s}_N(n)$  与  $s_N(n)$  相减后将得到  $s_B(n)$  的最佳估计  $\hat{s}_B(n)$ ,这样就可以把  $s_N(n)$  和  $s_B(n)$  分开。

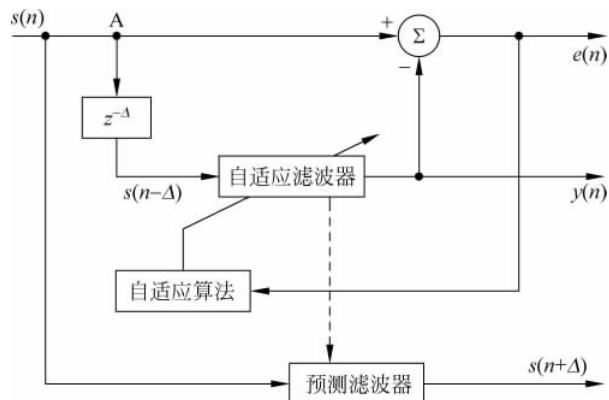


图 5.7 自适应预测器原理框图

如果宽带信号是白噪声,窄带信号是周期信号,则分离后滤波器输出为  $y(n) = \hat{s}_N(n)$ , 即谱线被突出了,这就是所谓的谱线增强。另外,录音磁带中的交流声,留声机转台中的隆隆声等均可利用上述原理进行消除。下面也通过一个类似的例子进行演示。

**例 5.3** 采用自适应滤波器恢复被白噪声污染的单频信号,算法实现流程如图 5.8 所示。

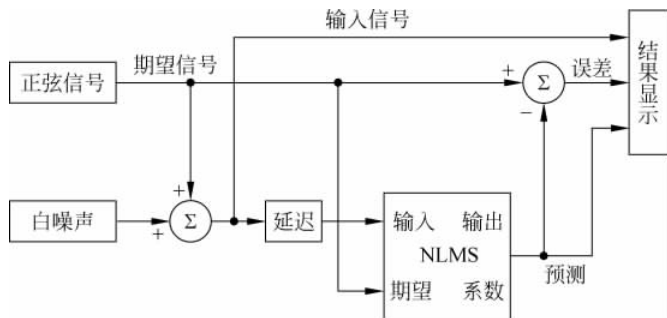


图 5.8 NLMS 算法实现流程图

**解：**在此采用 NLMS(归一化最小二乘)算法进行预测, MATLAB 源代码如下(见 chp5sec5\_3.m), 实验结果如图 5.9~图 5.12 所示。

```

% 采用自适应滤波器恢复白噪声下的正弦信号 chp5sec5_3.m
clear;
clc;
N = 500;
% 输入信号, 两个单频
sig = [sin(2 * pi * 0.015 * (0:N-1)), 0.5 * cos(2 * pi * 0.008 * (0:N-1))];
figure(1); plot(0:2 * N - 1, sig); axis([0, 2 * N, -3 3]);
grid on; title('输入信号');
nvar = 0.5; % 白噪声方差
noise = sqrt(nvar) * randn(1, 2 * N);
n = sig + noise; % 输入信号 = 信号 + 噪声
x = [0 n]; % 线性预测的延时
d = [sig 0]; % 预期输入滤波器的 sig 信号
M = 32; % 滤波器阶数
step = 0.2; % 步长
figure(2); plot(0:2 * N, x); axis([0, 2 * N, -3 3]);
grid on; title('输入自适应滤波器前的信号');

% 通过自适应滤波器预测信号
Hadapt = adaptfilt.nlms(M, step, 1, 1e-6);
[y, e] = filter(Hadapt, x, d);
figure(3); plot(0:2 * N, [d', y']); axis([0, 2 * N, -3 3]);
grid on; title('自适应滤波器输出的信号');
legend('实际信号', '预测信号');

% 预测误差分析
X = xcorr(e(50:end), 'coeff');
[maxX idx] = max(X);
  
```



```
figure(4);plot(X[idx:end]);  
grid on;title('预测信号的自校正');
```

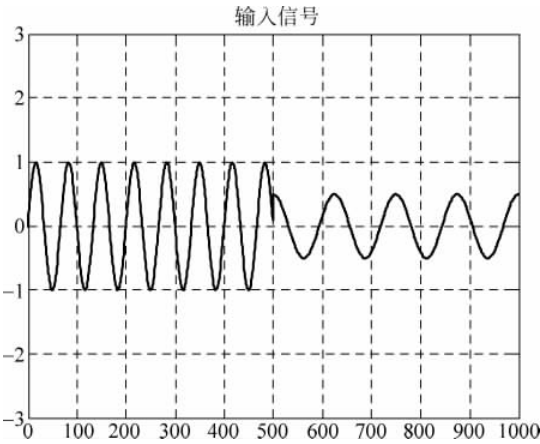


图 5.9 输入信号波形

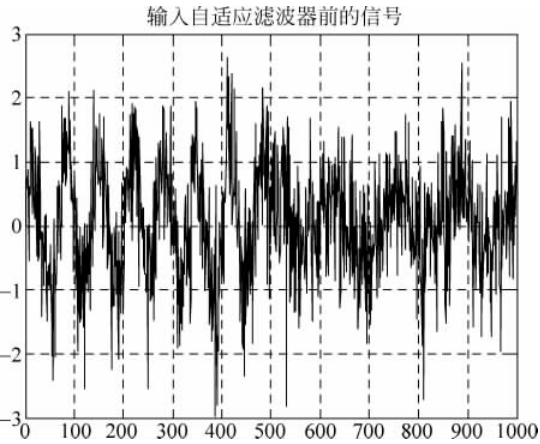


图 5.10 滤波前的信号

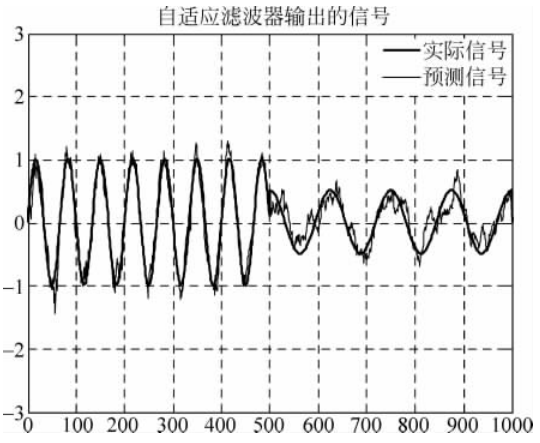


图 5.11 滤波后的结果

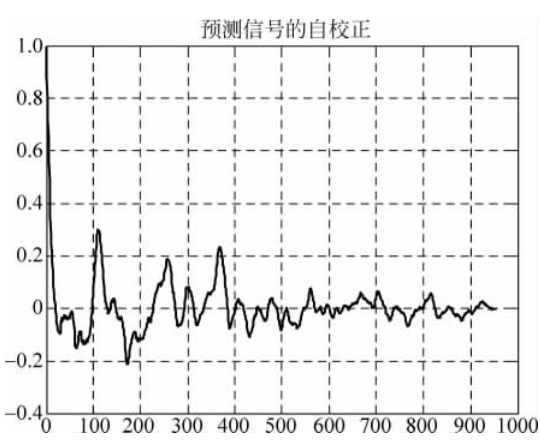


图 5.12 误差分析

5.4.3 自适应信号分离器

参考信号是原始输入的  $k$  步延时的自适应对消器可以组成自适应预测系统、谱线增强系统以及信号分类系统等。自适应信号分离器的原理框图如图 5.13 所示。

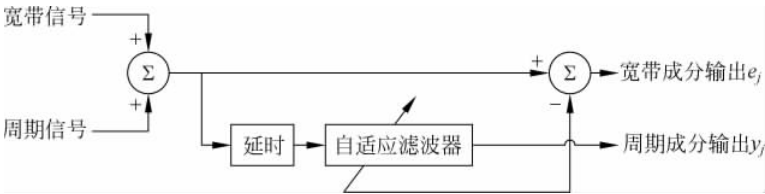


图 5.13 自适应信号分离器原理框图

当输入信号包括两种成分：宽带信号(或噪声)与周期信号(或噪声)时,为了分离两种信号,可以一方面将该输入信号送入  $d_j$  端,另一方面把它延时足够长时间后送入自适应滤波器的输入端。经过延时后宽带成分与原来的输入不相关,而周期成分延时前后保持强相关。于是在  $e_j$  输出中将周期成分抵消只存在宽带成分,在  $y_j$  输出中只存在周期成分,此时自适应滤波器自动调节  $W^*$ ,以达到对周期成分选通的作用。如果将所得到的  $W^*$  利用 FFT 变换成频域特性,则将得到窄带选通的“谐振”特性曲线。该方法可以有效地应用于从白噪声中提取周期信号。下面通过两个例子进行演示。

**例 5.4** 选取信号为正弦信号  $s[n] = \sin(\pi n/5)$ , 宽带噪声信号为高斯白噪声, 时延为  $D=1$ 。请给出收敛因子  $u$  为 0.001 和 0.3 时的滤波效果。

**解:** MATLAB 源代码如下(见 chp5sec5\_4.m), 实验结果如图 5.14 和图 5.15 所示。

```
% 采用自适应信号分离器恢复白噪声下的正弦信号 chp5sec5_4.m
clear;
clc;
N = 100;
n = 0:N-1;
var_noise = 0.3^2; % 白噪声方差
s = sin(2 * pi * n/10);
x = s + sqrt(var_noise) * randn(1,N);
subplot(2,1,1);
plot(n,x); title('自适应滤波器输入'); grid on;

k = 5;
e = 0;
D = 1; % 时延
u = 0.001; % 收敛因子
y = zeros(1,N);
y(1:D+k) = x(1:D+k);
W = 1/k * ones(1,k);
for i = (D+k+1):N
```

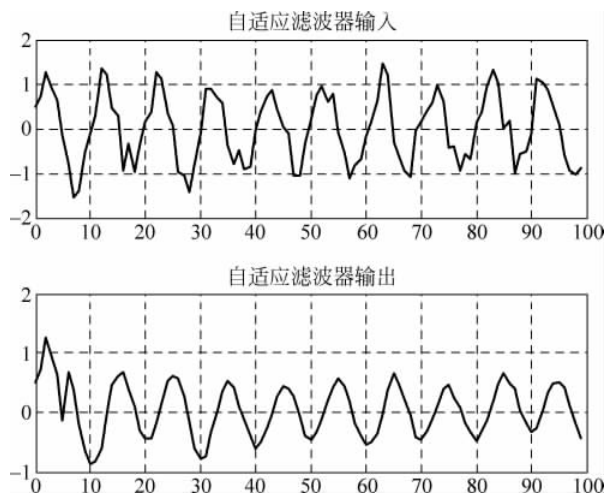


图 5.14 自适应信号分离器输出, 收敛因子为 0.001

```

X = x((i-D-k+1):(i-D));
y(i) = W * X';
e = x(i) - y(i);
W = W + 2 * u * e * X;
end
subplot(2,1,2);
plot(n,y); title('自适应滤波器输出'); grid on;

```

将收敛因子改为 0.3 得到的输出结果如图 5.15 所示。

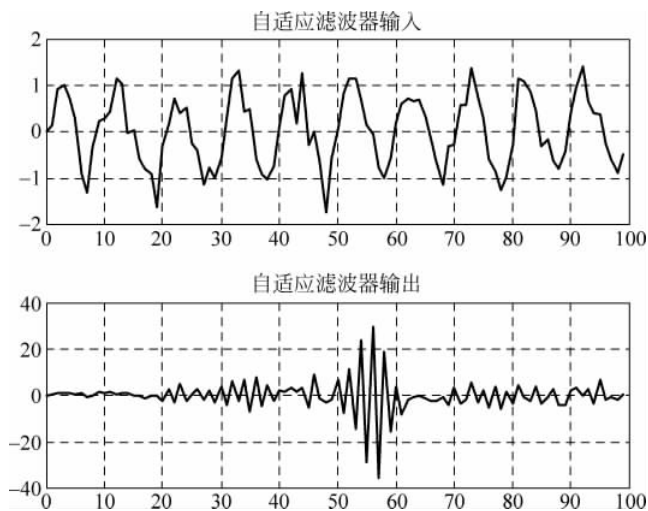


图 5.15 自适应信号分离器输出,收敛因子为 0.3

从上面两种不同的滤波结果可以看出,当收敛因子选取适当时,滤波器输出效果较好;当收敛因子超过一定门限时,滤波器输出结果发散。

**例 5.5** 选取信号为  $s = \sin(\pi t/10)$ , 干扰信号为  $n = 0.5 \cos(\pi t/5 + \varphi)$ , 请用自适应滤波器消除干扰信号,  $\varphi = \pi/3$ 。

**解:** MATLAB 源代码如下(见 chp5sec5\_5.m), 实验结果如图 5.16 所示。

```

% 采用自适应陷波器分离两个单频信号 chp5sec5_5.m
clear;
clc;
N = 200;
t = 0:N-1;
s = sin(pi * t/10); % 原始信号
fai = pi/3;
n = 0.5 * cos(2 * pi * t/10 + fai); % 干扰信号
x = s + n;
subplot(2,2,1); plot(t,s);
title('原始信号'); axis([0,N, -1.5,1.5]); grid on;
subplot(2,2,2); plot(t,x);
axis([0,N, -1.5,1.5]); grid on; title('受干扰后的信号');
x1 = cos(2 * pi * t/10);
x2 = sin(2 * pi * t/10);
w1 = 0.1;

```

```

w2 = 0.2;
e = zeros(1,N);
y = 0;
u = 0.05;           % 收敛因子
for i = 1:N;
    y = w1 * x1(i) + w2 * x2(i);
    e(i) = x(i) - y;
    w1 = w1 + u * e(i) * x1(i);
    w2 = w2 + u * e(i) * x2(i);
end
subplot(2,2,3);plot(t,e);
axis([0,N, -1.5,1.5]); grid on; title('滤波输出结果');
subplot(2,2,4);plot(t,s - e);
axis([0,N, -1.5,1.5]); grid on; title('滤波误差');

```

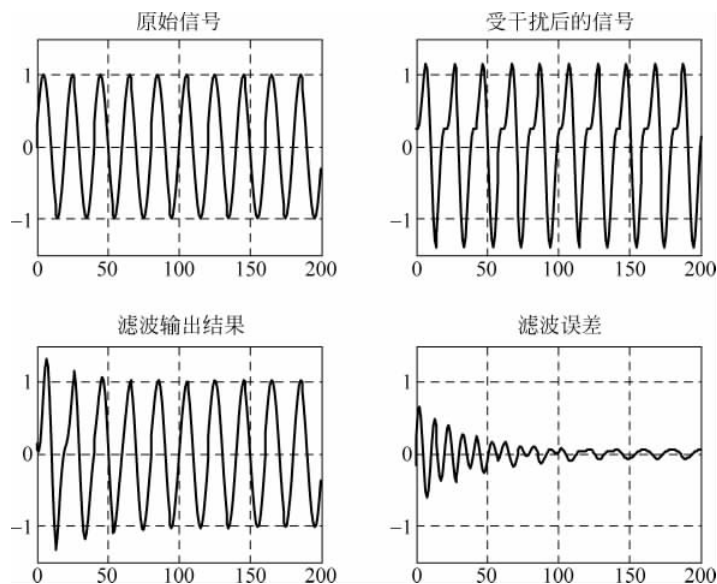


图 5.16 自适应滤波器分离两个单频信号(自适应陷波器)

例 5.5 给出的噪声是单色干扰,此时的自适应信号分类器也称为自适应陷波器。理想自适应陷波器的缺口肩部任意窄,可马上进入平的区域,具有能够自适应地准确跟踪干扰频率、带宽容易控制等优点。

#### 5.4.4 自适应图像去噪

函数 `wiener2` 能够自适应地对带噪声图像进行滤波处理,当噪声方差较大时该函数取较大的平滑窗,当噪声方差较小时取较小的平滑窗。函数 `wiener2` 的滤波效果比传统的线性滤波效果较好,但是以提高运算量为代价的。当噪声功率谱平坦的时候(如高斯白噪声),函数 `wiener2` 的滤波效果最好。

**例 5.6** 采用维纳滤波器来自适应的滤除图像文件 `saturn.png` 中的噪声(注:该例题来自于 MATLAB 的帮助文档)。

**解:** MATLAB 源代码如下(见 `chp5sec5_6.m`),实验结果如图 5.17 所示。

```

% 自适应滤波器(wiener2)滤除带噪图像 chp5sec5_6.m
clear;
clc;
RGB = imread('saturn.png');
I = rgb2gray(RGB);
subplot(1,3,1);imshow(I);title('原始图像');
J = imnoise(I,'gaussian',0,0.025);
subplot(1,3,2);imshow(J);title('未处理');
K = wiener2(J,[5 5]);
subplot(1,3,3);imshow(K);title('滤波结果');

```

% 读取图像数据  
 % RGB 真彩色转灰度图  
 % 在图像中加入高斯白噪声  
 % wiener2 型滤波器滤除噪声

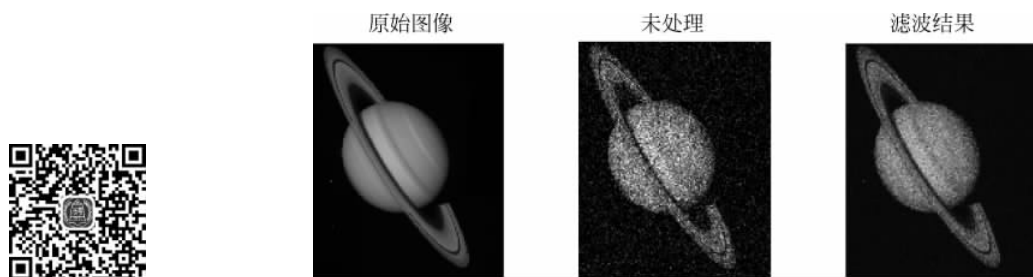


图 5.17 维纳滤波处理后的图像

### 5.4.5 自适应信道均衡

在实际通信环境中,由于信道带宽有限会导致前后脉冲信号相互干扰(码间干扰现象, ISI),信道均衡的目的就是用来抑制或降低码间干扰。从滤波器的观点来看,信道均衡相当于设计一个可以对信道频率响应进行白化处理的滤波器,力图将不平整的信道频率响应尽量拉直。

**例 5.7** 采用自适应均衡器对传输信道进行均衡处理,请分别给出 BPSK 和 QPSK 调制的情况(注:该例题来自于 MATLAB 的帮助文档)。

**解:** MATLAB 源代码如下(见 chp5sec5\_7.m),实验结果如图 5.18 所示。

```

% 自适应信道均衡 chp5sec5_7.m
clear;
clc;
% 设置各种参数
M = 4;
msg = randint(1500,1,M);
modmsg = pskmod(msg,M);
trainlen = 500;
chan = [.986; .845; .237; .123 + .311i];
filtmsg = filter(chan,1,modmsg);

% 调制字母表 alphabet 大小
% 随机信息序列
% M-PSK 调制
% 训练序列长度
% 信道模型系数
% 模拟信道传输失真,ISI 现象

% 对接收信号进行均衡
eq1 = lineareq(8, lms(0.01));
eq1.SigConst = pskmod([0:M-1],M);
[symblest,yd] = equalize(eq1,filtmsg,modmsg(1:trainlen));

```

% 信号星座图  
 % 信道均衡

```

h = scatterplot(filtmsg,1,trainlen,'bx'); hold on;
scatterplot(symboltest,1,trainlen,'g.',h);
scatterplot(eq1.SigConst,1,0,'r*',h);
legend('接收到的信号','均衡后的信号','理想信号星座图');
hold off;

```

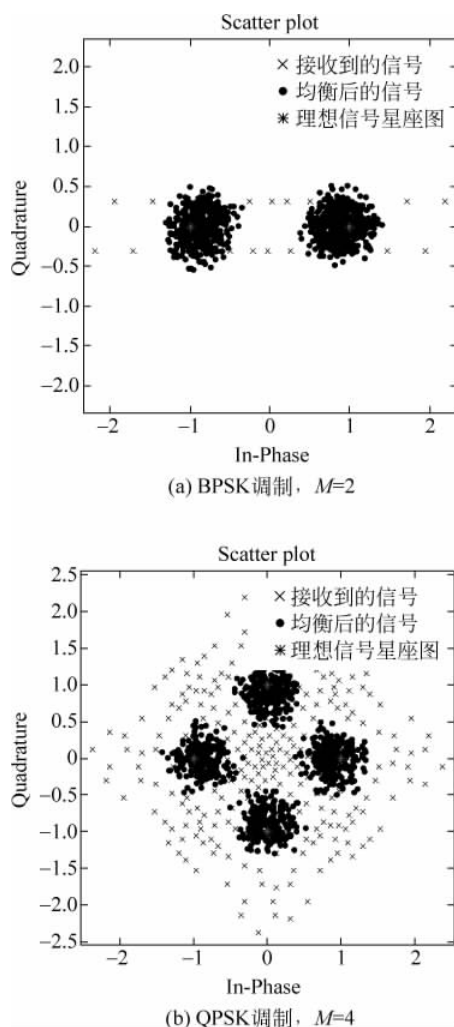


图 5.18 自适应信道均衡

## 5.5 参考文献

- [1] 皇甫堪,陈建文,楼生强.现代数字信号处理[M].北京:电子工业出版社,2003.
- [2] 万建伟,王玲.信号处理仿真技术[M].长沙:国防科技大学出版社,2008.
- [3] Manolakis D G, Ingle V K, Kogon S M. 统计与自适应信号处理(中译版)[M].北京:电子工业出版社,2003.

本章用到的 MATLAB 函数总结

| 函数名称及调用格式                                                 | 函数用途                    | 对应章节数              |
|-----------------------------------------------------------|-------------------------|--------------------|
| ha = adaptfilt, lms(l,step,leakage,coeffs,states)         | LMS 自适应滤波器              | 5.4 节(例 5.1)       |
| sound(s,Fs)                                               | 按照速率 $F_s$ 来播放信号 $s$    | 5.4 节(例 5.1,例 5.2) |
| ha = adaptfilt, rls(l,lambda,invcov,coeffs,states)        | RLS 自适应滤波器              | 5.4 节(例 5.2)       |
| X = eye(m,n)                                              | 产生 $n \times n$ 维单位矩阵   | 5.4 节(例 5.2)       |
| ha = adaptfilt, nlms(l,step,leakage,offset,coeffs,states) | NLMS 自适应滤波器             | 5.4 节(例 5.3)       |
| X = ones(m,n)                                             | 产生 $m \times n$ 维全 1 数组 | 5.4 节(例 5.4)       |
| A = imread(filename, fmt)                                 | 读取图像文件                  | 5.4 节(例 5.6)       |
| J = imnoise(I,type)                                       | 在图像中加入噪声                | 5.4 节(例 5.6)       |
| imshow(I,[low high])                                      | 显示图像                    | 5.4 节(例 5.6)       |
| J = wiener2(I, [m n], noise)                              | 二维自适应滤除噪声               | 5.4 节(例 5.6)       |
| I = rgb2gray( RGB)                                        | 把 RGB 真彩色转换为灰度图         | 5.4 节(例 5.6)       |
| out = randint(m,n,rg)                                     | 产生均匀分布的随机整数             | 5.4 节(例 5.7)       |
| y = pskmod(x,M)                                           | M-PSK 调制                | 5.4 节(例 5.7)       |
| y = equalize(eqobj,x)                                     | 信道均衡                    | 5.4 节(例 5.7)       |
| scatterplot(x,n)                                          | 绘制散点图                   | 5.4 节(例 5.7)       |

注：randint 为 MATLAB 2008 版本函数名，在 MATLAB 2015 版本中该函数已更名为 randi。