第5章 小波用于信号处理

同传统的处理方法相比，小波变换在信号处理方面具有更大的优势。例如，小波分析可以用于电力负载信号的分析与处理，小波变换可以用于语音信号的分析、变换和综合，还可以检测噪声中的未知瞬态信号等。

5.1 小波用于信号滤波

传统的基于傅里叶变换的信号滤波方法要求信号和噪声的频带重叠部分尽可能小，这样在频域就可以通过时不变滤波方法将信号和噪声区分开。但是，当它们的频谱重叠时，这种方法就无能为力了。

基于信号和噪声的小波系数在尺度上的不同性质，采用相应的规则，对含噪信号的小波系数进行取舍、抽取或切削等非线性处理，达到滤波的目的。事实证明，这种去噪处理能取得较好的效果。

5.1.1 小波滤波的原理

小波变换具有时频局部化特性、多分辨特性、去相关特性和选基灵活性。前两个性质决定了小波滤波方法与传统方法相比具有独特的优势，即能够在去除噪声的同时，很好地保留信号的突变部分或图像的边缘。

小波滤波的特点如下：

(1) 非平滑性。平滑是去除高频信息而保留低频信息，而小波滤波是试图去除所有噪声，保留所有信号，并不考虑它们的频率范围。

(2) 在小波变换域对小波系数进行非线性处理。

5.1.2 小波滤波的基本方法

小波滤波研究主要集中在3个方向：基于信号奇异性的模极大值的重构滤波、基于信号尺度间相关性的空域相关滤波和基于小波变换解相关特性的小波阈值滤波。

1. 方法一

第一种方法是 MATLAB 提出的基于小波变换模极大值原理的去噪方法，根据信号和噪声在小波变换各尺度上的不同传播特性，剔除由噪声产生的模极大点，保留信号对应的模极大值点，然后利用剩余模极大值点重构小波系数，进而恢复信号。

小波变换在信号和噪声中有着不同的传播特性，即随着尺度的增大，信号和噪声所对应的模极大值分别增大和减小，因此，连续进行若干次小波变换后，由噪声对应的模极大值已基本去除或幅值很小，而所余极值点主要由信号控制。

换句话说，小尺度上的模极大值点主要由噪声控制，而大尺度上的点主要由信号控制，所以按照尺度从大到小（从下到上）进行去噪，去噪算法的步骤如下。

（1）对加噪信号进行二维小波变换，并求出每一尺度上小波变换系数的模极大值。

（2）从最大尺度（设为 4）开始，选一阈值 A，如果极大值点对应幅值的绝对值小于 A，则去掉该极值点；否则予以保留。这样就得到最大尺度上新的模极大值点。

（3）在尺度为 $k-1(k=4,3)$ 上寻找尺度为 k 上小波变换模极大值点的传播点，即保留由信号产生的极值点，去除由噪声引起的极值点，具体方法见步骤（4）。

（4）在尺度为 k 上的极大值点位置构造一个邻域 $o(n_{k_1},\varepsilon_k)$。其中，n_{k_1} 为尺度 k 上的第 i 个极值点，ε_k 为仅与尺度 k 有关的常数。在尺度为 $k-1$ 上的极大值点保留落在每一邻域 $o(n_{k_1},\varepsilon_k)$ 上的极大值点，而去除邻域外面的极值点，从而得到 $k-1$ 尺度上新的极值点。然后令 $k=k-1$，重复步骤（4），直到 $k=2$ 为止。

（5）把 $k=1$ 时的极值点都置为 0，然后直接把 $k=2$ 时的极值点复制上去。

（6）将每一尺度上保留下来的极值点利用适当的重构方法对信号进行恢复。

2. 方法二

第二种方法是 Xu 等人于 1994 年提出的基于空域相关性的噪声去除方法，根据信号与噪声的小波变换系数在相邻尺度之间的相关性进行滤波。该方法虽不够精确，但很直接，易于实现。空域相关滤波法的思想是，信号的突变点在不同尺度的同一位置有较大的峰值出现，噪声能量随着尺度的增大而减小。

因此，可以取相邻尺度的小波系数直接相乘进行相关计算。这样将在锐化信号边缘域其他重要特征的同时抑制噪声，而且能够提高信号主要边缘的定位精度，更好地刻画真实信号。设

$$\mathrm{CW}_{j,k}=w_{j,k}\cdot w_{j+1,k}$$

称 $\mathrm{CW}_{j,k}$ 为尺度 j 上 k 点处的相关系数。

由于信号在各尺度之间具有一定的相关性，而噪声无此特性，因此尺度空间上的相关运算削弱了噪声，增强了信号的边缘。

为了使得相关系数与小波系数具有可比性，定义归一化相关系数。设

$$\overline{W}_{j,k}=\mathrm{CW}_{j,k}\sqrt{\frac{\mathrm{PW}_j}{\mathrm{PCW}_j}}$$

称 $\overline{W}_{j,k}$ 为归一化相关系数，式中，

$$\mathrm{PW}=\sum_k w_{j,k}^2,\quad \mathrm{PCW}_j=\sum_k \mathrm{CW}_{j,k}^2$$

归一化相关系数$\bar{w}_{j,k}$与小波系数$w_{j,k}$具有相同的能量。

相关去噪的核心环节是，通过比较$\bar{w}_{j,k}$与$w_{j,k}$的绝对值的大小来抽取信号的边缘信息。而且，$\bar{w}_{j,k}$与$w_{j,k}$做比较是合理的。剩下的是噪声对应的小波系数，经过若干次迭代后，所剩下的小波系数的能量会低于某一门限，则认为信号已经被完全提取出来了。

3. 方法三

第三种方法是 Donoho 和 Johnstone 等人于 1994 年提出的小波阈值滤波法。信号经过小波变换后，认为由信号产生的小波系数包含信号的重要信息，其幅值比较大，但数目较少，而噪声对应的小波系数幅值小。

基于这一思想，Donoho 等人提出软阈值和硬阈值去噪方法。通过在不同尺度上选取合适的阈值，并将小于该阈值的小波系数置零，而保留大于阈值的小波系数，从而使信号中的噪声得到有效的抑制。最后进行小波逆变换，得到滤波后的重构信号。这就是小波阈值滤波。

上述介绍的 3 种方法的定性比较如表 5-1 所示。

表 5-1　3 种滤波方法的定性比较

滤波方法	计　算　量	稳　定　性	滤波效果	适用范围
模极大值法	大	稳定	较好	低信噪比信号
空域相关法	较大	较稳定	好	高信噪比信号
小波阈值法	小	依赖于信噪比	好	低信噪比信号

比较发现，小波阈值滤波法实现具有简单、计算量小、滤波效果好、适用于低信噪比信号的处理等特点。因此，考虑到滤波效果和硬件实现的可行性，用小波阈值滤波法较好。

5.1.3　滤波器

数字信号处理的目的之一是设计某种设备或建立某种算法用以处理序列，使序列具有某种确定的性质，这种设备或算法结构就称为数字滤波器。

1. 陷波滤波器

数字陷波滤波器是抑制或滤除数字信号中的窄带干扰的有效途径。在通信技术、生物工程、雷达声呐、测量仪器等领域的应用非常广泛。在一维信号中，基于有限冲激响应(FIR)和无限冲激响应(IIR)的数字陷波滤波器的设计和性能分析被很多学者研究。

二维信号中，Pei 等人通过设计几个简单的一维 IIR 滤波器来实现二维 IIR 陷波滤波器。通常，在陷波带宽很小的情况下，IIR 陷波滤波器更优，因为它的计算量比 FIR 陷波滤波器小很多。

过去 20 年，IIR 陷波滤波器通过代数的方法设计而没有最优过程。现有的方法大致

可以分为两大类：

第一类是零点 IIR 陷波滤波器——它的零点在单位圆上，角度等于陷波频率，极点在零点的半径线上。

第二类是通过设计全通滤波器 $F(z)$来合成陷波滤波器 $H(z)$

$$H(z)=\frac{1}{2(1+F(z))}$$

这种类型的滤波器，零点仍然在单位圆上，角度等于陷波频率，但是极点只是靠近零点而和零点具有不同的角频率。这种方法的优点是，分子和分母多项式的镜像关系可以在灵敏度很低的情况下获得高计算效率的桥型网络滤波器。

理想的陷波滤波器的频率响应为

$$D(w)=\begin{cases}0, & w=\pm w_{N_k}, k=1,2,\cdots,M\\ 1, & \text{其他}\end{cases}$$

式中，w_{N_k}为陷波频率，不失一般性，假设 $w_{N_1}<w_{N_2}<\cdots<w_{N_M}$。有两种方法可实现这样的陷波滤波器。

1）方法一

数字陷波滤波器的传递函数

$$H_1(z)=\frac{\prod_{k=1}^{M}[1-2\cos(w_{N_M})z^{-1}+z^{-2}]}{\prod_{k=1}^{M}[1-2r\cos(w_{N_M})z^{-1}+r^2z^{-2}]}=\frac{B(z)}{B(r^{-1}z)} \tag{5-1}$$

式中，$B(z)=\sum_{k=0}^{2M}b_kz^{-k}$ 是一个对称多项式，有

$$b_0=b_{2M}=1,\quad b_{2M-k}=b_k,\quad k=1,2,\cdots,M-1$$

给定陷波频率 w_{N_k}，就可计算获得系数 b_k。此外，滤波器的 $2M$ 个零点和极点如下：

- 零点为 $e^{\pm jw_{N_k}}$；
- 极点为 $re^{\pm jw_{N_k}}$，$k=1,2,\cdots,M$。

零点被放置在陷波频率为 w_{N_k}的单位圆上，极点放置在与零点相同的半径线上。

2）方法二

这种类型的 IIR 数字陷波滤波器的传递函数

$$H_2(z)=\frac{1+F(z)}{2}$$

式中，$F(z)$是一个 $2M$ 阶的全通滤波器：

$$F(z)=\frac{f_2M+\cdots+f_1z^{-2M+1}+z^{-M}}{1+f_1z^{-1}+\cdots+f_{2M}z^{-2M}}$$

当指定了陷波频率 w_{N_k}，只需要计算系数 $f_k(k=1,2,\cdots,2M)$。$H_2(z)$的 $2M$ 个零点也是 $e^{\pm jw_{N_k}}$，但是 $2M$ 个极点都紧邻零点，用来补偿通带单位增益。

下面给出设计陷波滤波的步骤：

(1) 将 $2M$ 个零点置于单位圆上，幅角为$\pm w_{N_k}(k=1,2,\cdots,M)$，因此，传递函数的分子可写为：$\prod_{k=1}^{M}(1-2\cos(w_{N_M})z^{-1}+z^{-2})$。

(2) 在单位圆内紧邻零点的地方放置 $2M$ 个极点，使得其在非陷波处的频率相位为单位增益。当 $2M$ 个极点非常靠近 $2M$ 个零点时，陷波滤波器就变成了理想滤波器。

2. 单陷波滤波器

单陷波滤波器传递函数

$$H(z)=\frac{B(z)}{A(z)}=\frac{1-2\cos(w_{N_1})z^{-1}+z^{-2}}{1-2raz^{-1}+r^2z^{-2}}$$

式中，$a=\cos(w_P)$。这个滤波器的极点为 $re^{\pm jw_P}$，且参数 a 必须满足 $-1\leqslant a\leqslant 1$。

设计中，极点半径 r 是预先指定的。因此，这个问题就转化为怎样寻找相角 $w_P=\arccos(a)$，使得

$$J(a)=\int_R W(w)\mid 1-H(e^{jw})\mid dw$$

所示的代价函数最小。其中，积分区间 $R=[0,w_{N_1}-\varepsilon]\cup[w_{N_1}+\varepsilon,\pi]$，$W(w)$ 为加权函数，ε 为一个小的正数。由上式可得

$$\begin{aligned}J(a)&=\int_R W(w)\left|1-\frac{B(e^{jw})}{A(e^{jw})}\right|^2 dw\\&=\int_R\frac{W(w)}{\mid A(e^{jw})\mid^2}\mid A(e^{jw})-B(e^{jw})\mid^2 dw\\&=\int_R\frac{W(w)}{\mid A(e^{jw})\mid^2}\mid p(w)-q(w)a\mid^2 dw\end{aligned}\tag{5-2}$$

式中，$p(w)$ 和 $q(w)$ 分别为

$$p(w)=(r^2-1)e^{-2jw}+2\cos(w_{N_1})e^{-jw}$$

$$q(w)=-2re^{-jw}$$

式(5-2)最优化问题可通过以下迭代方法解决：

$$\begin{aligned}J_k(a_k)&=\int_R\frac{W(w)}{\mid A_{k-1}(e^{jw})\mid^2}\mid p(w)-q(w)a_k\mid^2 dw\\&=\rho_{k-1}a_k^2+2\beta_{k-1}a_k+\alpha_{k-1}\end{aligned}$$

式中，ρ_{k-1}、β_{k-1} 和 α_{k-1} 分别为

$$\rho_{k-1}=\int_R\frac{W(w)}{\mid A_{k-1}(e^{jw})\mid^2}\mid q(w)\mid^2 dw$$

$$\beta_{k-1}=\int_R\frac{W(w)}{\mid A_{k-1}(e^{jw})\mid^2}\mathrm{Real}(p(w)q^*(w))dw$$

$$\alpha_{k-1}=\int_R\frac{W(w)}{\mid A_{k-1}(e^{jw})\mid^2}\mid p(w)\mid^2 dw$$

需要注意的是，分母多项式是作为前面加权函数的一部分进行如上所示的迭代得到的。此外，在第 k 次迭代时，$A_{k-1}(e^{jw})$ 是已知的，因此 a_k 可通过解以下的标准二次规划问题而得到：

$$\begin{aligned}&\min\ \rho_{k-1}a_k^2+2\beta_{k-1}a_k+\alpha_{k-1}\\&\text{s.t.}\quad -1\leqslant a_k\leqslant 1\end{aligned}$$

在该优化过程中只需考虑单变量问题，且代价函数是一条向上开口的抛物线，即最优解为

$$a_k=\begin{cases}1, & \dfrac{-\beta_{k-1}}{\rho_{k-1}}\geqslant 1\\ -1, & \dfrac{-\beta_{k-1}}{\rho_{k-1}}\leqslant -1\\ \dfrac{-\beta_{k-1}}{\rho_{k-1}}, & -1<\dfrac{-\beta_{k-1}}{\rho_{k-1}}<1\end{cases}$$

3. 多频率陷波滤波器

IIR 多陷波滤波器的传递函数

$$H(z)=\frac{B(z)}{A(z)}$$

分母多项式定义为

$$A(z)=\prod_{k=1}^{M}\left[1-2r\cos(w_{p_k})z^{-1}+r^2z^{-2}\right]$$

式中，w_{p_k} 为极点相角。传递函数 $H(z)$为传统的第一种类型的陷波滤波器。

当 w_{p_k} 等于陷波频率 w_{N_k} 时，$H(z)$就是式(5-1)描述的第一类陷波滤波器 $H_1(z)$。但是，这种选择不是最佳的，最好的选择是使得函数

$$J(w_{p_1},\cdots,w_{p_M})=\int_R W(w)\mid 1-H(\mathrm{e}^{\mathrm{j}w})\mid^2\mathrm{d}w \tag{5-3}$$

的极点相角$\{w_{p_1},w_{p_2},\cdots,w_{p_M}\}$最小。

因为 J 不是极点相角 w_{p_k} 的二次方，所以不能用二次规划算法来解这类非线性优化问题。为了得到一个较好的结果，将 $A(z)$重写为

$$A(z)=\sum_{k=0}^{2M}a_kr^kz^{-k}$$

$A(z)$的系数具有对称特性：$a_0=a_{2M}=1$，$a_{2M-k}=a_k(k=1,2,\cdots,M-1)$。

定义两个向量 $\boldsymbol{a}=[a_1,\cdots,a_M]^{\mathrm{T}}$ 和

$$\boldsymbol{e}(z)=\begin{bmatrix}rz^{-1}+r^{2M-1}z^{-(2M-1)}\\ r^2z^{-2}+r^{2M-2}z^{-(2M-2)}\\ \vdots\\ r^{M-1}z^{-(M-1)}+r^{M+1}z^{-(M+1)}\\ r^Mz^{-M}\end{bmatrix} \tag{5-4}$$

那么，多项式 $A(z)$可写为

$$A(z)=1+r^{2M}z^{-2M}+\boldsymbol{a}^{\mathrm{T}}\boldsymbol{e}(z) \tag{5-5}$$

由式(5-5)可知，寻找使代价函数 J 最小的极点相角 w_{p_k}，转化为找到一个使 J 最小的系数向量 $\boldsymbol{a}$。因此式(5-3)可改写为

$$\begin{aligned}J(\boldsymbol{a})&=\int_R W(w)\mid 1-H(\mathrm{e}^{\mathrm{j}w})\mid^2\mathrm{d}w=\int_R W(w)\left|1-\frac{B(\mathrm{e}^{\mathrm{j}w})}{A(\mathrm{e}^{\mathrm{j}w})}\right|^2\mathrm{d}w\\ &=\int_R\frac{W(w)}{\mid A(\mathrm{e}^{\mathrm{j}w})\mid}\mid u(w)+\boldsymbol{v}(w)^{\mathrm{T}}\boldsymbol{a}\mid^2\mathrm{d}w\end{aligned} \tag{5-6}$$

式中，

$$u(w)=1+r^{2M}\mathrm{e}^{-\mathrm{j}2Mw}-\sum_{k=0}^{2M}b_k\mathrm{e}^{-\mathrm{j}kw}$$

$$\boldsymbol{v}(w)=\boldsymbol{e}(\mathrm{e}^{\mathrm{j}w})$$

积分区间 $R=[0,w_{N_1}-\varepsilon]\cup[w_{N_1}+\varepsilon,w_{N_2}-\varepsilon]\cup\cdots\cup[w_{N_M}+\varepsilon,\pi]$。

利用式(5-4),那么式(5-6)的优化问题就可以通过以下的迭代方法来解:

$$J_k(\boldsymbol{a}_k)=\int_R\frac{W(w)}{|A_{k-1}(\mathrm{e}^{\mathrm{j}w})|^2}|u(w)+\boldsymbol{v}(w)^{\mathrm{T}}\boldsymbol{a}|^2\mathrm{d}w=\boldsymbol{a}_k^{\mathrm{T}}\boldsymbol{Q}_{k-1}\boldsymbol{a}_k+2\boldsymbol{p}_{k-1}^{\mathrm{T}}\boldsymbol{a}_k+c_{k-1}$$

式中,矩阵 $\boldsymbol{Q}_{k-1}$、矢量 $\boldsymbol{p}_{k-1}$和标量 c_{k-1}分别为

$$\boldsymbol{Q}_{k-1}=\int_R\frac{W(w)}{|A_{k-1}(\mathrm{e}^{\mathrm{j}w})|^2}\mathrm{Real}(\boldsymbol{v}(w)u^*(w))\mathrm{d}w$$

$$\boldsymbol{p}_{k-1}=\int_R\frac{W(w)}{|A_{k-1}(\mathrm{e}^{\mathrm{j}w})|^2}\mathrm{Real}(\boldsymbol{v}(w)v^{\mathrm{H}}(w))\mathrm{d}w$$

$$c_{k-1}=\int_R\frac{W(w)}{|A_{k-1}(\mathrm{e}^{\mathrm{j}w})|^2}|u(w)|^2\mathrm{d}w$$

因为第 k 次迭代时,$A_{k-1}(w)$已知,所以 $\boldsymbol{a}_k$ 可通过以下优化问题得到:

$$\min\boldsymbol{a}_k^t\boldsymbol{Q}_{k-1}\boldsymbol{a}_k+2\boldsymbol{p}_{k-1}^t\boldsymbol{a}_k+c_{k-1}$$

$$\text{s.t.}\quad A_k(z)\text{ 所有的零点都在单位圆上}$$

因为系数 $\boldsymbol{a}_k$ 对称是所有极点都在半径为 r 的圆上的必要不充分条件,$0\leqslant r<1$ 的条件不能够保证在单陷波频率的情况下滤波器稳定。

5.1.4 小波滤波器函数的实现

MATLAB 中,提供了相关函数用于实现相关小波滤波器的设计。这些函数在前面的章节已作介绍,在此不再展开介绍。下面通过几个实例来演示小波滤波器的实现。

【例 5-1】 设置一个 db4 小波滤波器。

```
>> clear all;
load db4;                          % 载入 db4 小波尺度
w = db4;
subplot(411);stem(w);
title('初始尺度滤波器');
% 计算 4 个滤波器
[Lo_D,Hi_D,Lo_R,Hi_R] = orthfilt(w);
subplot(423);stem(Lo_D);
title('分解后的低通滤波器');
subplot(424);stem(Hi_D);
title('分解后的高通滤波器');
subplot(425);stem(Lo_R);
title('重构后的低通滤波器');
subplot(426);stem(Hi_R);
title('重构后的高通滤波器');
df = [Lo_D;Hi_D];
rf = [Lo_R;Hi_R];
id = df * df'
df = [Lo_D 0 0;Hi_D 0 0];
dft = [0 0 Lo_D;0 0 Hi_D];
zer = df * dft'
% 高低频说明
fftld = fft(Lo_D);
ffthd = fft(Hi_D);
```

```
freq = [1:length(Lo_D)]/length(Lo_D);
subplot(427);plot(freq,abs(fftld));
title('变换: 低通');
subplot(428);plot(freq,abs(ffthd));
title('变换: 高通');
```

运行程序,输出如下,效果如图 5-1 所示。

```
id =
    1.0000         0
         0    1.0000
zer =
   1.0e-14 *
   -0.9992   -0.0008
         0   -0.9997
```

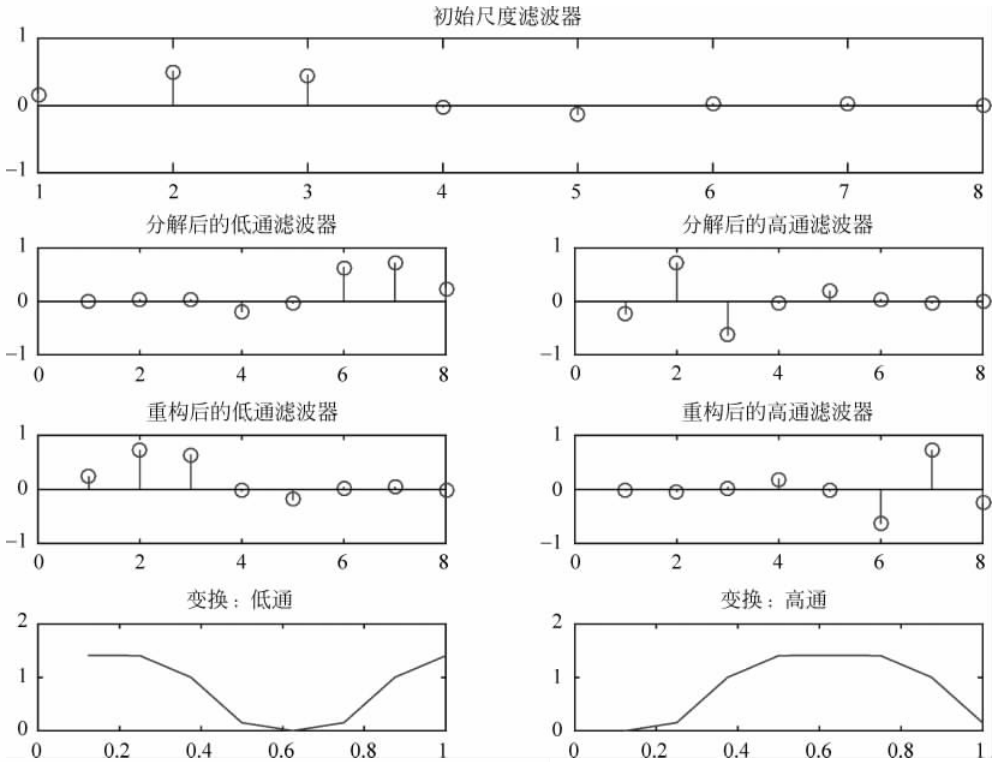

图 5-1　正交小波滤波器

下面通过一个例子介绍滤波函数与 dwt 函数的混合使用。

【例 5-2】　利用 db4 小波单层分解带有随机噪声的信号,并且设置其高、低通滤波器,利用滤波器来分解信号,最后重构信号。

```
>> clear all;
randn('seed',531316785);
s = 2 + kron(ones(1,8),[1  -1]) + ((1:16).^2)/32 + 0.2 * randn(1,16);
[cal,cdl] = dwt(s,'db4');
subplot(311);plot(s);
title('原始信号');
```

```
subplot(323);plot(cal);
title('近似系数');
subplot(324);plot(cdl);
title('细节系数');
%计算 db4 小波高、低通滤波器
[Lo_D,Hi_D] = wfilters('db4');
[ca2,cd2] = dwt(s,Lo_D,Hi_D);
subplot(325);plot(ca2);
title('利用高低通滤波器分解出近似系数');
subplot(326);plot(cd2);
title('利用高低通滤波器分解出细节系数');
figure;
%重构信号
ss = idwt(cal,cdl,'db4');
err = norm(s - ss);
subplot(211);plot([s;ss]');
title('初始与重构信号');
xlabel(['误差 = ',num2str(err)]);
[Lo_R,Hi_R] = wfilters('db4','r');
ss1 = idwt(cal,cdl,Lo_R,Hi_R);
subplot(212);plot(ss1,'r');
title('初始与利用滤波器重构信号');
```

滤波器设置与利用滤波器重构信号如图 5-2 及图 5-3 所示。可见，小波重构的过程是简单地类似 FIR 滤波运算的过程。

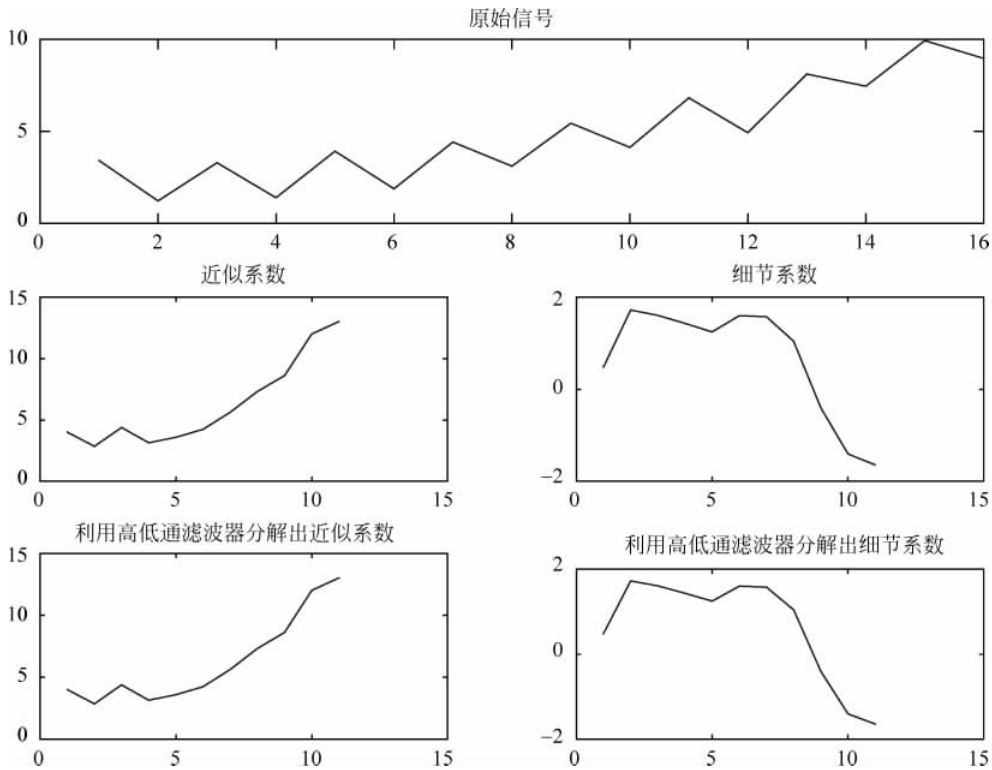

图 5-2　利用滤波器分解信号

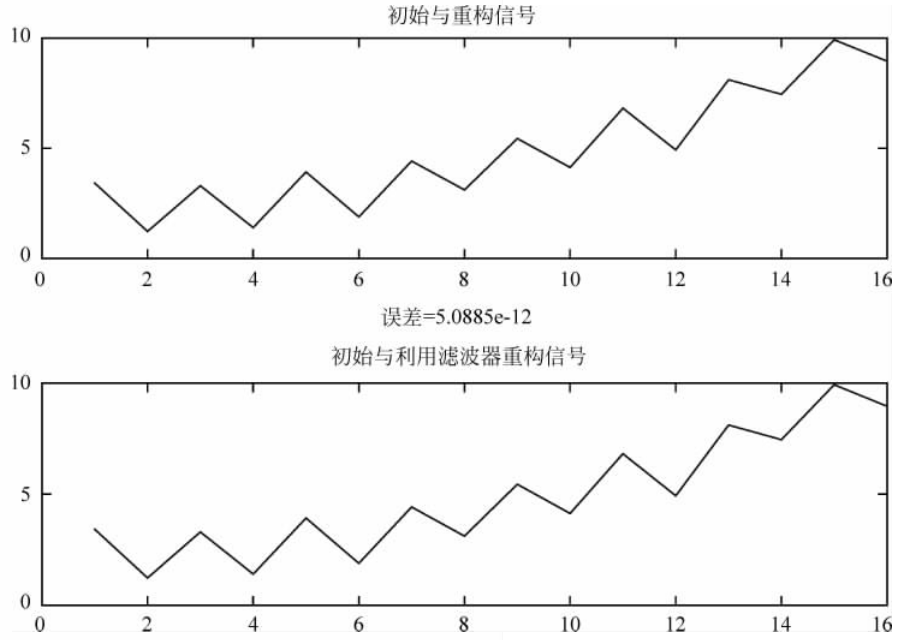

图 5-3　利用滤波器重构信号

5.1.5　重构滤波器组

由前面了解到，小波重构的过程简单地类似 FIR 滤波运算的过程。那么，小波滤波的过程是什么呢？重构后的信号和原信号之间的误差是怎样产生的？

编写小波滤波算法，其过程实际上就是完全重构滤波器进行滤波的过程。一个简单滤波系统如图 5-4 所示，其中 H 是 FIR 滤波器。

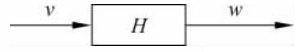

图 5-4　简单滤波系统

设 $v=[v(1)v(2)v(3)v(4)v(5)]$，$h=[a(1)a(2)a(3)a(4)]$，则有

$$V(z) = v(1) + v(2)z^{-1} + v(3)z^{-2} + v(4)z^{-3} + v(5)z^{-4}$$

$$H(z) = a(1) + a(2)z^{-1} + a(3)z^{-2} + a(4)z^{-3}$$

由 $W(z)=H(z)V(z)$，对应的 $W(z)$ 可为一个 w 序列。MATLAB 中，提供的 conv 函数恰好用于计算多个输入的序列。

下面通过一个实例来演示完全重构滤波组。

【例 5-3】　利用 MATLAB 自带信号，选用 db4 小波验证完全重构滤波器。

```
>> clear all;
load leleccum;
N1 = 1001;
v = leleccum(1:N1);
subplot(211);plot(v);
```

```
title('初始信号');
[h0,h1,h2,h3] = wfilters('db4');
w0 = conv(h0,v);
x0 = conv(h1,v);
[ca1,cd1] = dwt(v,'db4');
N = floor(length(x0)/2);
w00 = reshape(w0,2,N);
x00 = reshape(x0,2,N);
w1 = w00(1,:);
x1 = x00(1,:);
w2 = [w1;zeros(1,N)];
w3 = w2(:)';
x2 = [x1;zeros(1,N)];
x3 = x2(:)';
y = conv(h2,w3) + conv(h3,x3);
y1 = y(7:length(y) - 8);
subplot(212);
plot(y1);title('完全重构');
```

运行程序,效果如图 5-5 所示。

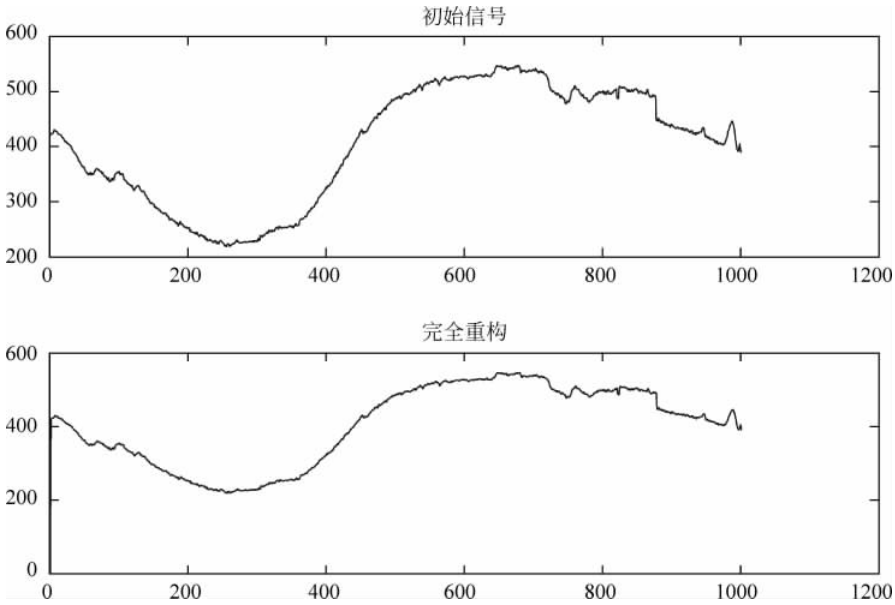

图 5-5 验证完全重构滤波器

5.1.6 小波滤波器构造

前面曾经介绍过 wavedec 和 wrcoef 两个函数,它们是小波分解与重构的主要函数,同时也是构成一维小波滤波程序的主要部分。

下面通过一个实例来了解其滤波过程。

【例 5-4】 对噪声信号 noisbloc 进行滤波处理，利用 db4 小波对其进行 5 层分解。

```
>> clear all;
load noisbloc
x = noisbloc;
% 使用 db4 小波对其进行 6 层分解
[c,l] = wavedec(x,6,'db4');
ca = wrcoef('a',c,l,'db4',6);
% 从 I 中读出所有细节系数所在下标
index = l(1) + l:l(8);
% 得到一个 c 的复本
c1 = c;
% 对细节部分进行一定程度的抑制
cl(index) = c(index)/100;
if abs(c(index))> 0.05                % 设定小波系数域值,令值大于 0.1 时,系数为 0
    c(index) == 0;
end;
% 通过抑制后的系数重建小波
x2 = waverec(c1,l,'db4');
% 求去噪后的 ca 信号在原信号的重构成分
per1 = norm(ca)/norm(x)
% 求去噪后的 x2 信号在原信号的重构成分
per2 = norm(x2)/norm(x)
% 将 ca 作为去噪信号,求其与原信号的标准差
err1 = norm(ca - x)
% 将 x2 作为去噪信号,求其与原信号的标准差
err2 = norm(x2 - x)
subplot(311);plot(x);                % 原始信号
legend('原信号');
subplot(312);plot(ca);               % 这里 ca 为第 5 层近似信号
legend('第 5 层近似信号');
subplot(313);plot(x2);               % 这里 x2 为通过抑制后系数重构的信号
legend('滤波后信号');
```

运行程序，输出如下，效果如图 5-6 所示。

```
per1 =
    0.9030
per2 =
    1.0000
err1 =
  103.8246
err2 =
   4.5964e - 10
```

【例 5-5】 通过 FIR 构造小波滤波器，并且对信号 $s=8\sin(50\pi nt)e^{-10nt}$ 消噪处理。

```
>> clear all;
% 用于滤波器设计和消噪
a = pi/4;                                    % 角度赋初值
b = pi/4;
% 低通重构 FIR 滤波器 h0(n)冲激响应赋值
h0 = cos(a) * cos(b);
h1 = sin(a) * cos(b);
h2 =- sin(a) * sin(b);
```

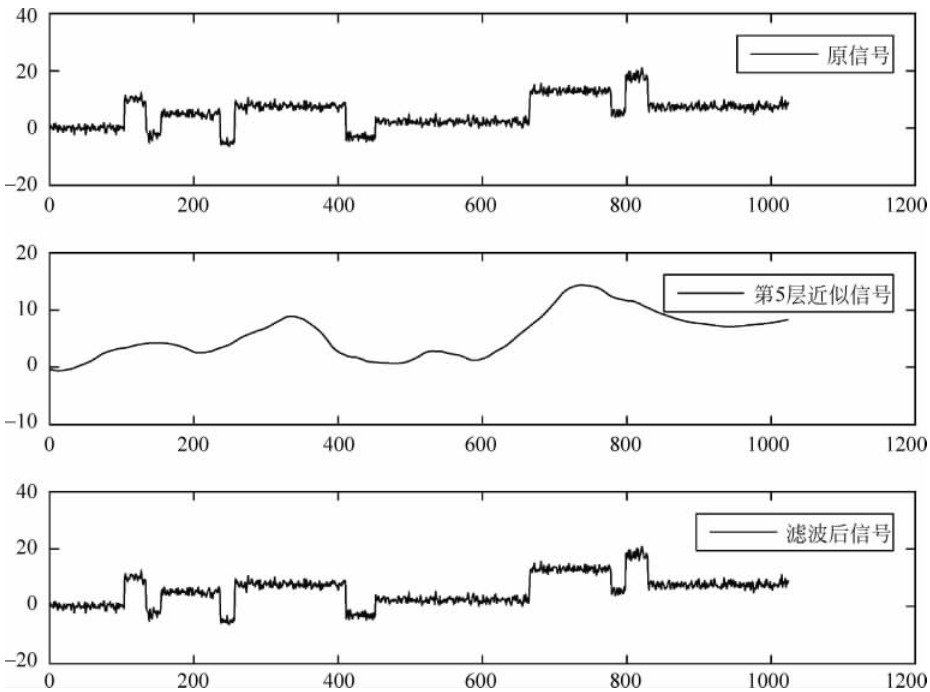

图 5-6　滤波后的信号

```
h3 = cos(a) * sin(b);
low_construct = [h0,h1,h2,h3];
L_fre = 4.5;                                    % 滤波器长度
low_decompose = low_construct(end: - 1:1);      % 确定 h0( - n),低通分解滤波器
for i_h = 1:L_fre;                              % 确定 h1(n) = ( - 1)^n 高通重构滤波器
    if(mod(i_h,2) == 0);
        coefficient =- 1;
    else
        coefficient = 1;
    end
    h_construct(1,i_h) = low_decompose(1,i_h) * coefficient;
end
h_decompose =  h_construct(end: - 1:1);         % 高通分解滤波器
L_signal = 1000;                                % 信号长度
n = 1:L_signal;                                 % 信号赋值
f = 10;
t = 0.001;
y = 8 * sin(pi * 50 * n * t). * exp( - 10 * n * t);
figure;
plot(y);title('原信号');                        % 效果如图 5 - 7 所示
check1 = sum(h_decompose);                      % 对 h0(n)性质检验
check2 = sum(low_decompose);
check3 = norm(h_decompose);
check4 = norm(low_decompose);
l_fre = conv(y,low_decompose);                  % 卷积
l_fre_d = dyaddown(l_fre);                      % 抽取,得到低频细节
h_fre = conv(y,h_decompose);
h_fre_d = dyaddown(h_fre);                      % 信号高频细节
figure;                                         % 效果如图 5 - 8 所示
```

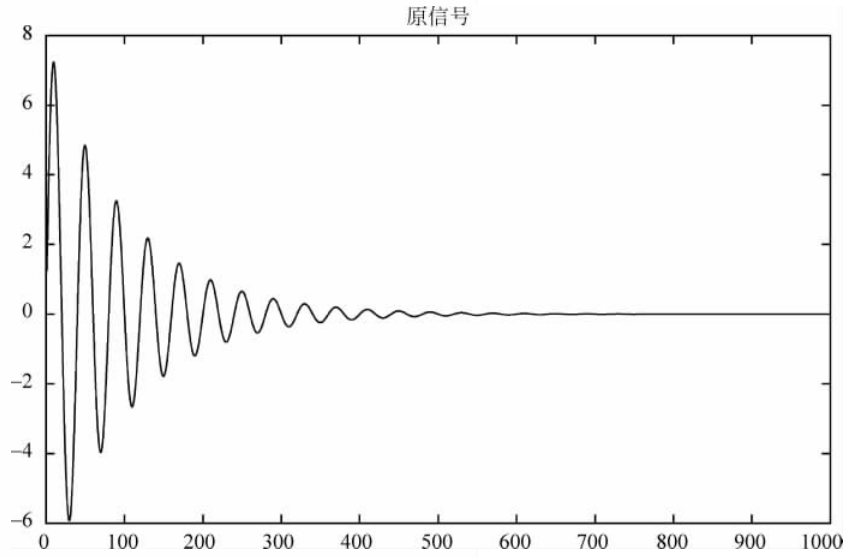

图 5-7　原始信号

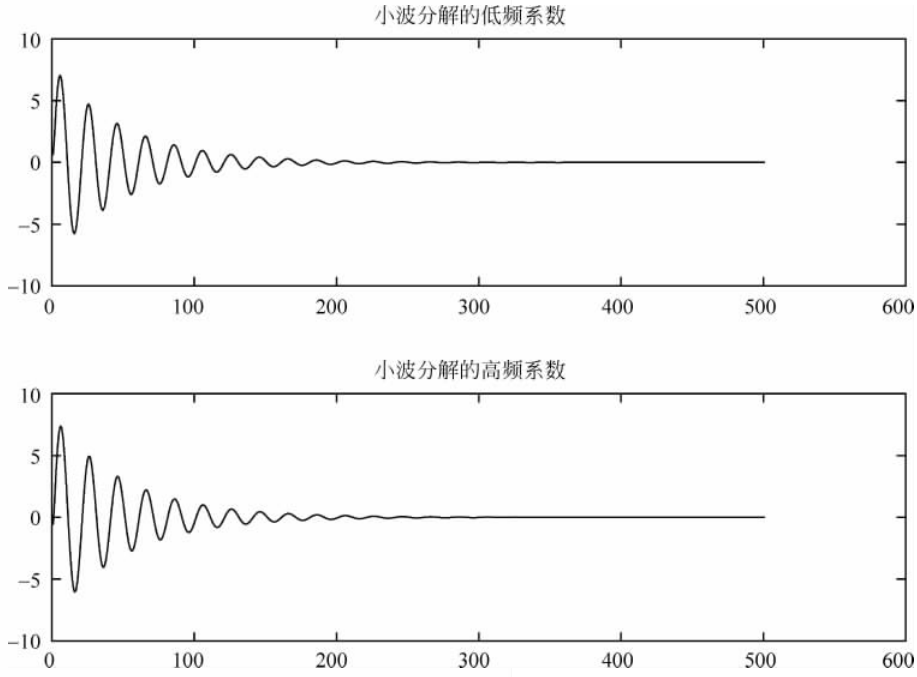

图 5-8　分解系数

```
subplot(211);plot(l_fre_d);
title('小波分解的低频系数');
subplot(212); plot(h_fre_d);
title('小波分解的高频系数')
l_fre_p = dyadup(l_fre_d);                    % 零差值
h_fre_p = dyadup(h_fre_d);
l_fre_de = conv(low_construct,l_fre_p);
h_fre_de = conv(h_construct,l_fre_p);
```

```
l_fre_k = wkeep(l_fre_de,L_signal);          %取结果的中心部分,消除卷积影响
h_fre_k = wkeep(h_fre_de,L_signal);
sig_de = l_fre_k + h_fre_k;                  %信号重构
compare = sig_de - y;                        %与原信号比较
figure;                                      %效果如图 5-9 所示
subplot(311);plot(y);
title('原信号');
subplot(312);plot(sig_de);
title('重构信号');
subplot(313);plot(compare);
title('原信号与消噪后信号的比较');
```

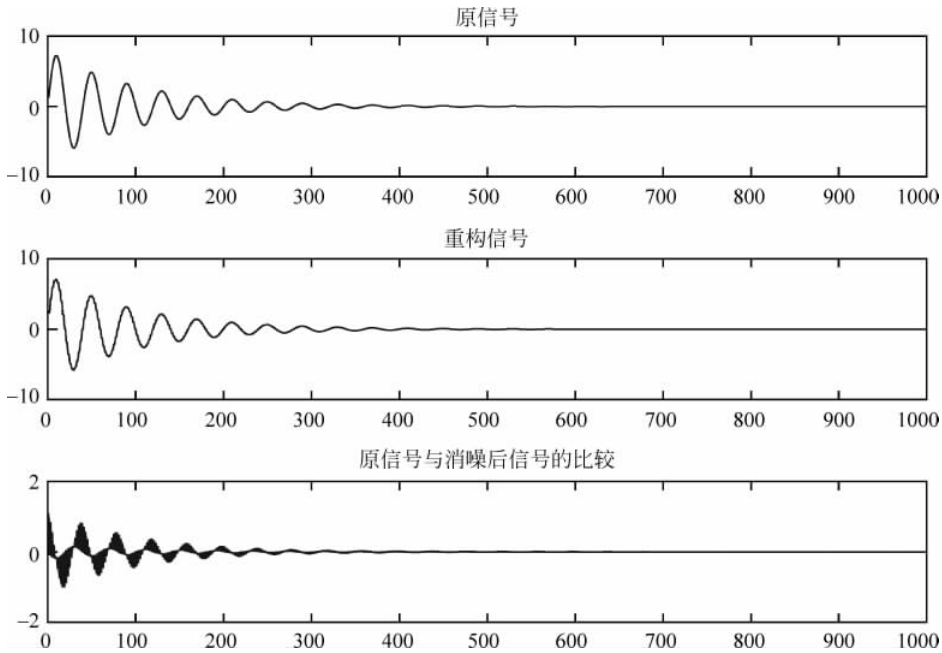

图 5-9　原信号与消噪后信号的比较

5.2　信号去噪

信号去噪是信号处理领域的经典问题之一。传统的去噪方法主要包括线性滤波方法和非线性滤波方法，如中值滤波和 Wiener 滤波等。传统去噪方法的不足在于使信号变换后的熵增高，无法刻画信号的非平稳特性，并且无法得到信号的相关性。为了克服上述缺点，人们开始使用小波变换解决信号去噪问题。

5.2.1　小波变换特性

小波变换具有下列良好特性。

(1) 低熵性：小波系数稀疏分布，使信号变换后的熵较低。

(2) 多分辨率特性：可以非常好地刻画信号的非平稳特性，如边缘、尖峰、断点等。

(3) 去相关性：可去除信号的相关性，且噪声在小波变换后有白化趋势，所以更利于去噪。

(4) 选基灵活性：由于小波变换可以灵活选择基函数，因此可根据信号特点和去噪要求选择适合小波。

5.2.2 信号去噪过程

一般地，信号去噪的过程可分为如下 3 个步骤。

(1) 信号的小波分解。选择一个小波并确定分解的层次，然后进行分解计算。

(2) 小波分解高频系数的阈值量化。对各个分解尺度下的高频系数选择一个阈值进行软阈值量化处理。

(3) 小波重构。根据小波分解的最底层低频系数和各层高频系数进行一维小波重构。

这 3 个步骤中，最关键的是如何选择阈值以及进行阈值量化。在某种程度上，它关系到信号消噪的质量。

总体上，对于一维离散信号来说，其高频部分所影响的是小波分解的第一层细节，其低频部分所影响的是小波分解的最深层和低频层。如果对一个仅由白噪声所组成的信号进行分析，可得出这样的结论：高频系数的幅值随着分解层次的增加而迅速地衰减，且其方差也有同样的变化趋势。

5.2.3 信号去噪方法

小波分析进行消噪处理，一般有下述 3 种方法。

(1) 默认阈值消噪处理。该方法利用函数 ddencmp 生成信号的默认阈值，然后利用函数 wdencmp 进行消噪处理。

(2) 给定阈值消噪处理。在实际的消噪处过程中，阈值往往可通过经验公式获得，且这种阈值比默认阈值的可信度高。在进行阈值量化处理时可利用函数 wthresh。

(3) 强制消噪处理。该方法是将小波分解结构中的高频系数全部置为 0，即滤掉所有高频部分，然后对信号进行小波重构。这种方法比较简单，且消噪后的信号比较平滑，但是容易丢失信号中的有用成分。

小波在信号去噪领域已得到越来越广泛的应用。阈值去噪方法是一种实现简单、效果较好的小波去噪方法。下面从阈值函数和阈值估计两方面对阈值去噪方法进行介绍。

1. 阈值函数

常用的阈值函数主要包括硬阈值和软阈值函数。

(1) 硬阈值函数的表达式为

$$\eta(w) = wI(|w| > T)$$

(2) 软阈值函数的表达式为

$$\eta(w) = [w - \mathrm{sgn}(w)T]I(|w| > T)$$

一般来说，硬阈值方法可以很好地保留信号边缘等局部特征，软阈值处理结果相对更平滑，但会造成边缘模糊等失真现象。为了克服上述缺陷，提出了一种半软阈值函数。它可以兼顾软阈值和硬阈值方法的优点，其表达式为

$$\eta(w)=\operatorname{sgn}(w)\frac{T_2(|w|)-T_1}{T_2-T_1}I(T_1<|w|<T_2)+wI(|w|>T_2)$$

式中，$0<T_1<T_2$。

在软阈值的基础上，可对其改进，使其具有更高阶。表达式为

$$\eta(w)=\begin{cases}w+T-\dfrac{T}{2k+1}, & w<-T\\ \dfrac{1}{(2k+1)T^{2k}}w^{2k+1}, & |w|\leqslant T\\ w-T+\dfrac{T}{2k+1}, & w>T\end{cases}$$

2. 阈值估计

Donoho 在 1994 年提出了 VisuShrink 方法(或称统一阈值去噪方法)。其是针对多维独立正态变量联合分布，在维数趋向无穷时得出的结论，在最小最大估计的限制下得出最优阈值。阈值的选择满足

$$T=\sigma_n\sqrt{2\ln N}$$

式中，σ_n 为噪声标准方差，N 为信号的长度。Donoho 给出了证明这种估计在信号属于 Besov 集时，在大量风险函数下获得近似理想的去噪风险。Donoho 的统一阈值方法在实际应用中效果欠理想，产生过扼杀现象，1997 年 Janse 提出了基于无偏估计的阈值计算法。

风险函数定义为

$$R(t)=\frac{1}{N}\|\hat{f}-f\|^2$$

由于小波变换的正交性，风险函数可同样在小波域中写成

$$R(t)=\frac{1}{N}\|\eta_t(Y)-X\|^2$$

设 $T(t)=\dfrac{1}{N}\|\eta_t(Y)-X\|^2$，则

$$\begin{aligned}\mathrm{ET}(t)&=\frac{1}{N}E\|\eta_t(Y)-X\|^2\\&=\mathrm{ER}(t)+\sigma_n^2-\frac{2}{N}E<V,\eta_t(Y)>\\&=\frac{1}{N}E[\|\eta_t(Y)-X\|^2+\|X-Y\|^2+2<\eta_t(Y)-X,X-Y>]\end{aligned}$$

最后可得到风险函数的表达式

$$\begin{aligned}\mathrm{ER}(t)&=\mathrm{ET}(t)-\sigma_n^2+\frac{2\sigma_n^2}{N}\sum_{i=1}^{N}I(|Y_i|>t)\\&=\frac{1}{N}\sum_{i=1}^{N}I(|Y_i|\wedge t)^2+\sigma_n^2-\frac{2\sigma_n^2}{N}\sum_{i=1}^{N}I(|Y_i|<t)\end{aligned}$$

式中，$\wedge$ 为示性函数，I 为两数取小。于是，最佳的阈值选择可通过最小化风险函数取得，即

$$t^* = \arg\min_{t>0} \mathrm{ER}(t)$$

5.2.4 信号去噪原理

小波去噪的基本原理可归纳为如下 3 个模型。

1. 层内模型

目前最常使用的层内系数模型是广义高斯分布模型，这是对自然信号在同一层内的小波系数分布进行统计得到的规律，Laplace 分布和高斯分布是它的两个特例。

Hansen 等人利用与自然信号对应的小波系数近似服从 Laplace 分布的特点，在小波系数是独立同分布的假设下，得到利用带噪声信号小波系数来估计原信号的公式，并用于去噪，收到了非常好的效果。稍微复杂些的模型则将信号小波系数看成是独立的，但非同分布。

事实上，由于小波变换的去相关作用，相邻小波系数之间并无明显的关系，但是小波系数的绝对值或平方值却是符合 Markov 分布。Mihcak 等人由此提出小波系数的层内混合模型，在此模型中，原信号小波系数被用作一个双随机过程，其方差与局部高度相关，但在给定方差时，则成为相互独立且均值为零的高斯分布；而 Chang 等人则打破空间位置的限制，从小波系数值邻域来考察方差的相关性，从而得到了具有很强的局部适应性的阈值方法。

2. 层间模型

层间模型描述的是跨尺度小波系数之间的关系。较早但相对粗糙的模型是由 Shapiro 提出的，以这种模型为基础的零树编码方法已经在图像编码中得到了广泛的应用。

这种模型的主要特点是认为在小尺度上，较小的小波系数，其子孙很有可能也较小；而 Baraniuk 等人则通过隐式马氏链的转移矩阵将这种关系量化。在小波去噪中，这个特征可以用来区分图像边缘和由噪声引起的伪边缘。

3. 混合模型

混合模型主要综合了上述两种模型，它既考虑了层间系数之间的相关性，又顾及层内系数的关系，由于小波系数层实际上对应一个尺度，所以这种模型也称为空间-尺度混合模型。Liu 等人通过设定一个阈值 T，并将大于 T 的系数称为重要(significant)的。由此根据其父波是否重要将小波系数分成 C_{sig} 和 C_{insig} 两类，并假定 C_{sig} 服从指定分布，C_{insig} 则服从方差局部高度相关的高斯分布，最后得到新的混合模型。其与隐马氏树(hidden Markov tree，HMT)模型相比，新模型考虑了层内系数之间的关系；而与层内模型相比，则能更好地提示信号特征在跨尺度情形下的行为。

另外，Baraniuk 等人将小波系数分布几种状态（如小和大），并通过考察小波系数状态在层内和层间的变化，提出了独立状态模型（independent model，IM）、隐马氏链（hidden Markov chain，HMC）模型和隐马氏树模型（HMT）3 种模型，统称为小波马尔可夫模型。前两者实际上属于层内模型，而后一种则属于层间模型。

5.2.5 MATLAB 用于信号去噪

1. MATLAB 中用于去噪的函数

一般来说，小波分析用于去噪的过程分为 3 个主要步骤，MATLAB 提供了灵活的使用方式，既可以通过某种通用的方式直接对信号去噪，也可以对其中的某个环节灵活控制。MATLAB 用于小波去噪的函数的主要有以下几类。

（1）自动对信号进行去噪：wden，wdencmp。

（2）对阈值进行处理的函数：thselect，wthrmngr。

（3）根据信号噪声强度求得阈值：ddencmp，wbpen，wdcmb。

（4）直接对分解系数作用阈值的函数：wpthcoef，wthcoef，wthresh。

（5）估计噪声的函数：wnoisest。

（6）生成噪声的函数：wnoise。

1）wden 函数

MATLAB 中，提供了 wden 函数用于对一维信号进行自动消噪。函数的调用格式为：

[XD，CXD，LXD] = wden(X，TPTR，SORH，SCAL，N，'wname')：参数 X 为需要消噪的一维信号；参数 TPTR 为阈值选择规则，包括 rigrsure（自适应阈值选择）、heursure（启动式阈值选择）、sqtwolog（阈值为 sqrt(2 * log(length(X)))）和 minimaxi（极大极小值阈值选择）；参数 SORH 的值为 s（软阈值）或 h（硬阈值）；参数 SCAL 用于控制阈值的重新调整，包括 one（不调整）、sln（根据第一层的系数估计的噪声水平调整阈值）和 mln（根据不同层次的噪声水平调整阈值）；参数 N 为小波分解的层数；参数 wname 为使用的小波函数名称。函数返回消噪后的信号 XD 和消噪后信号的小波分解结构 CXD 和 LXD。

[XD，CXD，LXD] = wden(C，L，TPTR，SORH，SCAL，N，'wname')：参数 C 和 L 为消噪信号的小波分解结构。

【例 5-6】 利用 wden 函数对信号进行自动消噪。

```
>> clear all;
snr = 3; init = 2055615866;
%创建一个标准的高斯白噪声
[xref,x] = wnoise(3,11,snr,init);
%一维信号的自动消噪处理
lev = 5;
xd = wden(x,'heursure','s','one',lev,'sym8');
%绘制信号
subplot(711), plot(xref);
```

```
axis([1 2048 -10 10]);                          %标注坐标
title('原始信号');
subplot(712), plot(x);
axis([1 2048 -10 10]);
title(['噪声信号-信号比',...
num2str(fix(snr))]);
subplot(713), plot(xd);
axis([1 2048 -10 10]);
title('去噪后的信号-启动式阈值选择 SURE');
%使用软阈值消噪
xd = wden(x,'heursure','s','one',lev,'sym8');
%绘制信号
subplot(714), plot(xd);
axis([1 2048 -10 10]);
title('去噪后的信号 - SURE');
%采用阈值 sqrt(2*log)去噪
xd = wden(x,'sqtwolog','s','sln',lev,'sym8');
%绘制信号
subplot(715), plot(xd);
axis([1 2048 -10 10]);
title('去噪后的信号- 固定式阈值');
%采用极大极小阈值去噪
xd = wden(x,'minimaxi','s','sln',lev,'sym8');
%绘制信号
subplot(716), plot(xd);
axis([1 2048 -10 10]);
title('去噪后的信号-极大极小阈值');
%信号分解
[c,l] = wavedec(x,lev,'sym8');
%阈值的分解结构[C,L]
xd = wden(c,l,'minimaxi','s','sln',lev,'sym8');
subplot(717), plot(xd);
axis([1 2048 -10 10]);
title('去噪后的信号-信号分解');
```

运行程序,效果如图 5-10 所示。

2) wdencmp 函数

MATLAB 中,提供了 wdencmp 函数使用小波对信号进行消噪或压缩。函数的调用格式为:

[XC,CXC,LXC,PERF0,PERFL2] = wdencmp('gbl',X,'wname',N,THR,SORH,KEEPAPP):对输入信号 X(一维或二维)进行消噪或压缩后,返回 XC(消噪或压缩后的结果)。其中,wname 为指定所用的小波函数,'gbl'(global 的缩写)表示各层都是用同一个阈值处理。输出参数[CXC,LXC]为 XC 的小波分解结构。PERF0 和 PERFL2 是恢复和压缩 L^2 范数百分比。如果[C,L]是 X 的小波分解结构,则 PERFL2 = 100(CXC 向量的范数/C 向量的范数)^2;如果 X 是一个一维信号,小波 wname 是一个正交小波,则 PERFL2 = $\frac{100 \parallel XC \parallel^2}{\parallel X \parallel^2}$。N 为小波分解的层数,wname 为一个包含小波名的字符串,SORH 是软阈值或硬阈值的选择('s'或'h')。如果 KEEPAPP=1,则低频系数不进行阈值量化(也就是说系数不会被改变),反之,低频系数要进行阈值量化(即系数会

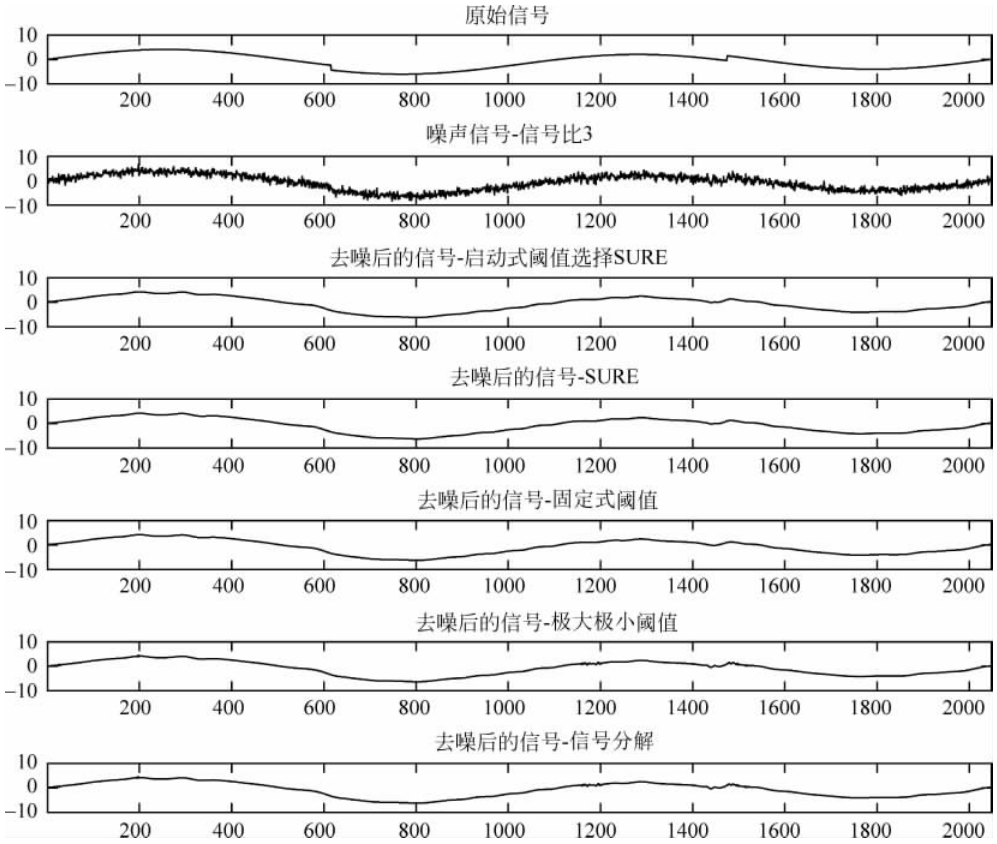

图 5-10　一维信号的消噪效果

被改变)。

[XC,CXC,LXC,PERF0,PERFL2] = wdencmp('lvd',X,'wname',N,THR,SORH)和[XC,CXC,LXC,PERF0,PERFL2] = wdencmp('lvd',C,L,'wname',N,THR,SORH):对一维情况和'lvd'(level-dependent,即每层用一个不同的阈值)选项,如果用相同的输入选项,这两种调用格式都具有相同的输出变量,但是每层必须都要有一个阈值,故阈值向量 THR 的长度为 N。另外,低频系数被保存,与 wden 函数相比,wdencmp 更灵活,可按照用户的消噪方式来消噪。对二维情况和'lvd'选项,这两种调用格式中的 THR 必须是一个三维矩阵,其含有水平、对角线、垂直 3 个方向的独立阈值,且长度为 N。

【例 5-7】 利用 wdencmp 函数对图像实现消噪处理。

```
>> clear all;
load sinsin;                                    % 载入信号
Y = X + 18 * randn(size(X));                    % 添加噪声
[thr,sorh,keepapp] = ddencmp('den','wv',Y);
xd = wdencmp('gbl',Y,'sym4',2,thr,sorh,keepapp);
subplot(221)
imagesc(X); title('原始图像');
subplot(222);
```

```
imagesc(Y); title('带噪声图像');
subplot(223)
imagesc(xd); title('消噪后图像');
```

运行程序,效果如图 5-11 所示。

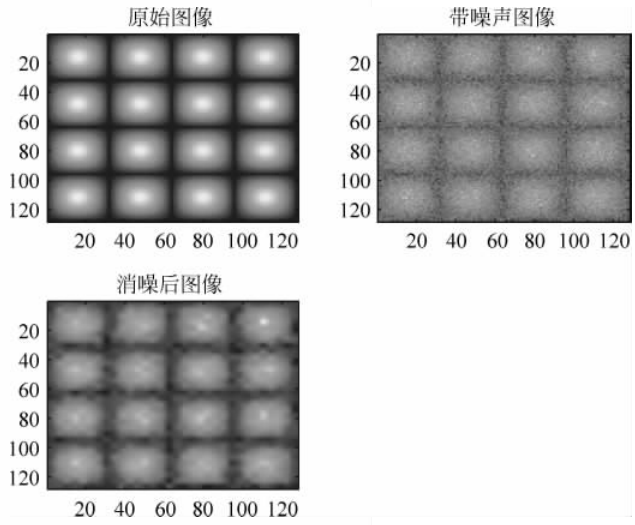

图 5-11　图像的消噪处理

3) thselect 函数

该函数用于信号的消噪阈值的选择。其调用格式为:

THR=thselect(X,TPTR):根据信号 X 和一个阈值选择标准来确定一个消噪的阈值。其返回的是对 X 进行消噪处理时所采用的自适应阈值。阈值的选取规则由字符串 TPTR 定义,其取值包括以下 4 种。

(1) TPTR = 'rigrsure',自适应阈值选择使用 Stein 的无偏风险估计原理。

(2) TPTR = 'heursure',使用启发式阈值选择。

(3) TPTR = 'sqtwolog',阈值等于 sqrt(2 * log(length(X)))。

(4) TPTR = 'minimaxi',用极大极小原理选择阈值。

阈值选择规则基于基本模型 $y=f(t)+e$,其中 e 为白噪声 $N(0,1)$,对于方差未知的噪声或非白噪声可重新调节输出阈值 THR。

用户可选择以下几种阈值。

(1) tptr = 'rigrsure',为一种基于史坦的无偏似然估计(二次方程)原理的自适应阈值选择。对一个给定的阈值 t,得到它的似然估计,再将非似然 t 最小化,就得到了所选的阈值,其是一种软件阈值估计器。

(2) tptr = 'sqtwolog',采用的是固定的阈值形式,产生的阈值大小是 sqrt(2 * log(length(X)))。

(3) tptr = 'heursure',为启发式阈值选择,是最优预测变量阈值选择。如果信噪比很小,SURE 估计有很大的噪声,在这种情况下,就采用这种固定的阈值。

(4) tptr = 'minimaxi',采用极大极小原理选择阈值,其产生一个最小均方误差的极值,而不是无误差。在统计学上,这种极值原理用于设计估计器。因为已经被消噪的信

号可以看作与未知回归函数的估计式相似，这种极值估计器可以在一个给定函数集中实现最大均方最小化。

【例 5-8】 对高斯白噪声的消噪阈值进行不同的选择。

```
>> clear all;
% 产生高斯白噪声
init = 2055415866; randn('seed',init);
x = randn(1,1000);
% 对每一个阈值选取规则,求取阈值
% 采用 SURE 的自适应阈值
thr1 = thselect(x,'rigrsure')
thr1 =
    1.8065
>>                                          % 固定形式的阈值
thr2 = thselect(x,'sqtwolog')
thr2 =
    3.7169
>>                                          % 启发式阈值
thr3 = thselect(x,'heursure')
thr3 =
    3.7169
>>                                          % 极大极小原理阈值
thr4 = thselect(x,'minimaxi')
thr4 =
    2.2163
```

从计算的结果来看，对于 Stein 的无偏似然估计(SURE)和极大极小(minimaxi)原理的阈值选择规则，仅保存了约 3%的系数；而其两种阈值选择规则，将所有的系数都变成了零。

同样地，对噪声进行小波分解时，也会产生高频系数，故一个信号的高频系数向量是有用信号和噪声信号的高频系数的叠加。由于 SURE 和 minimaxi 阈值选取规则较为保守(仅将部分系数置为零)，因此在信号的高频信息有很少一部分在噪声范围内时，这两种阈值非常有用，可以将弱小的信号提取出来。其他两种阈值选取规则，在去除噪声时更为有效，但是也可能将有用信号的高频部分当作噪声信号去除掉。

在实际的工程应用中，大多数信号可能包含许多尖峰或突变，而且噪声信号也并不是平稳的白噪声。对这种信号进行消噪处理时，传统的傅里叶变换完全是在频域中对信号进行分析，它不能给出信号在某个时间点上的变化情况，因此分辨不出信号在时间轴上的任何一个突变。但是小波分析能同时在时频域内对信号进行分析，所以它能有效地区分信号中的突变部分和噪声，从而实现对非平稳信号的消噪。

4) wthrmngr 函数

THR = wthrmngr(OPTION,METHOD,VARARGIN)：使用 OPTION 选项返回一个全局阈值或各层不同的阈值。输入 VARARGIN 依赖于 OPTION 和 METHOD 的取值。

该 M 文件返回的阈值，可用于整个 MATLAB 小波工具箱中的去噪和压缩工具。

METHOD 取值选项及说明如表 5-2 所示。

表 5-2 METHOD 取值及说明

METHOD 取值	说　明
'scarcehi'	当使用参数 M 的'high'默认值时，参见 wdcbm 或 wdcbm2
'scarceme'	当使用参数 M 的'medium'默认值时，参见 wdcbm 或 wdcbm2
'scarcelo'	当使用参数 M 的'low'默认值时，参见 wdcbm 或 wdcbm2
'sqtwolog'	参见 thselect 的'sqtwolog'选项，也可参见 wden
'sqtwologswn'	参见 thselect 的'sqtwolog'选项，当使用'sln'也选项时，可参见 wden
'rigsure'	参见 thselect 的'rigsure'选项，也可参见 wden
'heursure'	参见 thselect 的'heursure'选项，也可参见 wden
'minimaxi'	参见 thselect 的'minimaxi'选项，也可参见 wden
'penalhi'	当使用参数 ALPHA 的'high'默认值时，参见 wbmpen 或 wpbmpen
'penalme'	当使用参数 ALPHA 的'medium'默认值时，参见 wbmpen 或 wpbmpen
'penallo'	当使用参数 ALPHA 的'low'默认值时，参见 wbmpen 或 wpbmpen
'rem_n0'	该函数返回接近于 0 的阈值，THR 的典型值是 median(abs(coefficients))
'bal_sn'	该选项返回一个阈值，这样保留的能量和零点数的百分比是一致的
'sqrtbal_sn'	该选项返回一个等于该值方根的阈值，这样保留的能量和零点数的百分比是一致的

下面针对信号或图像在压缩或去噪时，使用的不同小波情况，给出了该函数的使用格式。

(1) 对于一维离散小波：

① 使用全局阈值进行压缩，X 是被压缩的信号，[C,L]是 X 的小波分解结构。

THR = wthrmngr('dw1dcompGBL','rem_n0',X)

THR = wthrmngr('dw1dcompGBL','bal_sn',X)

② 使用各层不同的阈值进行压缩，X 是被压缩的信号，[C,L]是 X 的小波分解结构，ALFA 为稀疏参数。

THR = wthrmngr('dw1dcompLVL','scarcehi',C,L,ALFA)，2.5 < ALFA < 10

THR = wthrmngr('dw1dcompLVL','scarceme',C,L,ALFA),1.5 < ALFA < 2.5

THR = wthrmngr('dw1dcompLVL','scarcelo',C,L,ALFA),1 < ALFA < 2

③ 使用各层不同的阈值进行去噪，[C,L]是被去噪信号的小波分解结构，SCAL 定义了修改后的乘性阈值，ALFA 为稀疏参数。

THR = wthrmngr('dw1ddenoLVL','sqtwolog',C,L,SCAL)

THR = wthrmngr('dw1ddenoLVL','rigrsure',C,L,SCAL)

THR = wthrmngr('dw1ddenoLVL','heursure',C,L,SCAL)

THR = wthrmngr('dw1ddenoLVL','minimaxi',C,L,SCAL)

THR = wthrmngr('dw1ddenoLVL','penalhi',C,L,ALFA),2.5 < ALFA < 10

THR = wthrmngr('dw1ddenoLVL','penalme',C,L,ALFA),1.5 < ALFA < 2.5

THR = wthrmngr('dw1ddenoLVL','penallo',C,L,ALFA),1 < ALFA < 2

(2) 对于一维离散平稳小波：使用各层不同的阈值进行去噪，SWTDEC 是被去噪信号的平稳小波分解结构，SCAL 定义了修改后的乘性阈值，ALFA 为稀疏参数。

THR = wthrmngr('sw1ddenoLVL',METHOD,SWTDEC,SCAL)

THR = wthrmngr('sw1ddenoLVL',METHOD,SWTDEC,ALFA)

(3) 对于二维离散小波：

① 使用全局阈值进行压缩，X 是被压缩的信号，[C,S]是 X 的小波分解结构。

THR = wthrmngr('dw2dcompGBL','rem_n0',X)

THR = wthrmngr('dw2dcompGBL','bal_sn',C,S)

THR = wthrmngr('dw2dcompGBL','sqrtbal_sn',C,S)

② 使用各层不同的阈值进行压缩，X 是被压缩的信号，[C,S]是 X 的小波分解结构，ALFA 为稀疏参数。

THR = wthrmngr('dw2dcompLVL','scarcehi',C,S,ALFA)，2.5 < ALFA < 10

THR = wthrmngr('dw2dcompLVL','scarceme',C,S,ALFA)，1.5 < ALFA < 2.5

THR = wthrmngr('dw2dcompLVL','scarcelo',C,S,ALFA)，1 < ALFA < 2

③ 使用各层不同的阈值进行去噪，[C,S]是被去噪图像的小波分解结构，SCAL 定义了修改后的乘性阈值，ALFA 为稀疏参数。

THR = wthrmngr('dw2ddenoLVL','penalhi',C,S,ALFA),2.5 < ALFA < 10

THR = wthrmngr('dw2ddenoLVL','penalme',C,S,ALFA),1.5 < ALFA < 2.5

THR = wthrmngr('dw2ddenoLVL','penallo',C,S,ALFA),1 < ALFA < 2

THR = wthrmngr('dw2ddenoLVL','sqtwolog',C,S,SCAL)

THR = wthrmngr('dw2ddenoLVL','sqrtbal_sn',C,S)

(4) 对于二维平稳小波：使用各层不同的阈值进行去噪，SWTDEC 为被去噪图像的平稳小波分解结构，SCAL 定义了修改后的乘性阈值，ALFA 为稀疏参数。

THR = wthrmngr('sw2ddenoLVL',METHOD,SWTDEC,SCAL)

THR = wthrmngr('sw2ddenoLVL',METHOD,SWTDEC,ALFA)

(5) 对于一维离散小波包：

① 使用全局阈值进行压缩，X 为被压缩的信号，WPT 是 X 的小波包分解结构。

THR = wthrmngr('wp1dcompGBL','bal_sn',WPT)

THR = wthrmngr('wp1dcompGBL','rem_n0',X)

② 使用全局阈值进行去噪，WPT 为被去噪信号 X 的小波包分解结构。

THR = wthrmngr('wp1ddenoGBL','sqtwologuwn',WPT)

THR = wthrmngr('wp1ddenoGBL','bal_sn',WPT)

THR = wthrmngr('wp1ddenoGBL','penalhi',WPT)

THR = wthrmngr('wp1ddenoGBL','penalme',WPT)

THR = wthrmngr('wp1ddenoGBL','penallo',WPT)

(6) 对于二维离散小波包：

① 使用全局阈值进行压缩，X 是被压缩的图像，WPT 是 X 的小波包分解结构。

THR = wthrmngr('wp2dcompGBL','bal_sn',WPT)

THR = wthrmngr('wp2dcompGBL','rem_n0',X)

THR = wthrmngr('wp2dcompGBL','sqrtbal_sn',WPT)

② 使用全局阈值进行去噪，WPT 为被去噪图像 X 的小波包分解结构。

THR = wthrmngr('wp2ddenoGBL','sqtwologswn',WPT)

THR = wthrmngr('wp2ddenoGBL','sqrtbal_sn',WPT)

THR = wthrmngr('wp2ddenoGBL','penalhi',WPT)

THR = wthrmngr('wp2ddenoGBL','penalme',WPT)

THR = wthrmngr('wp2ddenoGBL','penallo',WPT)

【例 5-9】 利用 wthrmngr 函数进行阈值管理。

```
>> clear all;
load noisbloc
%利用 db1 小波函数对信号进行 5 层分解
swc = swt(noisbloc,5,'db1');
SWC = swc;
%细节系数的 sqtwolog 阈值
ThreshSL = wthrmngr('sw1ddenoLVL','sqtwolog',swc,'sln');
%水平依赖阈值
ThreshML = wthrmngr('sw1ddenoLVL','sqtwolog',swc,'mln');
%使用硬阈值
%基于一个单一的电平阈值
for j = 1:5
swc(j,:) = wthresh(swc(j,:),'h',ThreshSL(1));
end
%电平相关的阈值
for j = 1:5
SWC(j,:) = wthresh(SWC(j,:),'h',ThreshML(j));
end
%重建
xSL = iswt(swc,'db1');
xML = iswt(SWC,'db1');
subplot(211);plot(noisbloc);
hold on;
plot(xSL,'r','linewidth',2);
title('单一阈值');
subplot(212);plot(noisbloc);
hold on;
plot(xML,'r','linewidth',2);
title('水平依赖阈值');
```

运行程序,效果如图 5-12 所示。

5) ddencmp 函数

ddencmp 函数可用于信号去噪或压缩的默认阈值的获取。函数的调用格式为:

[THR,SORH,KEEPAPP,CRIT] = ddencmp(IN1,IN2,X):返回小波或小波包对输入向量或矩阵 X 进行压缩或消噪的默认值。参量 THR 为阈值;参量 SORH 表示软、硬阈值;参量 KEEPAPP 允许保留近似系数;参量 CRIT 表示熵名(只用于小波包)。输入参量 IN1 为'den'时表示消噪,为'cmp'时表示压缩;当 IN2 为'wv'时表示小波,为'wp'时表示小波包。

[THR,SORH,KEEPAPP] = ddencmp(IN1,'wv',X):IN1='den'时,返回 X 消噪的默认值;IN1='cmp'时,返回 X 压缩的默认值。这些值可应用于 wdencmp 函数。如果输入为一个小波包,即输出 4 个参量。

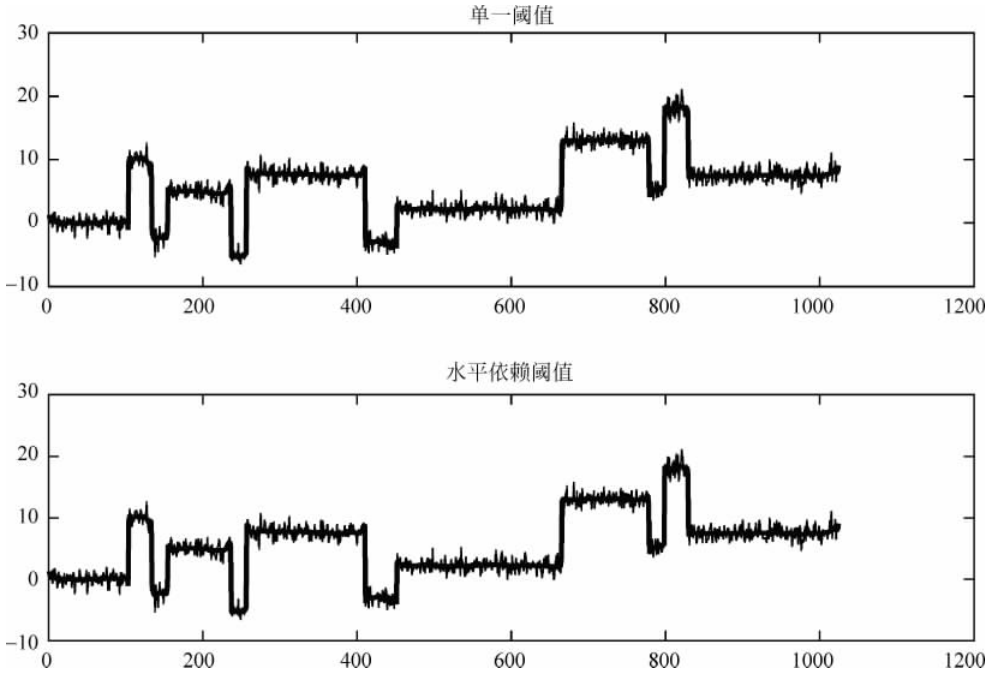

图 5-12　阈值管理

[THR,SORH,KEEPAPP,CRIT] = ddencmp(IN1,'wp',X)：IN1='den'时，返回X消噪的默认值；IN1='cmp'时，返回X压缩的默认值。这些值可应用于wpdencmp函数。

【例 5-10】 利用ddencmp函数获取小波消噪的阈值。

```
>> clear all;
% 产生高斯白噪声
init = 2055415866; randn('seed',init);
x = randn(1,1000);
% 求取小波分析的默认值(3 个输出变量)
% 这些值可以应用于 wdencmp(带参数'gb1')
% 求取信号消噪的默认阈值、软阈值,并且保留低频系数
[thr1,sorh1,keepapp1] = ddencmp('den','wv',x)
thr1 =
    3.8593
sorh1 =
         s
keepapp1 =
     1
>> % 求取信号压缩的默认阈值、硬阈值,并且保留低频系数
[thr2,sorh2,keepapp2] = ddencmp('cmp','wv',x)
thr2 =
    0.7003
sorh2 =
h
keepapp2 =
     1
```

```
>> %求取小波包分析的默认值(4个输出变量),这些值可应用于wdencmp
[thr3,sorh3,keepapp3,crit3] = ddencmp('den','wp',x)
thr3 =
    4.2911
sorh3 =
h
keepapp3 =
     1
crit3 =
sure
>> %求取信号压缩的默认阈值、硬阈值,并且保留低频系数
%默认的熵为"threshold"标准
[thr4,sorh4,keepapp4,crit4] = ddencmp('cmp','wp',x)
thr4 =
    0.7003
sorh4 =
h
keepapp4 =
     1
crit4 =
threshold
```

6）wbmpen 函数

wbmpen 函数可用于一维和二维信号去噪阈值的选择。函数的调用格式为：

THR = wbmpen(C,L,SIGMA,ALPHA)：参数 C 和 L 为尺度 length(L)－2 上的去噪信号的小波分解结构，算法参数 ALPHA 为大于 1 的实数，返回选择的阈值 THR。

wbmpen(C,L,SIGMA,ALPHA,ARG)：参数 ARG 为大于 1 的实数。

【例 5-11】 使用 wbmpen 函数实现信号阈值消噪处理。

```
>> clear all;
load leleccum;
ind = 1:1024;
x = leleccum(ind);
%产生含噪声信号
init = 2055615866;
randn('seed',init);
nx = x + 20 * randn(size(x));
%用小波函数 db6 对信号进行 3 层分解
[C,L] = wavedec(nx,3,'db6');
%估计尺度 1 的噪声标准偏差
sigma = wnoisest(C,L,1);
alpha = 2;
%获取消噪过程中的阈值
thr = wbmpen(C,L,sigma,alpha)
keepapp = 1;
%对信号进行消噪
xd = wdencmp('gbl',C,L,'db6',3,thr,'s',keepapp);
subplot(2,2,1);plot(x);
title('原始信号');
subplot(2,2,2);plot(nx);
title('含噪信号');
subplot(2,2,3);plot(xd);
```

```
title('消噪后的信号')
```

运行程序,输出如下,效果如图 5-13 所示。

```
thr =
   70.4861
```

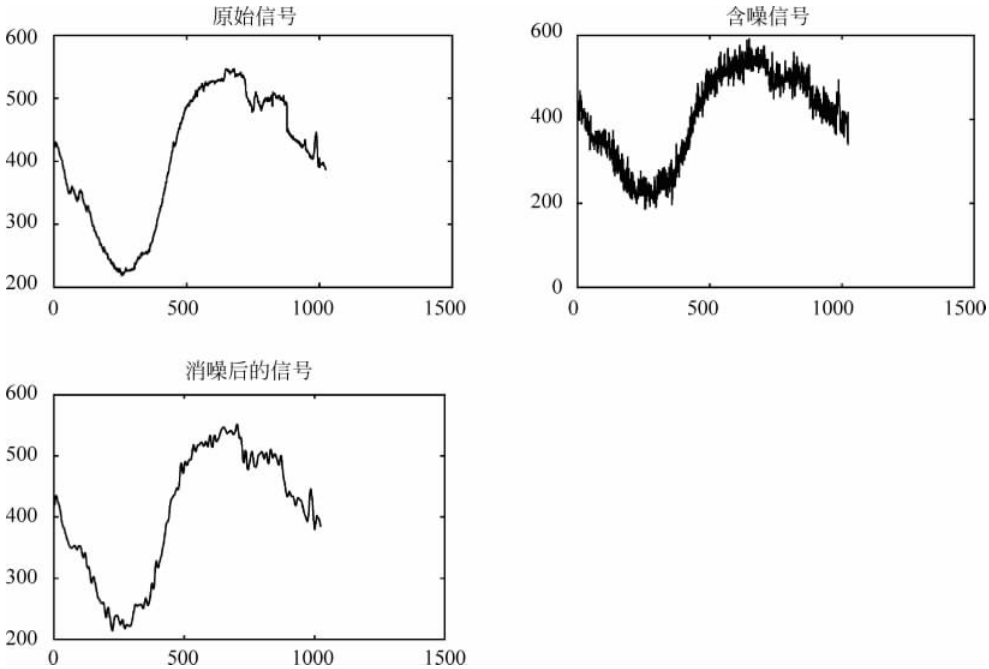

图 5-13　wbmpen 信号去噪处理

7) wdcbm 函数

MATLAB 中,提供了 wdcbm 函数用于使用 Birge-Massart 算法处理一维小波的阈值。函数的调用格式为:

[THR,NKEEP] = wdcbm(C,L,ALPHA,M):参数 C 和 L 为尺度 length(L)－2 上消噪信号的小波分解结构,算法参数 ALPHA 为大于 1 的实数,M 为大于 1 的实数;返回选择的阈值 THR 和系数的个数 NKEEP。

wdcbm(C,L,ALPHA):等价于 wdcbm(C,L,ALPHA,L(1))。

【例 5-12】 利用 wdcbm 函数使用 Birge-Massart 算法处理一维信号的阈值。

```
>> clear all;
 % 载入一个电信号并截取一部分
load leleccum; indx = 2600:3100;
x = leleccum(indx);
 % 用 db3 小波对信号进行 5 层分解
wname = 'db3'; lev = 5;
[c,l] = wavedec(x,lev,wname);
 % 用所给参数选择独立的阈值并进行信号压缩
alpha = 1.5; m = l(1);
[thr,nkeep] = wdcbm(c,l,alpha,m)
 % 用软阈值对信号进行压缩
```

```
[xd,cxd,lxd,perf0,perfl2] = wdencmp('lvd',c,l,wname,lev,thr,'h');
%绘制原始信号和压缩信号
subplot(211), plot(indx,x);
title('原始信号');
subplot(212), plot(indx,xd);
title('压缩后信号');
xlab1 = ['恢复百分比: ',num2str(perfl2)];
xlab2 = [' %   -- 压缩百分比: ',num2str(perf0), ' %'];
xlabel([xlab1 xlab2]);
```

运行程序,输出如下,效果如图 5-14 所示。

```
thr =
   19.5569   17.1415   20.2599   42.8959   15.0049
nkeep =
     1      2     3     4     7
```

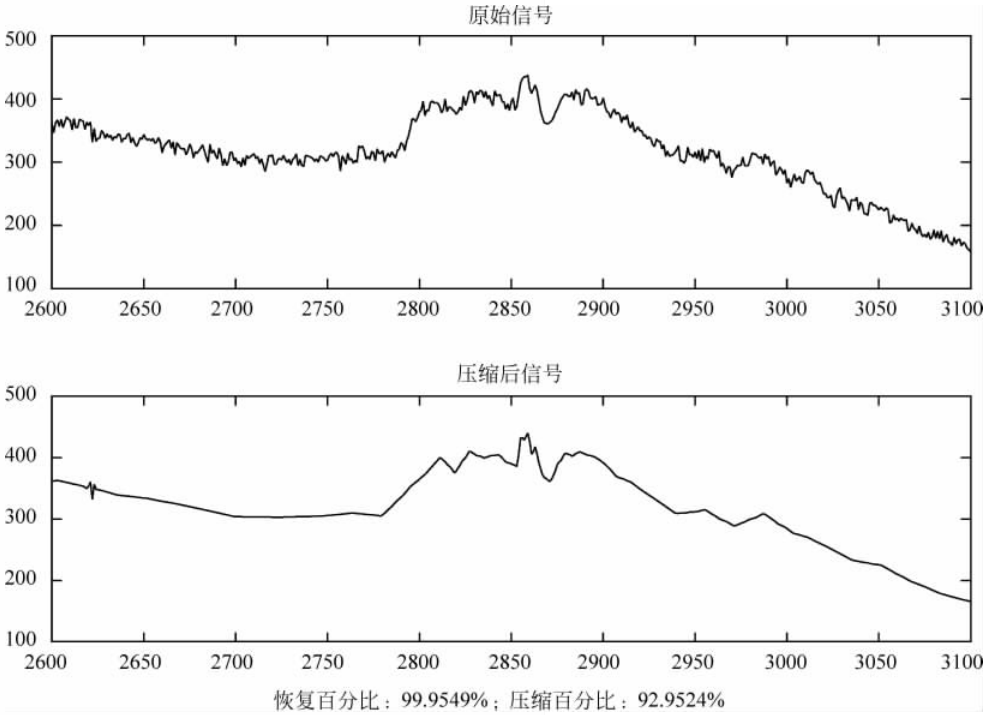

图 5-14　信号阈值

8) wpthcoef 函数

MATLAB 中,提供了 wpthcoef 函数进行小波包分解系数的阈值处理。函数的调用格式为:

NT = wpthcoef(T,KEEPAPP,SORH,THR):对信号的小波包树 T 进行系数的阈值处理后,返回一个新的小波包树 NT。如果 KEEPAPP=1,则近似系数不进行阈值处理,否则,进行处理。如果 SORH='s',使用软阈值;如果 SORH='h',则使用硬阈值。THR 为阈值的大小。

9) wthcoef 函数

MATLAB 中,提供了 wthcoef 函数对一维信号的小波系数进行阈值处理。函数的调用格式为:

NC = wthcoef('d',C,L,N,P):返回对小波分解结构[C,L]进行比例压缩后的小波分解向量 NC。其中,向量 N 定义待压缩的高频尺度,向量 P 定义把相应绝对值较小的系数置为 0 的百分比。

NC = wthcoef('d',C,L,N):返回小波分解结构[C,L]中,把 N 所指定尺度的高频系数全部置为 0 后的小波分解向量 NC。

NC = wthcoef('a',C,L):返回把小波分解结构[C,L]中低频系数全部置为 0 后的小波分解向量 NC。

NC = wthcoef('t',C,L,N,T,SORH):返回对小波分解结构[C,L]经过阈值处理后的小波分解向量 NC。N 为一个包含高频尺度的向量,T 为与尺度向量 N 相对应的阈值向量,它定义每个尺度相应的阈值,N 和 T 长度相等。参数 SORH 用来对阈值方式进行选择,当 SORH='s'时为软阈值,当 SORH='h'时为硬阈值。

【例 5-13】 对一维带噪信号进行消噪处理。

```
>> clear all;
% 载入一个带噪的信号
load noissin;
s = noissin(1:1000);
subplot(2,1,1);plot(s);
title('含噪原始信号');
% 用 sym2 小波对信号进行 3 层小波分解
[C,L] = wavedec(s,3,'sym2');
% 设置尺度向量 a
a = [1 2 3];
% 设置阈值向量 p
p = [97 98 95];
% 对高频系数进行阈值处理
nc = wthcoef('d',C,L,a,p);
% 对新的小波分解结构[nc,L]进行重构
sd = waverec(nc,L,'sym2');
subplot(2,1,2);plot(sd);
title('消噪后的信号')
```

运行程序,效果如图 5-15 所示。

10) wthresh 函数

该函数进行软阈值或硬阈值处理。其调用格式为:

Y = wthresh(X,SORH,T):该函数用于对信号 X 进行软阈值或硬阈值处理。向量 X 为待处理的信号,T 为阈值大小(T>0),SORH 参数用于软硬阈值的选择。

如果 SORH='s',则为软阈值处理。对于软阈值处理,Y=wthresh(X,'s',T)返回的是 $Y=\text{size}(X)\cdot(|X|-T)_+$,即把信号的绝对值与阈值进行比较,小于或等于阈值的点置为 0,大于阈值的点置为该点值与阈值的差值。

如果 SORH='h',则为硬阈值处理,对于硬阈值处理,$Y=X\cdot L_{(|X|>T)}$,即把信号的绝对值与阈值比较,小于或等于阈值的点置为 0,大于阈值的点保持不变。一般来说,用

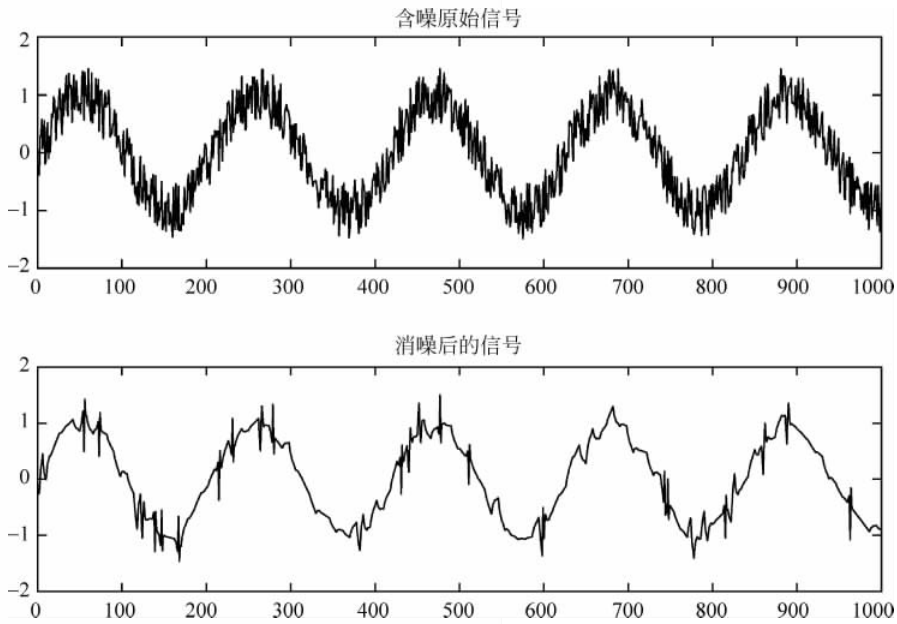

图 5-15　信号消噪处理

硬阈值处理后的信号比软阈值处理后的信号更粗糙。

【例 5-14】 对信号进行不同阈值处理。

```
>> clear all;
% 生成线性信号
y = linspace( - 1,1,100);
% 设置 T 阈值
thr = 0.4;
% 计算软、硬阈值
ythard = wthresh(y,'h',thr);
ytsoft = wthresh(y,'s',thr);
% 显示不同阈值后的信号
subplot(3,1,1); plot(y);
title('原始信号');
grid on;
subplot(3,1,2);plot(ythard);
title('硬阈值信号');
grid on;
subplot(3,1,3); plot(ytsoft);
title('软阈值信号');
grid on;
```

运行程序,效果如图 5-16 所示。

11) wnoisest 函数

MATLAB 中,提供了 wnoisest 函数用于估计一维小波系数的标准偏差。函数的调用格式为:

STDC = wnoisest(C,L,S):对输入信号向量 S 的各层节点,返回其高频系数的标准偏差的估计值,[C,L]为输入 S 的小波分解结构。这种估计值采用 Maximum 的绝对

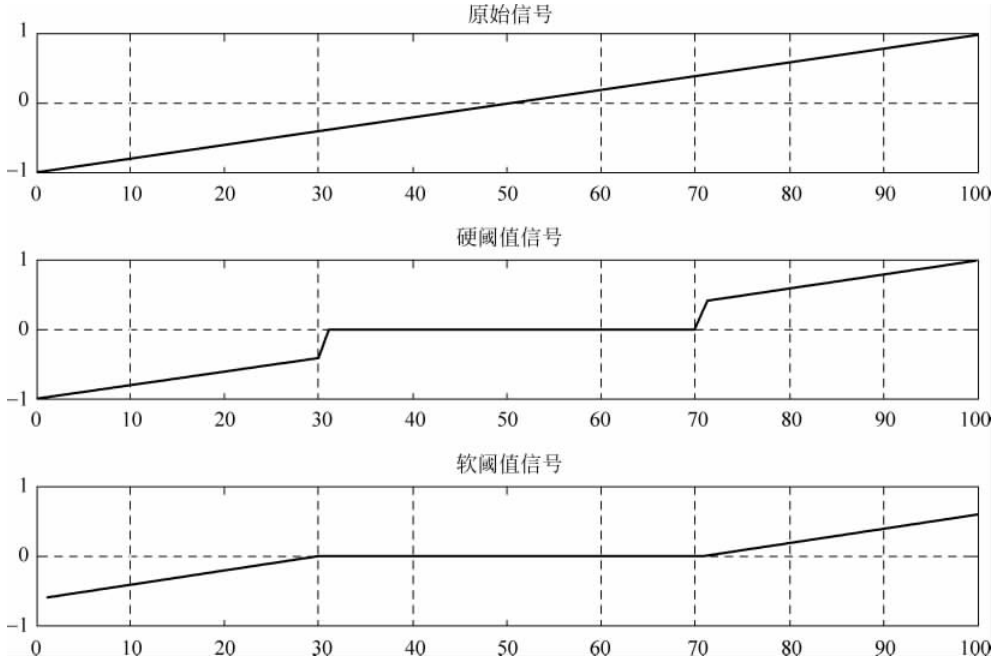

图 5-16　不同阈值下的信号

偏差除以 0.6745,它在一维零均值高斯白噪声的信号模型中进行消噪处理很有用。

【例 5-15】 估计一维小波系数的标准偏差。

```
>> clear all;
% 产生高斯白噪声
init = 2055415866; randn('seed',init);
x = randn(1,1000);
% 用 db3 小波对信号进行 2 层分解
[c,l] = wavedec(x,2,'db3');
% 估计第一层分解系数和第二层分解系数的标准偏差
% 由于信号 x 为单位方差白噪声,所以估计很接近 1
disp('显示估计标准偏差: ')
pc1 = wnoisest(c,l,1:2)
% 假设 x 包含 10 个非正常值
ind = 50:50:500;
x(ind) = 100 * ones(size(ind));
% 用 db3 小波对信号进行分解
[ca,cd] = dwt(x,'db3');
% 对 cd 标准偏差的鲁棒性估计,它过高估计了噪声水平
pc2 = std(cd)
% 对 cd 标准偏差的鲁棒性估计,使噪声水平接近 1
median(abs(cd))/0.6745
```

运行程序,输出如下:

```
显示估计标准偏差:
pc1 =
    1.0192    1.0922
pc2 =
```

```
    8.0209
ans =
    1.0601
```

12）wnoise 函数

MATLAB 中，提供了 wnoise 函数用于产生含噪声的测试函数数据。函数的调用格式为：

X = wnoise(FUN,N)：返回由 FUN 所给的测试信号，N 表示在[0,1]范围内采样个数为 2^N 个。

[X,XN] = wnoise(FUN,N,SQRT_SNR)：返回同上一个测试向量 X 和包含高斯白噪声 N(0,1)的测试向量 XN，XN 中的信噪比 SNR＝SQRT_SNR^2。参数 FUN 的取值为：

- FUN＝1 时，为 Blocks 噪声；
- FUN＝2 时，为 Bumps 噪声；
- FUN＝3 时，为 Heavy sine 噪声；
- FUN＝4 时，为 Doppler 噪声；
- FUN＝5 时，为 Quadchirp 噪声；
- FUN＝6 时，为 Mishmash 噪声。

[X,XN] = wnoise(FUN,N,SQRT_SNR,INIT)：设定发生器的值为 INIT。

【例 5-16】 利用 wnoise 函数创建各种类型噪声。

```
>> clear all;
% 产生一个采样点为 2^10 个的 Heavy sine 噪声
x = wnoise(3,10);
% 产生一个信噪比为 7,采样点个数为 2^10 的 Doppler 噪声
[x,noisyx] = wnoise(4,10,7);
% 产生一个 Doppler 信号
init = 2055415866;
[x,noisyx] = wnoise(4,10,7,init);
% 绘制所有的测试函数
ind = linspace(0,1,2^10);
for i = 1:6
    x = wnoise(i,10);
    subplot(6,1,i), plot(ind,x)
end
```

运行程序，效果如图 5-17 所示。

2. 通过抑制细节系数去噪

有了基本噪声模型和阈值的基础，通过手动添加的噪声来说明使用 MATLAB 小波工具箱去噪的方法，了解噪声被抑制的过程。

实例中，在一个光滑的信号上加入一个高斯白噪声，再对其进行小波分析，观察其信号在时间-频域上的成分。再通过作用阈值抑制噪声信号，重建信号，达到去噪的目的。

在小波分解过程中，每次分解得到的近似系数比以前更光滑，舍去的细节信息就存放在各层近似系数中，那么一个简单的思想就是重建第 i 层近似系数达到去噪的目的。

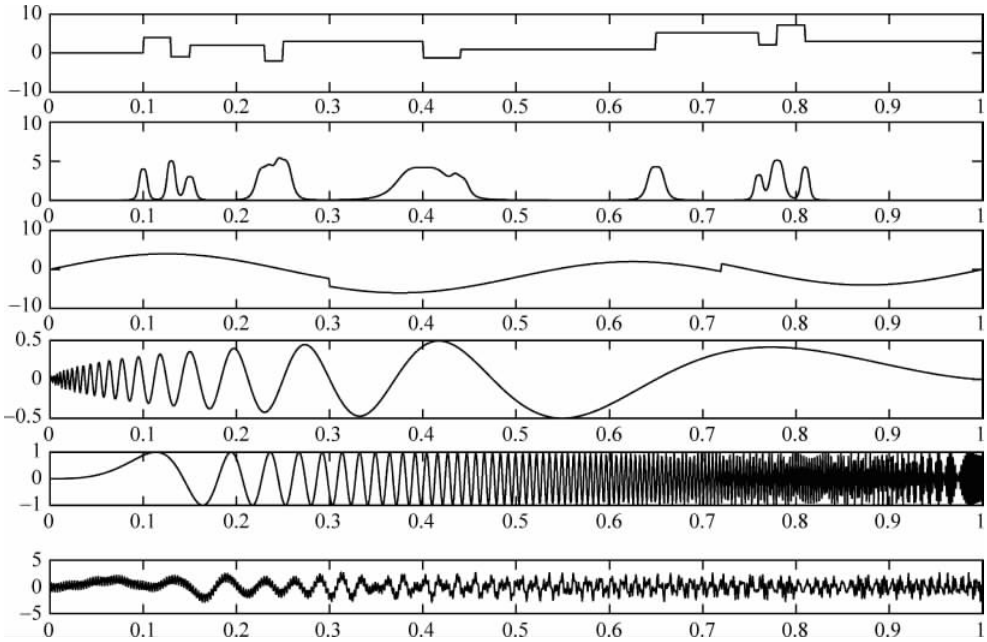

图 5-17　各类型噪声

那么进一步，为了保持原相对完整的信息，可以有选择地抑制各层的细节系数，也能实现这个目的。

【例 5-17】 通过抑制部分细节系数来实现信号的平滑。

```
>> clear all;
load noisdopp;   % 载入信号
x = noisdopp;
% 使用 db4 小波对其做 5 层分解
[c,l] = wavedec(x,5,'db4');
% 重建第 5 层近似信号
ca = wrcoef('a',c,l,'db4',5);
% 从 l 中读出所有细节系数所在下标
index = l(2) + 1:l(7);
% 得到一个 c 的复本
c1 = c;
% 对细节系数成分进行抑制
c1(index) = c(index)/3;
% 通过抑制后的系数重建小波
x2 = waverec(c1,l,'db4');
% 求去噪后的 ca 信号在原信号的能量成分
per1 = norm(ca)/norm(x)
% 求去噪后的 x2 信号在原信号中的能量成分
per2 = norm(x2)/norm(x)
% 将 ca 作为去噪信号，求其与原信号的标准差
err1 = norm(ca - x)
% 将 ca 作为去噪信号，求其与原信号的标准差
err2 = norm(x2 - x)
subplot(311);plot(x);
title('原始信号');
```

```
subplot(312);plot(ca);
title('重建后的第 5 层近似信号');
subplot(313);plot(x2);
title('通过抑制细节得到的去噪信号');
```

运行程序,输出如下,效果如图 5-18 所示。

```
per1 =
    0.9302
per2 =
    0.9387
err1 =
   48.8734
err2 =
   32.4658
```

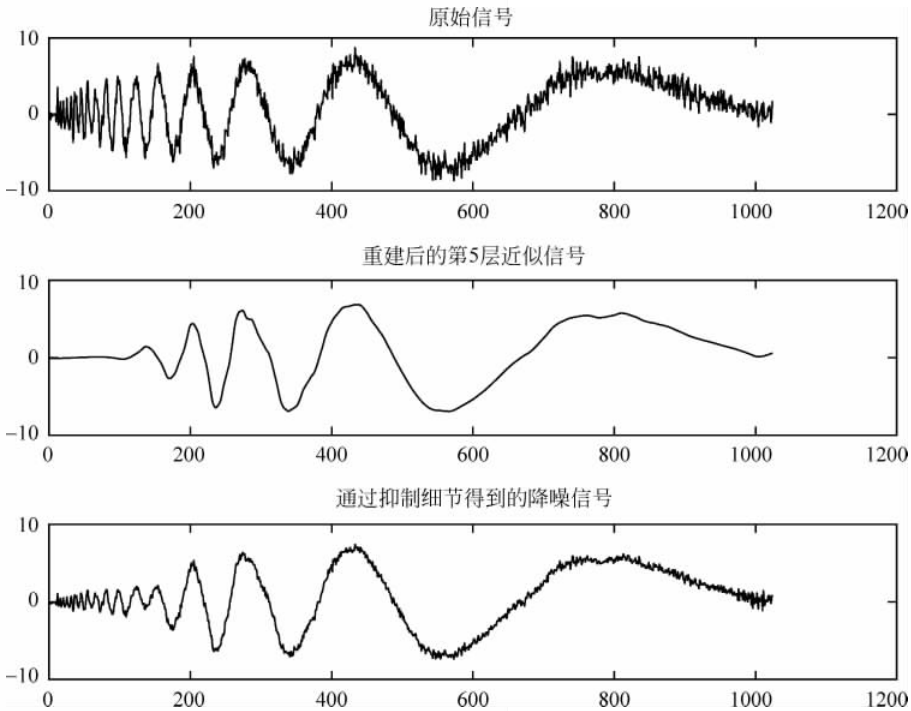

图 5-18　通过抑制系数的方法对信号去噪处理

从这个例子可看出,使用单纯的抑制细节系数的方式(因为重建近似信号等于将所有的系数抑制到 0),确实可以实现消除信号噪声的目的。但这种方式太过于粗略,因为这样并没有利用到噪声本身的信息,没有通过噪声本身来确定去噪的方法,所以作为衡量相似性的标准差仍然很大,而且去噪后的信号损失了很多原信号的能量成分(6%左右)。这就说明在去噪的过程中,不仅抑制了噪声,也抑制了很多有用的信息成分。

3. 通过 FFT 实现信号去噪

这里介绍使用 FFT 对信号去噪的方法,让读者更深刻地体会小波变换在时间频率域的处理较传统傅里叶分析只在频率进行变换的优点。

在小波域中的近似系数如果映射到傅里叶分析中的频域，则代表高频系数，如果只对高频系数进行抑制，同样可以达到去噪的效果。

具体步骤为：

(1) 对原始信号进行傅里叶变换，求出其频谱。

(2) 根据频谱，对比所关心的频谱成分，对不需要的频谱成分进行抑制。

(3) 对变换后的频谱作傅里叶逆变换，得到去噪后的信号。

这个过程其实就相当于对原信号在一定范围进行滤波，还原到时域则相当于对信号进行卷积运算。设原始信号为 $f(t)$，去噪后的信号为 $g(t)$。其傅里叶变换形式分别为 $F(\omega)$和 $G(\omega)$。那么这个过程可表示为

$$G(\omega) = H(\omega) \cdot F(\omega)$$

式中，$H(\omega)$为频域中的滤波器，用以抑制噪声信号的频谱。

【例 5-18】 用 FFT 方法实现抑制噪声，并给出其能量成分和标准差。

```
>> clear all;
load noisdopp;                                  %载入信号
x = noisdopp;
%信号长度为 1024,所以不用扩展直接进行长度为 1024 的快速傅里叶变换
y = fft(x,1024);
%求 y 的模平方
pyy = y. * conj(y);
%修改原坐标,512 以后的高频段,因为这是冗余信息
f = 1000 * (0:512)/1024;
plot(f,pyy(1:513));                             %绘制原信号频谱图,如图 5-18 所示
```

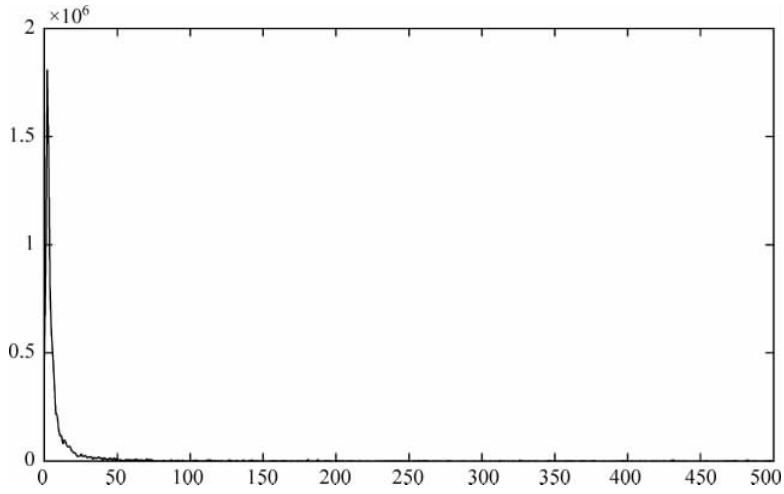

图 5-19 FFT 下的频谱图

由图 5-19 可看出，信号的能量主要集中在低频部分，在 20Hz 以后迅速衰减到零，50Hz 以后就几乎没有能量了。这样可以进行简单的低通滤波。

```
%使用不同宽度的滤波器对频谱进行滤波,抑制频谱直接令其为零
>> y1 = y;
y1(10:1014) = 0;
```

```
y2 = y;
y2(30:1004) = 0;
y3 = y;
y3(50:994) = 0;
% 对经过滤波的频谱做傅里叶逆变换,得到相应的去噪信号
xd1 = real(ifft(y1,1024));
xd2 = real(ifft(y2,1024));
xd3 = real(ifft(y3,1024));
% 绘制去噪后的波形,效果如图 5 - 20 所示
```

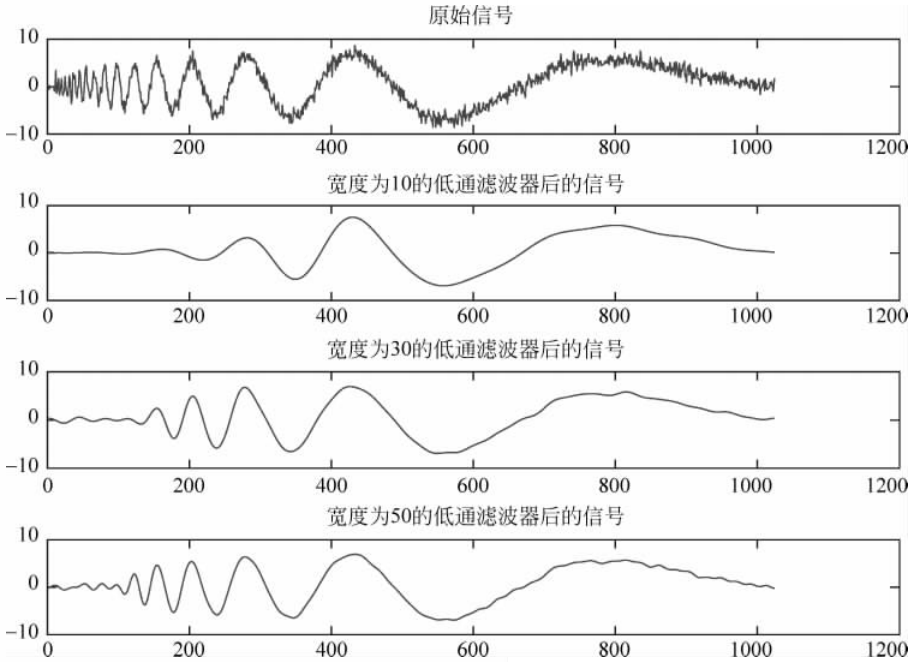

图 5-20　在频域使用低通滤波器的去噪效果

```
subplot(411);plot(x);title('原始信号');
subplot(412);plot(xd1);title('宽度为 10 的低通滤波器后的信号');
subplot(413);plot(xd2);title('宽度为 30 的低通滤波器后的信号');
subplot(414);plot(xd3);title('宽度为 50 的低通滤波器后的信号');
% 求各个去噪信号的能量比例
>> per1 = norm(xd1)/norm(x)
per1 =
    0.8710
>> per2 = norm(xd2)/norm(x)
per2 =
    0.9390
>> per3 = norm(xd3)/norm(x)
per3 =
    0.9542
% 求各个去噪信号与原信号的标准差
>> err1 = norm(xd1 - x)
err1 =
   62.6615
>> err2 = norm(xd2 - x)
err2 =
```

```
    43.3881
>> err3 = norm(xd3 - x)
err3 =
    36.8576
```

在实例中，信号 noisdopp 初始发展阶段的振荡频率很高，这被认为是系统自身的特性。但对于低通滤波器，这些成分被过滤掉了。所以单纯对频域滤波有“一刀切”的缺陷，也就是把带通之外的频谱不加区分地过滤掉。

将得到的结果同前面所用的小波域中抑制噪声的方法相比较，可以看出，傅里叶变换只能在频域范围内表述，对系数进行处理的手法也相对单一，而小波分解后可以在各个层次选择阈值，对噪声成分进行抑制，手段更加灵活。

还有一个现象值得注意，在使用小波变换进行噪声抑制时，去噪结果的能量比例(93.87%)虽然没有 FFT 滤波器的结果(95.42%)高，但是却保持了更高的与原信号的相似程度。这一点从去噪信号与原信号的标准差可看出。小波变换中，对细节系数进行抑制以后的滤波结果与原信号的标准差为 32.4658，比 FFT 的结果 36.8576 还小。而且这种整体缩减的抑制细节系数的方法还不是最好的去噪方法。这个实例客观地说明了多分辨分析在变换时对时间和频谱的兼顾，以及它与传统频域方法相比无可比拟的优势。

从通常意义上，去噪的两个准则，光滑性和相似性在时间和频率两个空间上体现的比重不同。时域分析，更容易体现信号的相似性，而不太好处理信号的光滑性，因为时域的分析可以很好地判断信号的动态性质；而在频域中，可以很方便地过滤掉高频的信号，使得信号无限光滑，但是在原信号中能量比重很小的很多有用的信号成分也可能因此被过滤掉。

在这个意义上，小波分析的多分辨概念有了广阔的应用空间。展开后的小波系数，可以根据某些频段的系数进行滤波，更重要的是可以根据某些频段分离出来的系数随时间的变化，通过某些准则来确定其是信号本身所包含的信息，还是在统计上无规律的噪声。有了这些非常符合实际应用的特性，小波变换才会被誉为“数学显微镜”。

4. MATLAB 默认的去噪函数

前面两小节从去噪的基本思想出发，讨论了在小波域和频域对信号噪声进行抑制的方法，并将两种方法得到的去噪结果进行了比较。但是从严格意义上讲，这些都不能很好地符合去噪所提出的两个基本要素——光滑性和相似性。前面介绍的阈值控制的方法在理论上指出了一种在小波域对系数进行操作，使得去噪信号最大程度地满足这两种要求的方法，就是所谓的“小波收缩”的方法。其原理就是前面介绍的根据方差最小的原则，通过对系数的偏似然估计确定阈值，这也是 MATLAB 小波工具箱中默认的去噪方法，关于这种方法的理论在此不展开介绍。

【例 5-19】 利用 MATLAB 默认的去噪函数实现信号的去噪处理。在实例中选择与 db4 相似，且能对称性更好的 sym4 小波。

```
>> clear all;
load noisdopp;                          %载入信号
x = noisdopp;
```

```
%根据信号计算噪声强度,给出全局阈值
[thr,sorh,keepapp] = ddencmp('den','wv',x);
%根据全局阈值对信号去噪
xd = wdencmp('gbl',x,'sym4',2,thr,sorh,keepapp);
%用 sym4 小波对信号做 4 层分解
[c,l] = wavedec(x,4,'sym4');
%得到每个层次的分层阈值
[thr1,nkeep] = wdcbm(c,l,2);
%根据分层阈值使用软阈值方法对信号去噪
[xd1,cxd,lxd,perf0,perfl2] = wdencmp('lvd',c,l,'sym4',4,thr1,'s');
%绘制原始信号和去噪后信号的图形,如图 5-21 所示
```

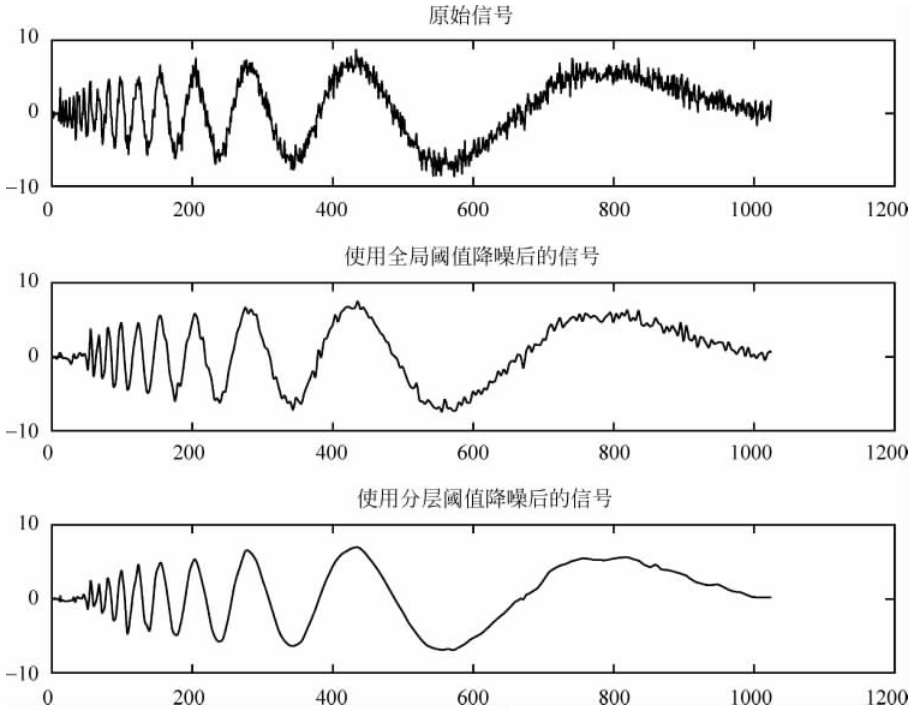

图 5-21　信号的去噪效果

```
subplot(311);plot(x);title('原始信号');
subplot(312);plot(xd);title('使用全局阈值去噪后的信号');
subplot(313);plot(xd1);title('使用分层阈值去噪后的信号');
%求得去噪信号的能量成分和与原信号的标准差
>> per1 = norm(xd)/norm(x)
per1 =
    0.9774
>> per2 = norm(xd1)/norm(x)
per2 =
    0.9645
>> err = norm(xd - x)
err =
   28.0714
>> err1 = norm(xd1 - x)
err1 =
   31.0548
```

由图 5-21 可看出，全局阈值和分层阈值方法去噪的信号都很好地保留了信号发展初期的高频特性，且性能参数优于抑制细节系数的策略和 FFT 方法。在这两者之间，分层阈值虽然损失了部分的性能(与原信号的相似性)，但比全局阈值的结果光滑很多。而且信号发展初期的高频系数也几乎不受影响，最大限度地反映了原信号本身的性质。

5.2.6 小波包进行信号去噪

小波包的信号去噪思想和小波分析基本相同。不同之处在于，小波包提供了一种更为复杂、更为灵活的分析手段，因为小波包分析对上一层的低频部分和高频部分同时实行分解，具有更加精确的局部分析能力。

对信号进行小波包分解时，可以采用多种小波包基，通常根据分析信号要求，从中选择最好的一种小波包基，即最优基。最优基的选择标准是熵标准。在 MATLAB 的小波工具箱中可通过 besttree 函数进行最优基的选择。

应用小波包分析对信号进行去噪处理是一个最基本的功能。小波包阈值去噪过程主要分为 4 个步骤：

(1) 信号的小波包分解。选择一种小波包基并确定所需分解的层次，然后对信号进行小波包分解。

(2) 最优小波包基的选择。

(3) 小波包分解系数的阈值化。对于每一个小波包分解系数，选择一个恰当的阈值对小波包分解后的系数进行阈值量化处理。

(4) 信号的小波包重构。对低频系数和经过处理后的高频系数进行小波包重构。

利用小波包进行去噪首先要选取小波包基和分解的层数。对称性好的小波不产生相位畸变，正则性好的小波易于获得光滑的重构信号。

【例 5-20】 利用小波包对一个给定的含噪信号进行去噪处理。

```
>> clear all;
load noisdopp;
s = noisdopp(1:1000);
subplot(511);plot(s);
title('原始信号');
[thr,sorh,deepapp,crit] = ddencmp('den','wp',s);
[x3,wpt,perf0,perf12] = wpdencmp(s,sorh,3,'sym6',crit,thr,deepapp);
subplot(512);plot(x3);
title('3 层分解去噪');
[x4,wpt,perf0,perf12] = wpdencmp(s,sorh,4,'sym6',crit,thr,deepapp);
subplot(513);plot(x4);
title('4 层分解去噪');
[x5,wpt,perf0,perf12] = wpdencmp(s,sorh,5,'sym6',crit,thr,deepapp);
subplot(514);plot(x5);
title('5 层分解去噪');
[x8,wpt,perf0,perf12] = wpdencmp(s,sorh,5,'sym6',crit,thr,deepapp);
subplot(515);plot(x8);
title('8 层分解去噪');
```

运行程序，效果如图 5-22 所示。

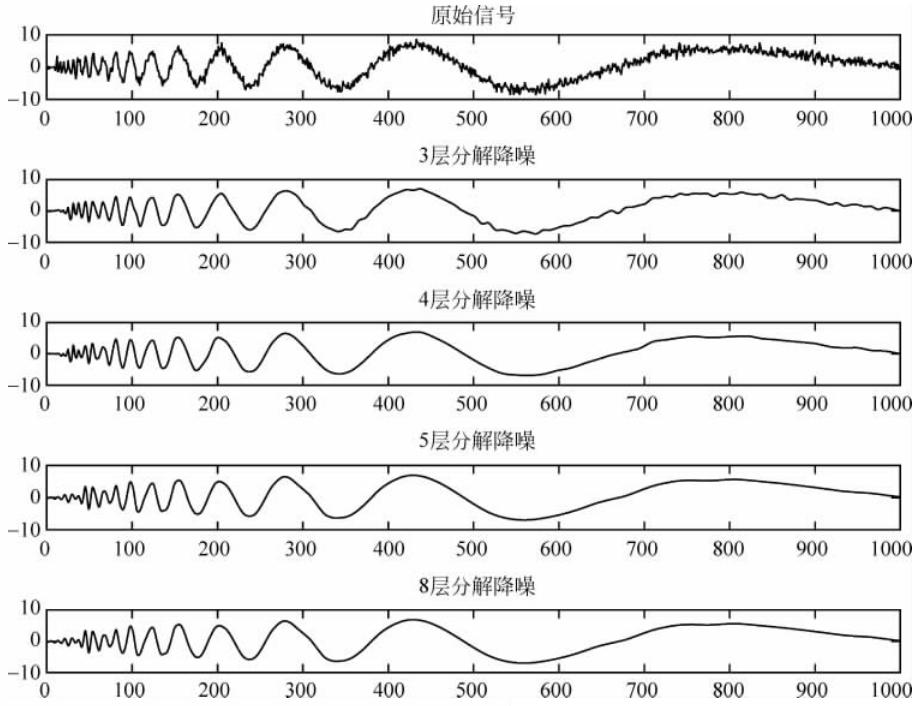

图 5-22　小波包去噪效果

小波包分解层数是去噪处理过程的一个关键参数，通过上面的程序可以确定。从得到的不同分解层次后的去噪效果可看出，随着小波包分解层次的增加，去噪效果变好，但是分解层次增加到加层后，去噪效果改善已经不明显，此时反而增加了计算量。

通过反复分析比较，确定分解层数为 4 层分解。这里按最优树进行小波包分解，得到相应的小波包分解低频和高频系数，再采用阈值法进行去噪处理。

5.3　信号压缩处理

信号压缩的动机源于对模拟信号的数字化。在离散化的过程中，大部分应用使用统一的采样率，而像一般数字电话的采样率都是统一的，所以在很多时间使用的采样率高于需要的精度，造成了信息的冗余。

与信号去噪相似，压缩领域中，由于小波的特殊优点，应用其进行压缩也受到了人们的重视，并获得了非常好的效果。

5.3.1　小波压缩概述

随着数据压缩理论的不断发展和日益成熟，一些研究人员逐渐开始对声音、文字和图像等信号的压缩技术进行研究，先后经历了经典压缩方法和现代压缩方法两个阶段。

经典压缩方法遵循香农信息论的基本理论，可分为无损压缩和有损压缩两类压缩

方法。

(1) 无损压缩方法(如 Huffman 编码、算术编码、游程编码等)设法改变信源的概率分布,使信号的概率分布尽可能地非均匀,再用最佳编码方法重新对每个信号分配码字,使平均码长逼近信源熵。该类方法的压缩效率都以其信息熵为上限。

(2) 有损压缩方法(如预测编码、变换编码、混合编码、矢量化等)设法去除信源之间的相关性,使之成为或差不多成为不相关的信源,该类压缩方法也受信息熵的约束。

随着小波变换、分形几何理论、数学形态学等数学理论和相关学科(如模式识别、人工智能、神经网络、感知生理心理学等)的深入发展,相继出现了新颖高效的现代压缩方法。该类方法包括子带编码、小波变换编码、神经网络编码、分形编码、模型基编码等。随着信号压缩技术在相关领域中的广泛应用,信号压缩的国际标准相继被制定,如在静止信号压缩方面已经制定了 CCIT T.81、ISO 10918(JPEG)及 JPEG2000 等标准。

从信号分析角度来讲,傅里叶变换是信号和数字信号处理的理论基础,小波分析将信号和数字信号处理带入一个崭新的领域。

多尺度几何分析是继小波分析后的新一代信号分析工具,它具有多分辨、局部化和多方向性等优良特性,更适合于处理高维信号。

为了缓解人们在处理信息的巨量需求时所造成的信号采样、传输和存储的巨大压力,近年来许多小波分析方面的专家又提出了压缩感知理论。与 Nyquist 采样定理不同,它指出,只要信号在某个变换域是稀疏的,那么就可以用一个与变换基不相关的观测矩阵将变换所得高维信号投影到一个低维空间上,然后通过求解一个优化问题就可从这些少量的投影中以高概率重构出原信号。可证明这样的投影包含了重构信号的足够信息。

在压缩感知理论中,信号与信号的采样和压缩同时以低速率进行,使传感器的采样和计算成本大大降低,而信号的恢复过程是一个优化计算的过程。因此,该理论指出了将模拟信号直接采样压缩为数字形式的有效途径,具有直接信息采样特性。

由于从理论上讲任何信号都具有可压缩性,只要能找到其相应的稀疏表示空间,就可有效地进行压缩采样,这一理论必将给信号采样方法带来一次新的革命。压缩感知理论的引人之处还在于它对应用科学和工程的许多领域具有重要的影响和实践意义,如统计学、信息论、编码等。

目前,研究此理论的机构越来越多。美国 Rice 大学成功地研制出了单相素 CS 相机,它可利用单一的信号光子检测器采样得到比信号像素点数少得多的点,恢复得到信号,并且具有对信号波长自适应的能力,这种自适应能力是传统的 CCD 和 CMOS 成像器件所不具备的。

5.3.2 信号压缩步骤

MATLAB 中,信号的压缩分为如下 3 个步骤。

(1) 对原始信号进行小波分解,得到分解系数。

(2) 对小波域中的系数进行处理,去除信号中的冗余。在此步骤中,指导去除信号冗余的是信号的能量剩余、小波域的模平方比和零系数成分等性能指标。

① 能量剩余：

$$\text{能量保留成分}=100\%\times\frac{\text{处理过的小波系数的模平方}}{\text{原信号的模平方}}$$

对双正交小波：

$$\text{能量保留成分}=100\%\times\frac{\text{压缩信号的模平方}}{\text{原信号的模平方}}$$

② 小波域的模平方比

$$\text{小波域的模平方比}=100\%\times\frac{\text{处理过的小波系数的模平方}}{\text{原信号小波系数的模平方}}$$

③ 零系数成分

$$\text{零系数成分}=100\%\times\frac{\text{处理后的小波系数中 0 的个数}}{\text{原信号的分解系数的个数}}$$

(3) 重建分解后的系数，得到压缩后的信号。

在选择小波基进行信号压缩的过程中，需要注意的是正交小波和双正交小波的选择。正交小波除 Haar 小波外都不具有对称性，那么对其小波系数的修改可能导致信号频谱产生偏移，改变信号的一些固有性质；而双正交小波都可构成对称的、具有线性相位的滤波器，这样把信号变换到小波域后，任何对系数的处理都只改变信号的幅值，而不会改变信号的频率分布。在实际的信号压缩标准中，一般采用支集长度为 9 或 7 的双正交小波。由于它在平移通道中的信息还存在冗余，如果某些信号损坏，还可以通过其他信号恢复出来，容错性好，更适合实际。

5.3.3 小波压缩实现法

在小波变换中，信号压缩可归结为在信号的小波分解域对小波系数进行量化的过程。由于量化的过程可以看作是系数的内部扰动，所以在小波域中系数对扰动的稳定性是至关重要的。对于实现来讲，一个首要的问题就是用更少的位来存储小波系数，并尽量保证重构信号的质量。

为了实现信息的分离，减少存储或传输的代价，压缩后用以存储或传输的数据应该是小波域经过量化后的数据。一种简单的量化方法是归一化量化方法，其基本思想是把整个系数的值域分解为不同的区间，各区间所有值出现的概率之和相同，且对每个区间的值用相同的位数加以量化。

信号压缩的具体流程如图 5-23 所示。

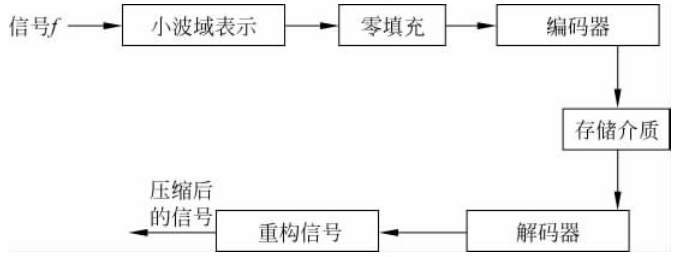

图 5-23 小波变换用于数据压缩的流程

信号 f：被压缩的原信号在小波分解域中是时频一致的。

零填充：去除采样过程中的时间冗余，对采样率为 N_0 的情况，作用如下算式：

$$Z_{N_0} = \begin{cases} c_n, & n = kn_0 \\ 0, & \text{其他} \end{cases}$$

编码器：对输入系数进行编码，去除其编码冗余，并根据需要进行量化。

存储介质：编码数据的存储容器。

解码器：将编码后的系数重新恢复到小波表示域，便于重构。

重构信号：恢复压缩后的信号。

5.3.4 信号压缩实现

尽管同消噪过程有很大区别，但是压缩算法也是通过在信号的小波域表示作用阈值来实现的，因此其 MATLAB 函数大部分与消噪过程相同。下面通过一个实例来演示信号的压缩。

【例 5-21】 利用小波分析，对给定的信号进行压缩处理。

```
>> clear all;
load leleccum;                                  % 载入信号
s = leleccum(2000:2800);                        % 采样点
subplot(411);plot(s);
title('原始信号');
% 用 db4 小波对原信号进行 3 层分解
[c,l] = wavedec(s,3,'db4');
% 指定不同阈值对系数进行处理
sorh = 'h';
keepapp = 1;
thr = 33;
[s1,CXC,LXC,PERF01,PERFL21] = wdencmp('gbl',c,l,'db4',3,thr,sorh,keepapp);
thr = 23;
[s2,CXC,LXC,PERF02,PERFL23] = wdencmp('gbl',c,l,'db4',3,thr,sorh,keepapp);
thr = 13;
[s3,CXC,LXC,PERF03,PERFL13] = wdencmp('gbl',c,l,'db4',3,thr,sorh,keepapp);
subplot(412);plot(s1);
grid on;
title('不同阈值对信号进行压缩处理');
xlabel(['能量剩余,num2str(PERFL21),零系数成分',num2str(PERF01)]);
subplot(413);plot(s2);
xlabel(['能量剩余,num2str(PERFL23),零系数成分',num2str(PERF02)]);
subplot(414);plot(s3);
xlabel(['能量剩余,num2str(PERFL13),零系数成分',num2str(PERF03)]);
```

运行程序，效果如图 5-24 所示。

从图 5-24 可看出，设置不同的阈值，可以很灵活地根据需要确定应用的压缩性能。

从该实例还可看出小波变换用于数据压缩的基本思想，就是将信号通过小波变换把低采样率的近似系数和少量需要保留的近似系数组合在一起，而信号其他的细节系数（不是信号的主要性质）通过阈值滤掉，完成压缩。

在实例中，阈值的作用方式是全局阈值，可能对信号的局部性质刻画得不够精细，读者可以自行设计分层阈值的例子，并就压缩性能与全局阈值方法进行比较。

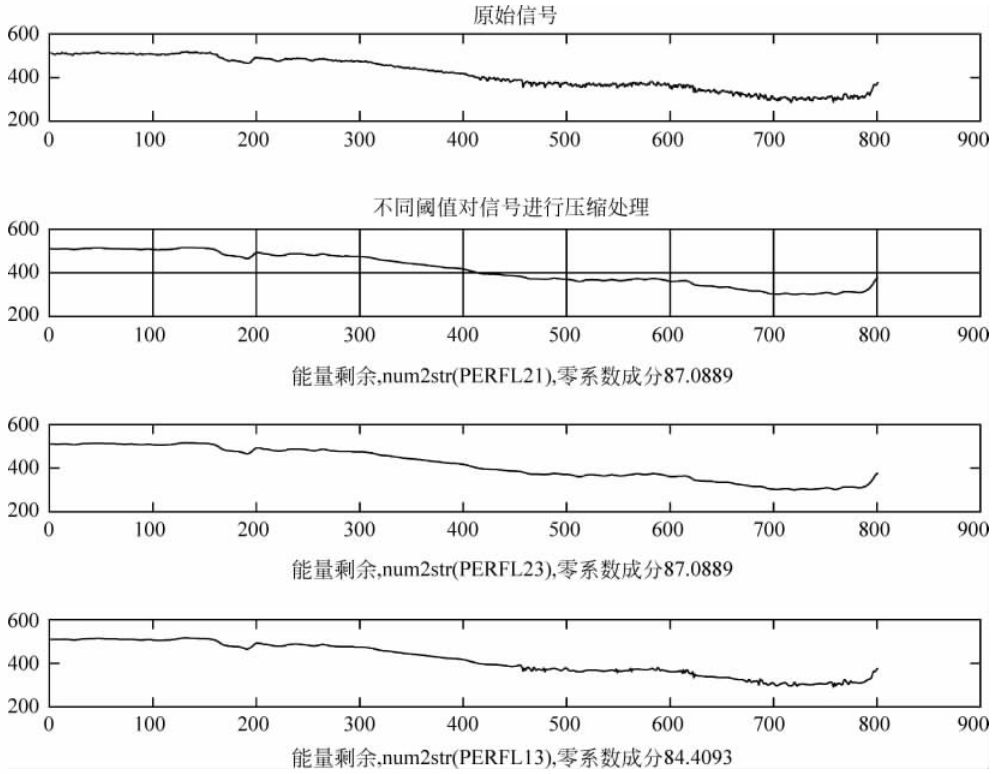

图 5-24　信号压缩

5.4　小波在信号处理中的应用

小波变换作为信号处理的一种手段,逐渐被越来越多领域的理论工作者和工程技术人员所重视和应用,并取得了显著的成果。

5.4.1　小波分解在信号中的应用

本小节主要介绍如何利用小波来分析信号。文中仅对某类小波分析结果做出一定的分析。有兴趣的读者可以观察各类信号使用小波分析方法所得到的结果,以加深小波分析用于信号处理的基本步骤和过程。

含噪的三角波与正弦波的组合,其表达式为

$$s(t)=\begin{cases}\dfrac{t-1}{500}+\sin(0.3t)+b(t) & 1\leqslant t\leqslant 500\\ \dfrac{1000-t}{500}+\sin(0.3t)+b(t) & 501\leqslant t\leqslant 1000\end{cases}$$

1. 正弦信号混合噪声

下面通过使用小波分析由正弦信号(正弦信号的周期约为 200)加白噪声组成的信

息，说明小波分析如何分离这两种信号。

【例 5-22】 正弦信号混合噪声演示。

```
>> clear all;
load noissin;
x = noissin;figure;
subplot(6,1,1);plot(x);
ylabel('x');
 %使用 db6 小波对信号进行 6 层分解
[C,L] = wavedec(x,5,'db6');
for i = 1:5
     %对分解的第 6 层到第 1 层的低频系数进行重构
    a = wrcoef('a',C,L,'db6',6 - i);
    subplot(6,1,i + 1);
    plot(a);
    ylabel(['a',num2str(6 - i)]);
end
figure;
subplot(6,1,1);plot(x);
ylabel('x');
for i = 1:5
     %对分解的第 5 层到第 1 层的高频系数进行重构
    d = wrcoef('d',C,L,'db6',6 - i);
    subplot(6,1,i + 1);
    plot(d);
    ylabel(['d',num2str(6 - i)]);
end
```

运行程序，效果如图 5-25 及图 5-26 所示。

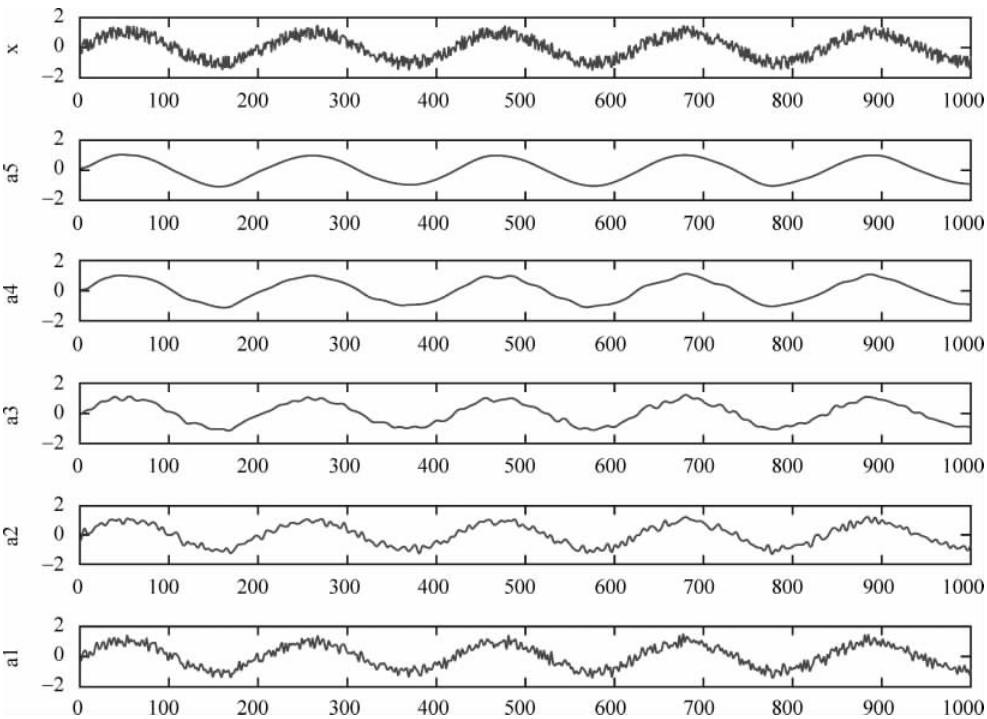

图 5-25 小波分解的低频系数

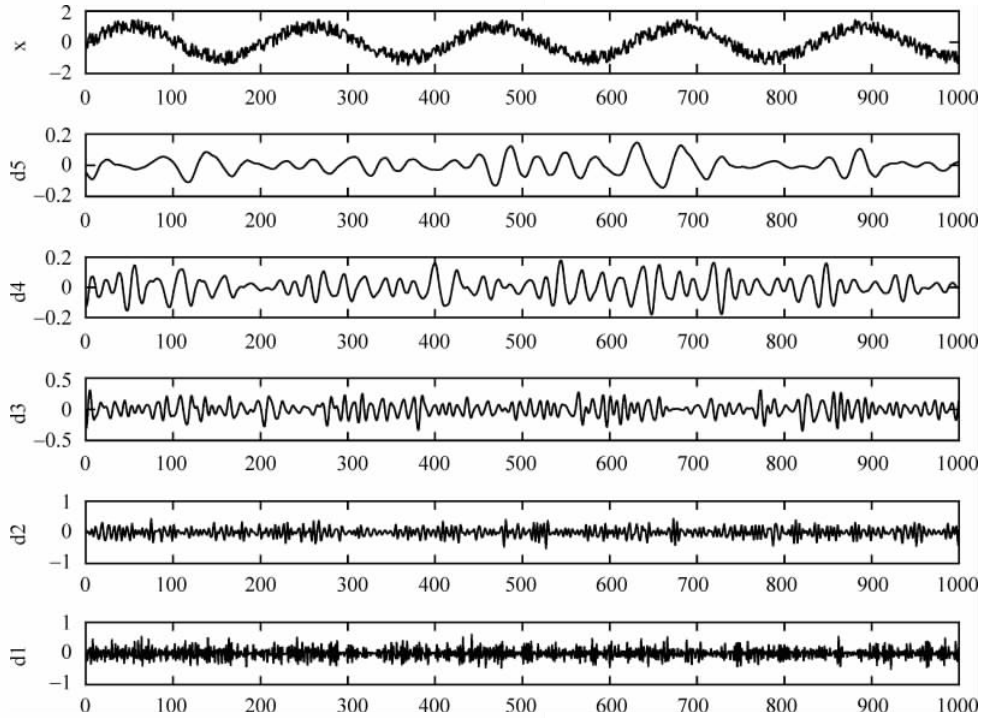

图 5-26　小波分解的高频系数

由图 5-26 可知，小波分解的细节信号是由白噪声分解得到的，而正弦信号可在图 5-25 的近似信号 a5 中得到，因为在这一层噪声对正弦信号的影响已经可忽略了。

2. 正弦信号混合三角波

【例 5-23】 通过使用小波分析一个由正弦信号（正弦信号的周期约为 20）加三角波组成的信号，说明小波分析如何分离这两种信号。

```
>> clear all;
 % 生成正弦信号
N = 1000;
t = 1:N;
sig1 = sin(0.3 * t);
 % 生成三角波信号
sig2(1:500) = ((1:500) - 1)/500;
sig2(501:N) = (1000 - (501:1000))/500;
figure(1);
subplot(2,1,1);plot(t,sig1);
xlabel('样本序号 n');
ylabel('幅值 A');
subplot(2,1,2);plot(t,sig2);
xlabel('样本序号 n');
ylabel('幅值 A');
 % 叠加信号
x = sig1 + sig2 + randn(1,N);
figure(2);plot(t,x);
```

```
xlabel('样本序号 n');
ylabel('幅值 A');
%一维小波分解
[c,l] = wavedec(x,7,'db5');
%重构第 1-7 层逼近系数.
a7 = wrcoef('a',c,l,'db5',7);
a6 = wrcoef('a',c,l,'db5',6);
a5 = wrcoef('a',c,l,'db5',5);
a4 = wrcoef('a',c,l,'db5',4);
a3 = wrcoef('a',c,l,'db5',3);
a2 = wrcoef('a',c,l,'db5',2);
a1 = wrcoef('a',c,l,'db5',1);
%显示逼近系数
figure(3)
subplot(7,1,1);plot(a7);
ylabel('a7');
subplot(7,1,2);plot(a6);
ylabel('a6');
subplot(7,1,3);plot(a5);
ylabel('a5');
subplot(7,1,4);plot(a4);
ylabel('a4');
subplot(7,1,5);plot(a3);
ylabel('a3');
subplot(7,1,6);plot(a2);
ylabel('a2');
subplot(7,1,7);plot(a1);
ylabel('a1');
xlabel('样本序号 n');
%重构第 1-7 层细节系数
d7 = wrcoef('d',c,l,'db5',7);
d6 = wrcoef('d',c,l,'db5',6);
d5 = wrcoef('d',c,l,'db5',5);
d4 = wrcoef('d',c,l,'db5',4);
d3 = wrcoef('d',c,l,'db5',3);
d2 = wrcoef('d',c,l,'db5',2);
d1 = wrcoef('d',c,l,'db5',1);
%显示细节系数
figure(4)
subplot(7,1,1);plot(d7);
ylabel('d7');
subplot(7,1,2);plot(d6);
ylabel('d6');
subplot(7,1,3);plot(d5);
ylabel('d5');
subplot(7,1,4);plot(d4);
ylabel('d4');
subplot(7,1,5);plot(d3);
ylabel('d3');
subplot(7,1,6);plot(d2);
ylabel('d2');
```

```
subplot(7,1,7);plot(d1);
ylabel('d1');xlabel('样本序号 n');
```

运行程序,效果如图 5-27～图 5-30 所示。

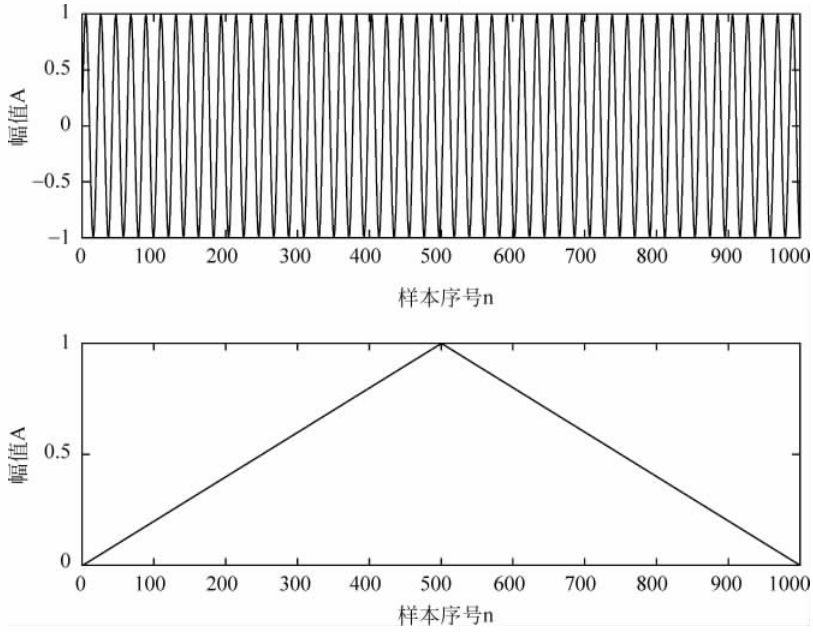

图 5-27　正弦波与三角波效果图

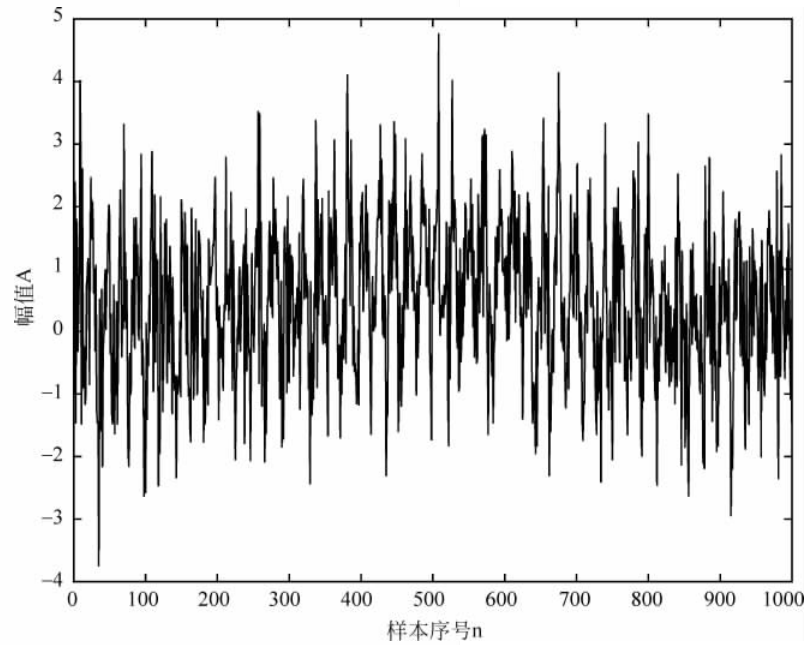

图 5-28　含噪的三角波与正弦波混合信号波形

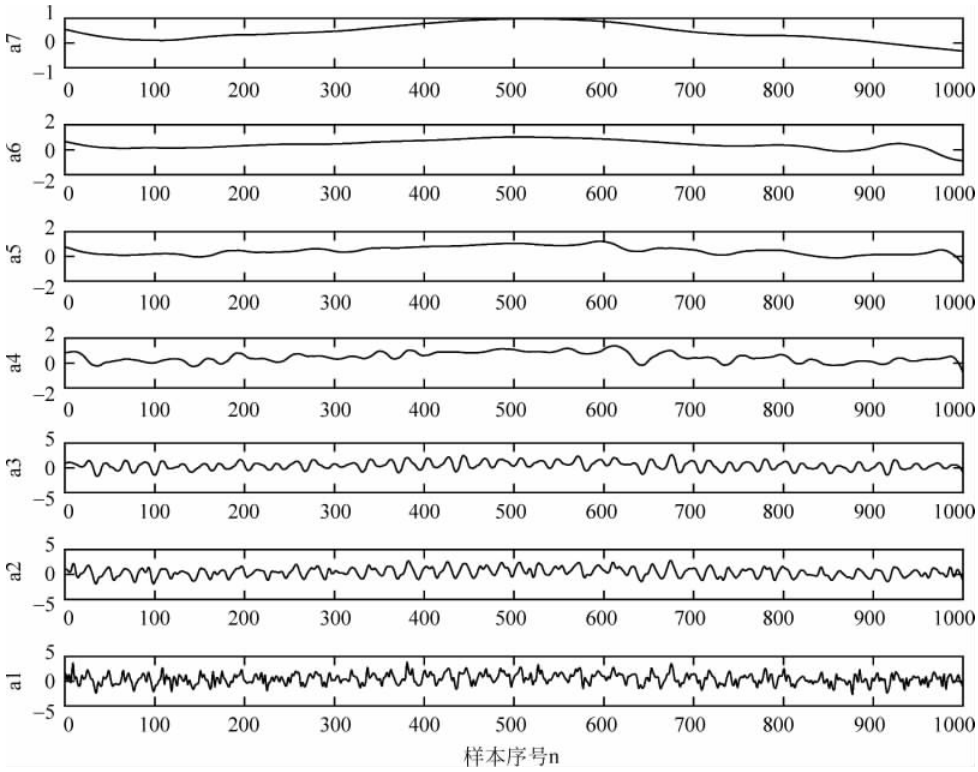

图 5-29　小波分解的低频系数

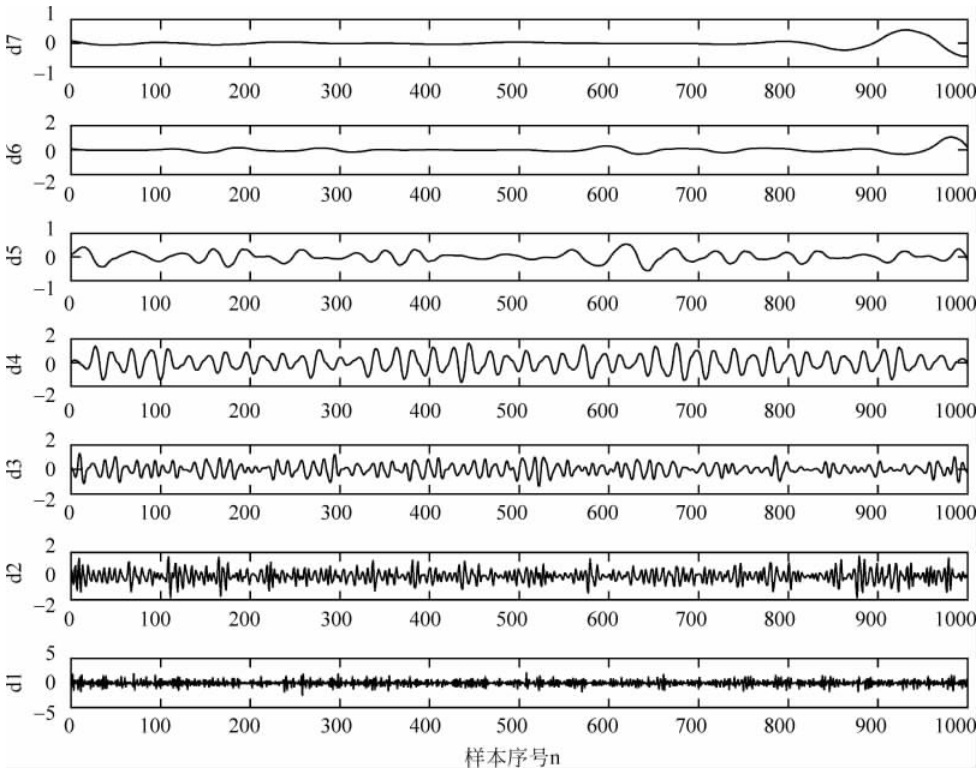

图 5-30　小波分解的高频系数

3. 含噪的多项式信号

含噪的多项式信号的表达式为

$$s(t) = t^2 - t + 1 + b_1(t)$$

【例 5-24】 分别应用 db2 和 db3 小波对上述多项式信号进行 4 层分解。

```
>> clear all;
% 生成含噪的多项式信号
N = 800;
t = 1:N;
sig = t.^2 - t + 1;
x = sig + randn(1,N);
% 一维小波分解
[c,l] = wavedec(x,4,'db2');
% [c,l] = wavedec(x,4,'db3');
% 重构第 1～4 层逼近系数
a4 = wrcoef('a',c,l,'db3',4);
a3 = wrcoef('a',c,l,'db3',3);
a2 = wrcoef('a',c,l,'db3',2);
a1 = wrcoef('a',c,l,'db3',1);
% 显示逼近系数
figure;
subplot(411);plot(a4);ylabel('a4');
subplot(412);plot(a3);ylabel('a3');
subplot(413);plot(a2);ylabel('a2');
subplot(414);plot(a1);ylabel('a1');
xlabel('样本序列 n');
% 重构第 1～4 层细节系数
d4 = wrcoef('d',c,l,'db3',4);
d3 = wrcoef('d',c,l,'db3',3);
d2 = wrcoef('d',c,l,'db3',2);
d1 = wrcoef('d',c,l,'db3',1);
figure;
subplot(411);plot(d4);ylabel('d4');
axis([0,N,-100,100]);
subplot(412);plot(d3);ylabel('d3');
axis([0,N,-10,30]);
subplot(413);plot(d2);ylabel('d2');
axis([0,N,-5,5]);
subplot(414);plot(d4);ylabel('d1');
xlabel('样本序列 n');
```

运行程序，效果如图 5-31 及图 5-32 所示。

利用 db2 小波分解后的逼近信号如图 5-31 所示，细节信号如图 5-32 所示。可以看出，这种情况下随分解层级的增加，其正则性增加，从而抑制了该多项式信号的零阶和一阶部分，而仅对该信号的二阶部分及噪声进行了分解。因此，在图 5-32 中，除了细节信号 d1 中包含该含噪信号的不规则性，其余各层细节中的信号周期性（规则性）随着层级的增加而增大。

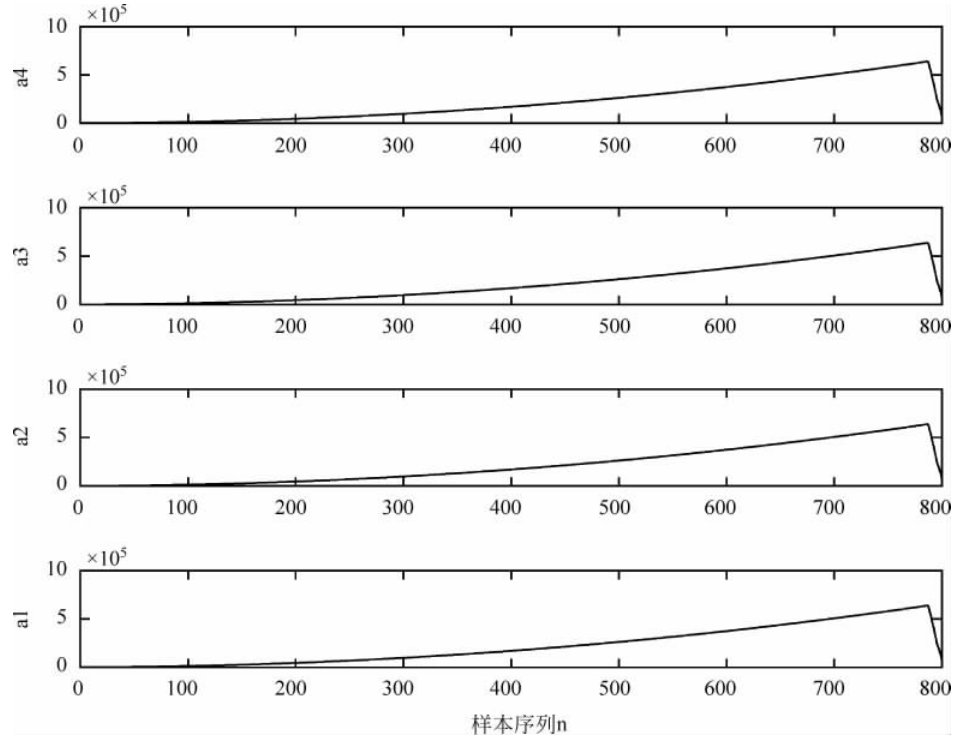

图 5-31　小波分解后各层逼近信号(db2)

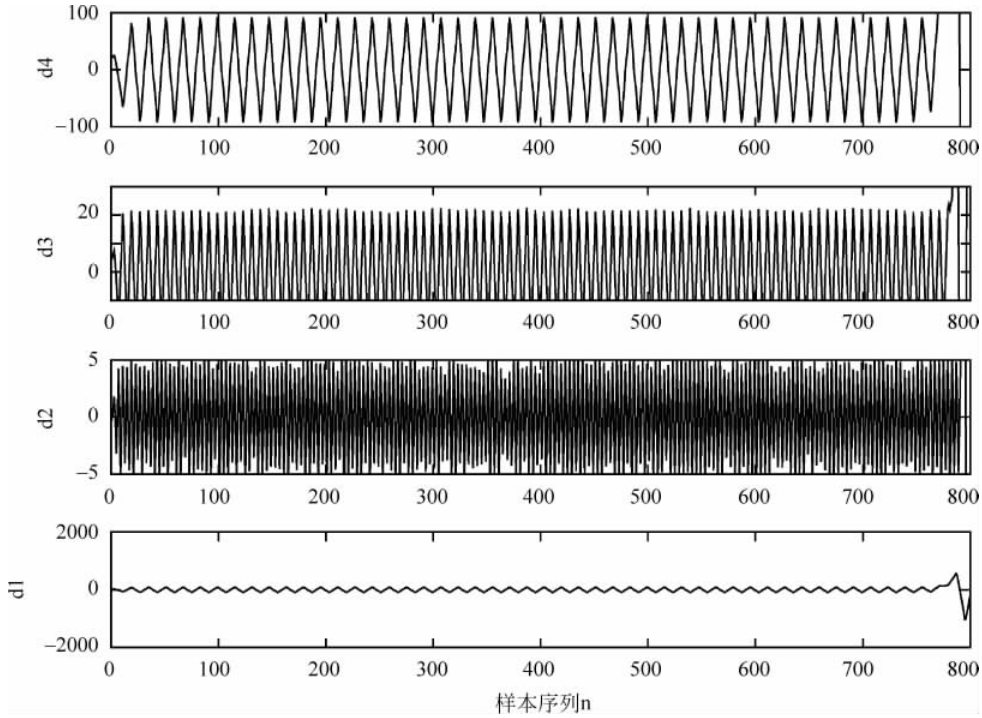

图 5-32　小波分解后各层细节信号(db2)

利用 db3 小波分解后的逼近信号如图 5-33 所示，细节信号如图 5-34 所示。可以看出，由于 db3 小波的正则性较差，所以它抑制了该信号的多项式部分，而析出了它的噪声部分。因此，利用小波分析可以较好地对该类信号进行抑制。

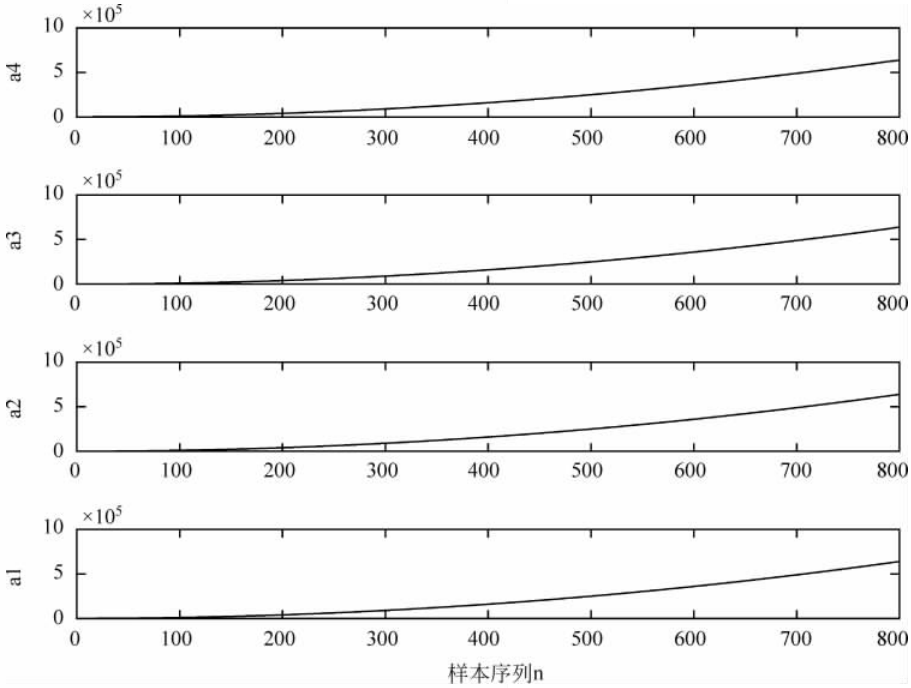

图 5-33　小波分解后各层逼近信号(db3)

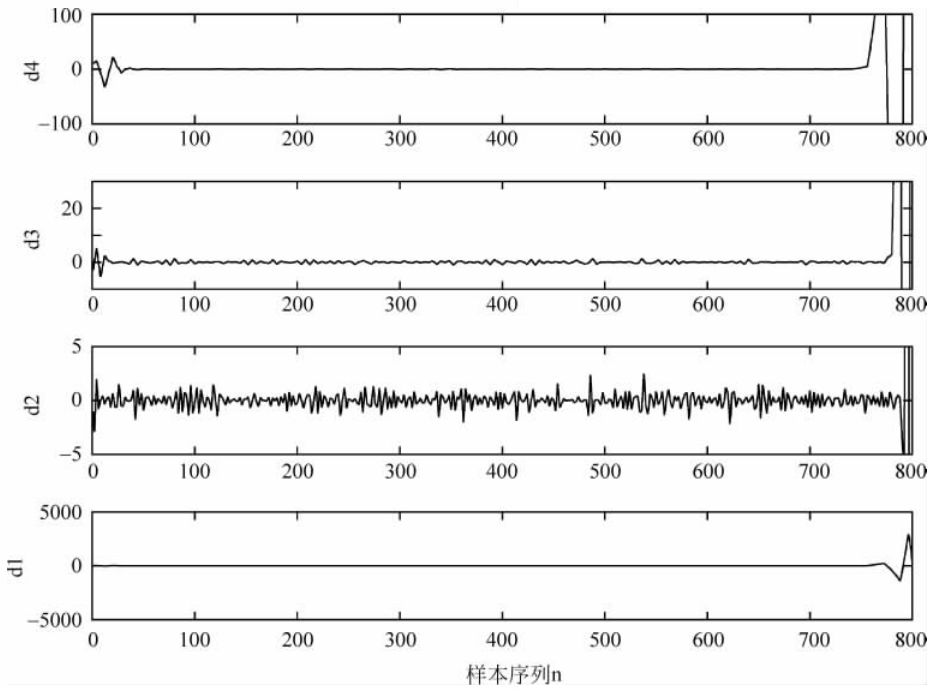

图 5-34　小波分解后各层细节信号(db3)

5.4.2 小波信号去噪应用

下面通过一个实例来演示怎样利用小波实现信号的去噪应用。

【例 5-25】 利用小波阈值方法进行去噪处理。

```
>> clear all;
% 分解的有用信号小波高频系数基本趋于零
% 对于噪声信号高频系数很大,便于阈值去噪处理
[l,h] = wfilters('db9','d');
low_construct = l;
L_fre = 12;                                    % 滤波器长度
low_decompose = low_construct(end: - 1:1);     % 确定 h0( - n),低通分解滤波器
for i_high = 1:L_fre;                          % 确定 h1(n) = ( - 1)^n,高通重构滤波器
    if(mod(i_high,2) == 0);
        coefficient = - 1;
    else
        coefficient = 1;
    end
    high_construct(1,i_high) = low_decompose(1,i_high) * coefficient;
end
high_decompose = high_construct(end: - 1:1);   % 高通分解滤波器 h1( - n)
L_signal = 100;                                % 信号长度
n = 1:L_signal;                                % 原始信号赋值
f = 10;
t = 0.001;
y = sin(2 * pi * 50 * n * t). * exp( - 10 * n * t);
% 信号 + 噪声
noise = [zeros(1,L_signal/2),2 * (rand(1,20) - 0.4),zeros(1,30)];
y_noise = y + noise;
figure;                                        % 得到原始信号与含噪信号,如图 5 - 35 所示
subplot(211);plot(y);title('原信号');
subplot(212);plot(y_noise);title('含噪的信号');
check1 = sum(high_decompose);                  % 性质检验,h0(n)
check2 = sum(low_decompose);
check3 = norm(high_decompose);
check4 = norm(low_decompose);
l_fre = conv(y_noise,low_decompose);           % 卷积
l_fre_down = dyaddown(l_fre);                  % 抽取,得低频细节
h_fre = conv(y_noise,high_decompose);
h_fre_down = dyaddown(h_fre);                  % 信号高频细节
figure;                                        % 得到小波分解系数效果图,如图 5 - 36 所示
subplot(211);plot(l_fre_down);title('低频系数');
subplot(212);plot(h_fre_down);title('高频系数');
% 去噪处理
for i_decrease = 31:44;
    if abs(h_fre_down(1,i_decrease))> = 0.000001
        h_fre_down(1,i_decrease) = (10 ^ - 7);
    end
end
l_fre_pull = dyadup(l_fre_down);               % 零差值
h_fre_pull = dyadup(h_fre_down);
```

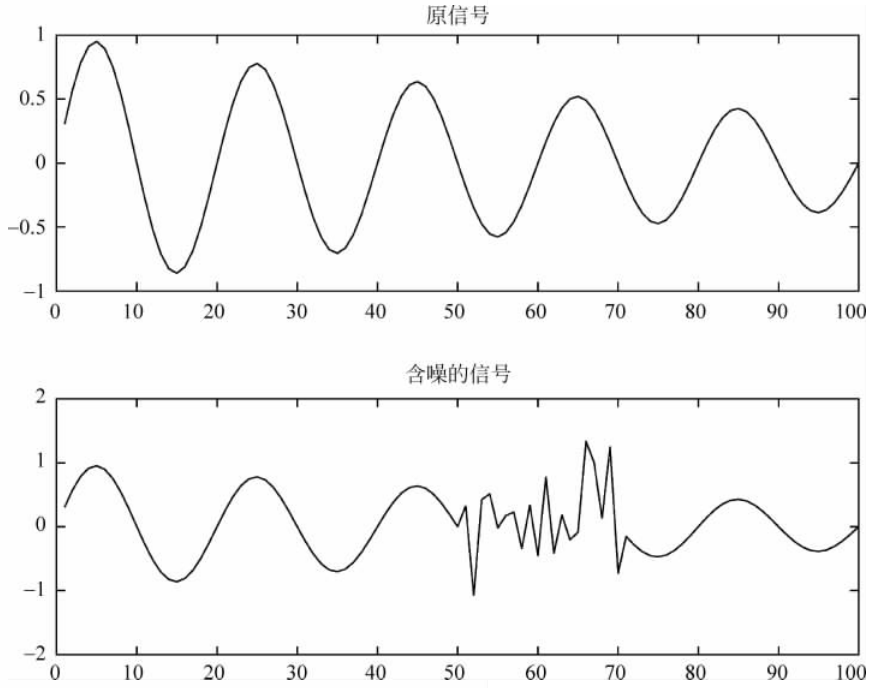

图 5-35　含噪信号

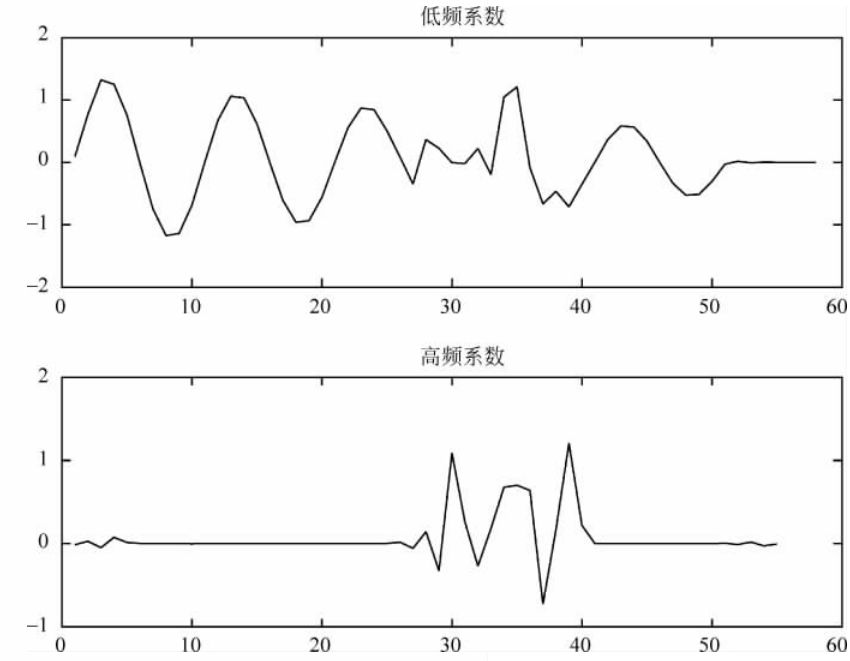

图 5-36　小波分解系数

```
l_fre_denoise = conv(low_construct,l_fre_pull);
h_fre_denoise = conv(high_construct,h_fre_pull);
l_fre_keep = wkeep(l_fre_denoise,L_signal);
%取结果的中心部分,消除卷积影响
h_fre_keep = wkeep(h_fre_denoise,L_signal);
sig_denoise = l_fre_keep + h_fre_keep;          %去噪后信号重构
%平滑处理
for j = 1:2
    for i = 60:70;
        sig_denoise(i) = sig_denoise(i - 2) + sig_denoise(i + 2)/2;
    end
end
compare = sig_denoise - y;                      %与原信号比较
figure;                                         %原信号与去噪后信号的比较,效果如图 5 - 37 所示
subplot(311);plot(y);ylabel('y');               %原信号
subplot(312);plot(sig_denoise);ylabel('sig\_denoise');   %去噪后信号
axis tight;
subplot(313);plot(compare);ylabel('compare');            %原信号与去噪后的信号比较
axis tight;
```

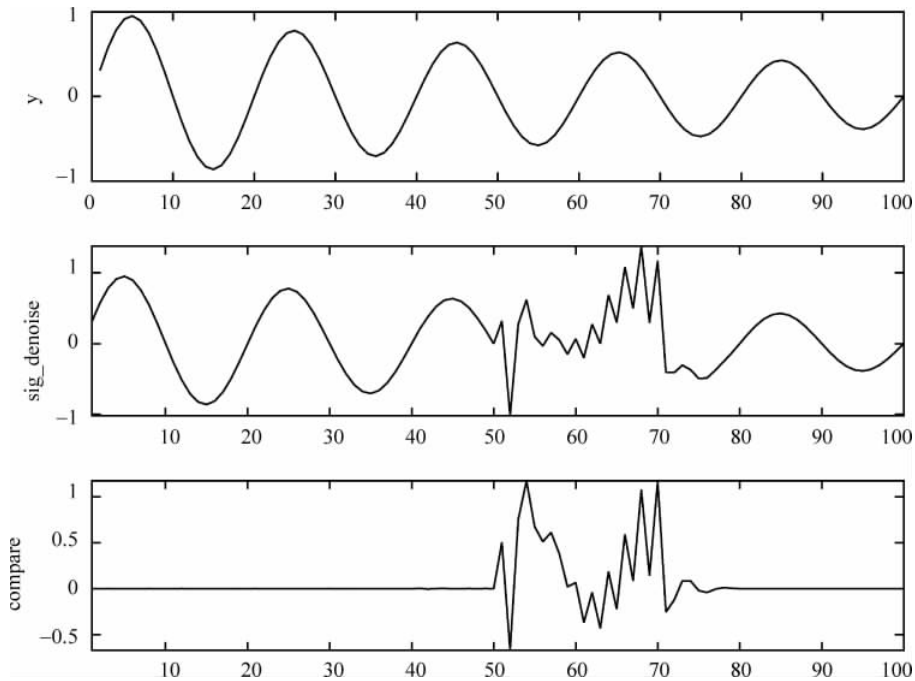

图 5-37　原信号与消噪后信号的比较