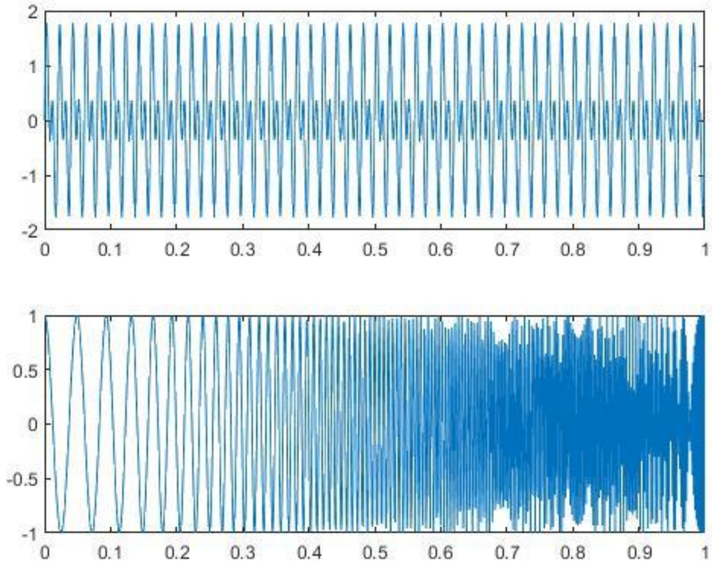
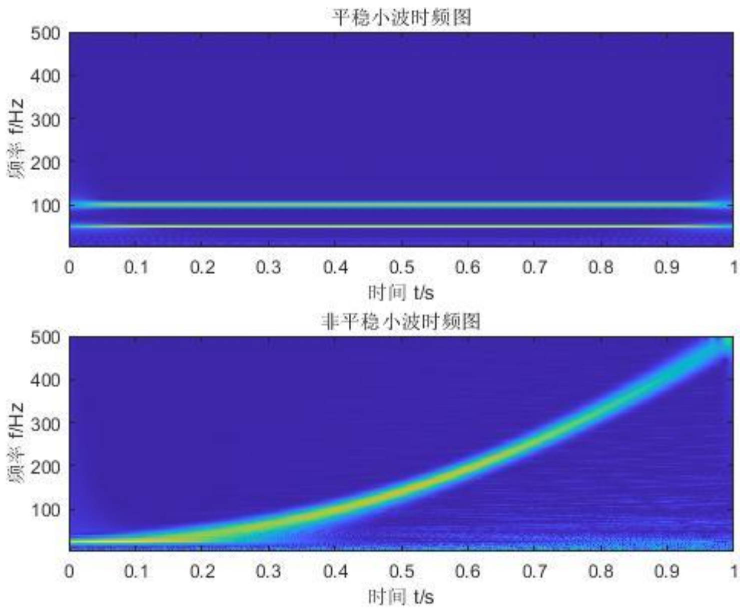
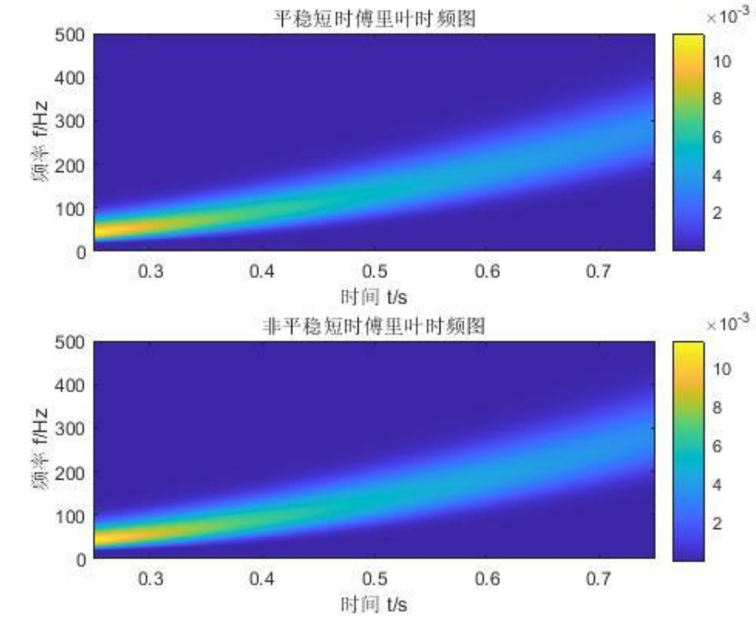
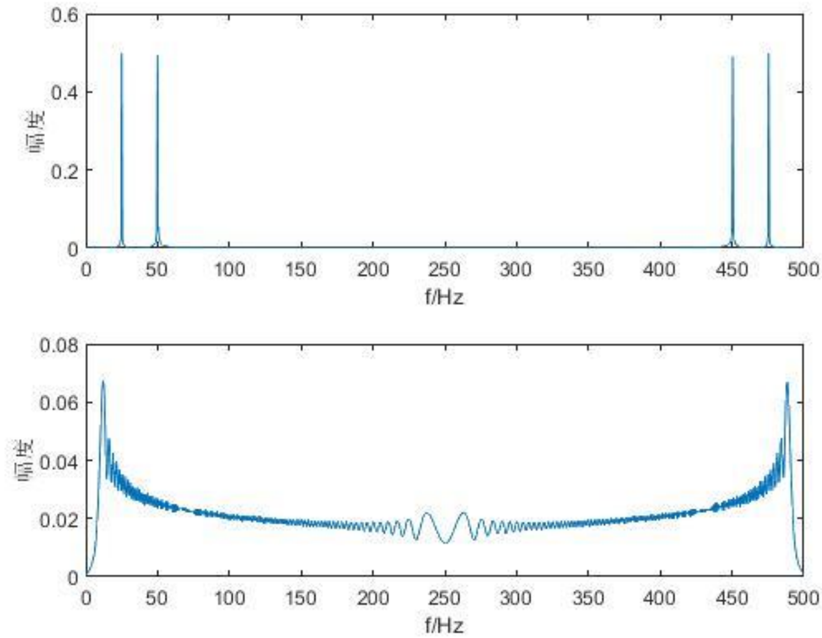
1、 掌握一维和二维小波变换的原理。编程实现用小波变换绘制由两个正弦波（50HZ 和 100Hz）构成的平稳信号和非平稳信号的时频图，并与两种信号的傅里叶变换、短时傅里叶变换进行比较分析，得出结论。

（1） **结果图**









（2）　结论

由结果图可知高频时小波变换效果好，因为小波在高频处分辨率可以自动调整，分辨率高，小波变换有窗口自适应的特点，即高频信号分辨率高（但是频率分辨率差），低频信号频率分辨率高（但是时间分辨率差）。

和傅里叶变换比，小波变换短时傅里叶变换都有着相同的优点，可以同时在时域和频域观察信号。所以小波变换在非定常信号的分析中有很大的作用。

傅里叶变换在频谱图中，可以看到频谱的分布情况，但要知道时域上的信号情况，

并不明显。小波变换和短时傅里叶变换是对信号处理的较优方法，具体用哪一个，需要分析低频和高频不同的情况。

（3） 代码

```
fs=1000;
f1=50;
f2=100;
t=0:1/fs:1;
s = sin(2*pi*f1*t)+sin(2*pi*f2*t);
figure(1)
subplot(211);
plot(t, s)
wavename='cmor3-3';
totalscal=256;
Fc=centfrq(wavename);
c=2*Fc*totalscal;
scals=c./(1:totalscal);
f=scal2frq(scals,wavename,1/fs);
coefs=cwt(s,scals,wavename);
figure(2);
subplot(211);
imagesc(t,f,abs(coefs));
set(gca,'YDir','normal')
xlabel('时间 t/s');
ylabel('频率 f/Hz');
title('平稳小波时频图');
N=size(t,2);
wlen=500;
hop=1;
x=wkeep1(x,N+1*wlen);
h=hamming(wlen);
[B, F, T, P] = spectrogram(x,h,wlen-hop,N,fs);   %
figure(4);
subplot(211);
imagesc(T,F,P);
set(gca,'YDir','normal')
colorbar;
xlabel('时间 t/s');
ylabel('频率 f/Hz');
title('平稳短时傅里叶时频图');
toc;
fs=1000;
f1=50;
f2=100;
```

```
t=0:1/fs:1;
s = sin(2*pi*f1*t)+sin(2*pi*f2*t);
s = chirp(t,20,1,500,'q');
figure(1);
subplot(212);
plot(t, s)
wavename='cmor3-3';
totalscal=256;
Fc=centfrq(wavename);
c=2*Fc*totalscal;
scals=c./(1:totalscal);
f=scal2frq(scals,wavename,1/fs);
coefs=cwt(s,scals,wavename);
figure(2);
subplot(212);
imagesc(t,f,abs(coefs));
set(gca,'YDir','normal')
xlabel('时间 t/s');
ylabel('频率 f/Hz');
title('非平稳小波时频图');

N=size(t,2);
wlen=500;
hop=1;
x=wkeep1(x,N+1*wlen);
h=hamming(wlen);
[B, F, T, P] = spectrogram(x,h,wlen-hop,N,fs);   %
figure(4);
subplot(212);
imagesc(T,F,P);
set(gca,'YDir','normal')
colorbar;
xlabel('时间 t/s');
ylabel('频率 f/Hz');
title('非平稳短时傅里叶时频图');
toc;

fs=1000;
f1=50;
f2=100;
t=0:1/fs:1;
T=2;
N=T*fs
y=sin(2*pi*f1*t)+sin(2*pi*f2*t);
```

```
figure
subplot(211);
[f,y]=get_fft(y,fs,N);
plot(f,y);
xlabel('f/Hz');
ylabel('幅度');

fs=1000;
f1=50;
f2=100;
t=0:1/fs:1;
T=2;
N=T*fs;
y=sin(2*pi*f1*t)+sin(2*pi*f2*t);
y = chirp(t,20,1,500,'q');

subplot(212);
[f,y]=get_fft(y,fs,N);
plot(f,y);
xlabel('f/Hz');
ylabel('幅度');
function [f, spectrum ] = get_fft(s,Fs,L)
y=fft(s);
p2=abs(y/L);
p1=p2(1:L/2+1);
p1(2:end-1)=2*p1(2:end-1);
f = Fs*(0:(L/2))/L;
spectrum=p1;
end
```

2、 编程实现一维信号的小波变换去噪算法。（load leleccum 或自己生成含噪信号）。

（1） 步骤

1）加载噪声信号或者生成原始信号加入噪声

2）选择合适的小波基系数

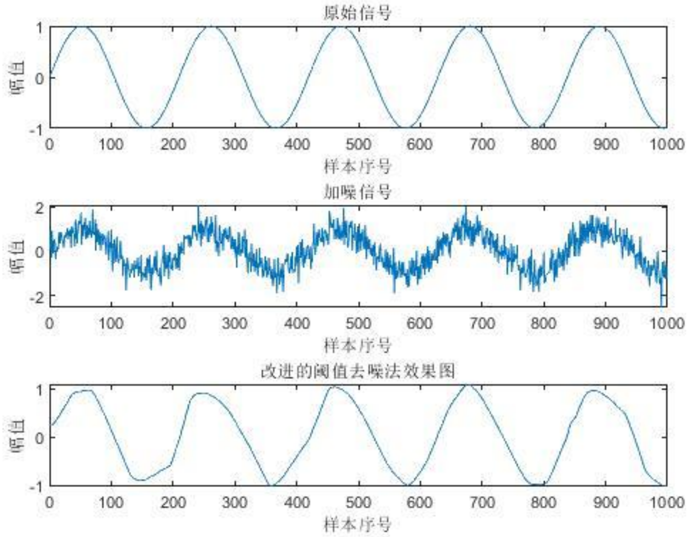
3）对信号进行指定层次的小波分解

4）对各分解层进行处理，重构

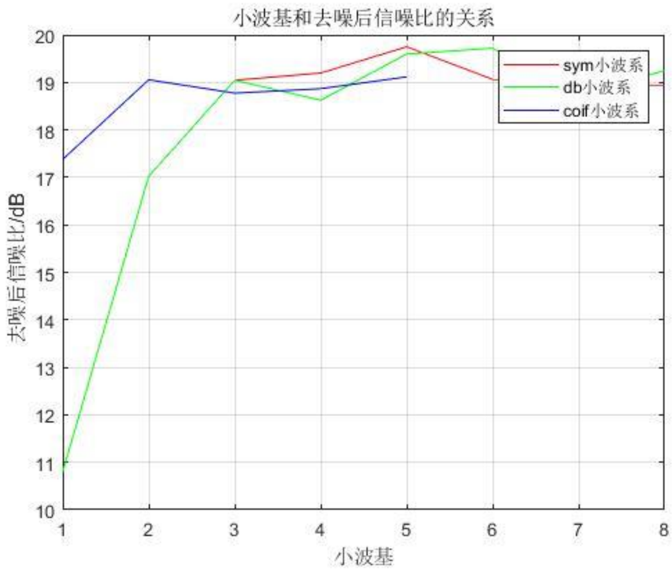
5）探讨阈值和阈值函数对重构效果的影响

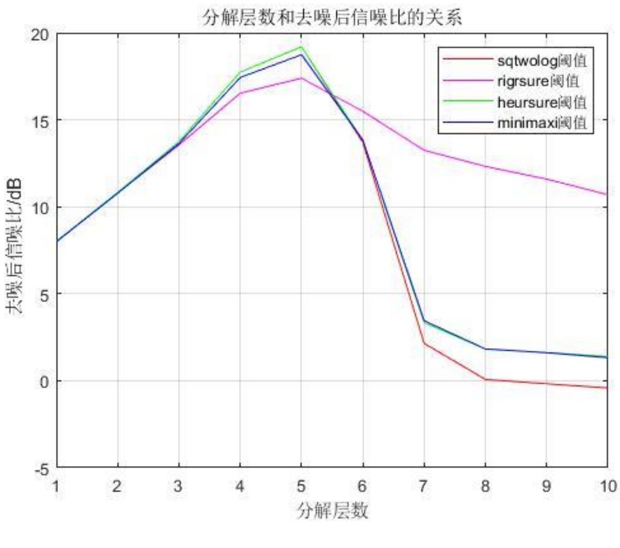
6）生成信号去噪对比图

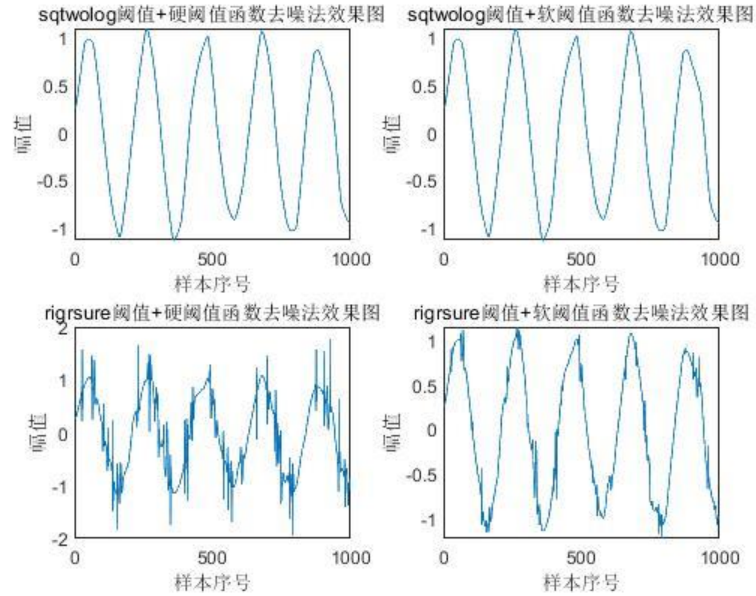
（2） 结果图


原始信号
幅值
样本序号
加噪信号
幅值
样本序号
改进的阈值去噪法效果图
幅值
样本序号

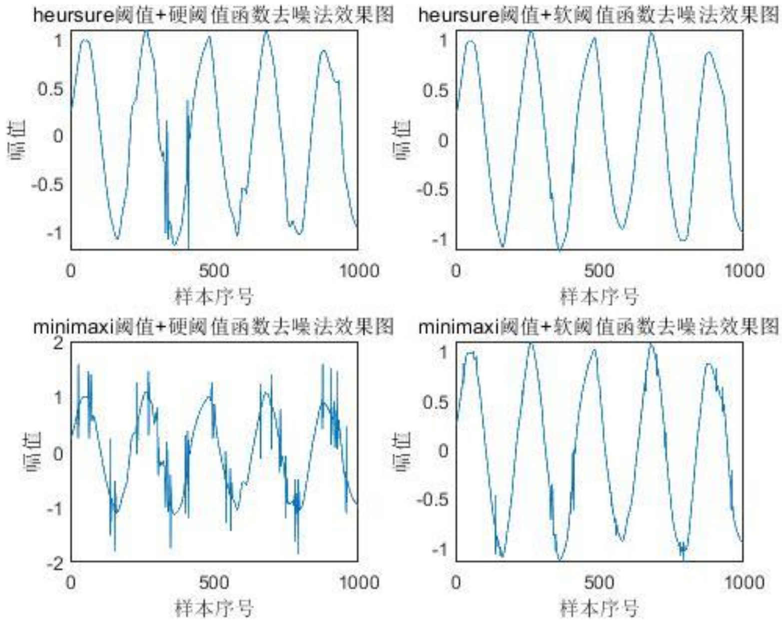
信号去噪对比图


小波基和去噪后信噪比的关系
sym小波系
db小波系
coif小波系
去噪后信噪比/dB
小波基


分解层数和去噪后信噪比的关系
sqtwolog阈值
rigrsure阈值
heursure阈值
minimaxi阈值
去噪后信噪比/dB
分解层数



阈值和阈值函数对重构效果的影响（1）



阈值和阈值函数对重构效果的影响）（2）

（3） 算法

```
%%% 生成原始信号和加噪声的信号
snr=5;
N=1000;
t=1:N;
y=sin(0.03*t);
[s,noise]=Gnoisegen(y,snr);
```

```
figure(1)
subplot(311);
plot(y);
xlabel('样本序号');
ylabel('幅值');
title('原始信号');
subplot(312);
plot(s);
xlabel('样本序号');
ylabel('幅值');
title('加噪信号');

%%小波基对去噪效果的影响
sym=['sym1';'sym2';'sym3';'sym4';'sym5';'sym6';'sym7';'sym8'];
db=['db1';'db2';'db3';'db4';'db5';'db6';'db7';'db8'];
coif=['coif1';'coif2';'coif3';'coif4';'coif5'];
snrsym=levelandth1(y,s,sym,8);
snrdb=levelandth1(y,s,db,8);
snrcoif=levelandth1(y,s,coif,5);
ksym=1:8;
kdb=1:8;
kcoif=1:5;
figure(2)
plot(ksym,snrsym,'r-',kdb,snrdb,'g-',kcoif,snrcoif,'b-'),grid on;
legend('symÐ小波系','db小波系','coif小波系');
xlabel('小波基');
ylabel('去噪后信噪比/dB');
title('小波基和去噪后信噪比的关系');

%%分解层数对去噪效果的影响
thrrr1='sqtwolog';
thrrr2='rigrsure';
thrrr3='heursure';
thrrr4='minimaxi';
wavec='sym4';
[M1,snrxd1]=level(y,s,thrrr1,wavec);
[M2,snrxd2]=level(y,s,thrrr2,wavec);
[M3,snrxd3]=level(y,s,thrrr3,wavec);
[M4,snrxd4]=level(y,s,thrrr4,wavec);
fprintf('sqtwolog阈值最佳分解层数:%d \n',M1);
fprintf('rigrsure阈值最佳分解层数: %d\n',M2);
fprintf('heursure阈值最佳分解层数: %d\n',M3);
fprintf('minimaxi阈值最佳分解层数: %d\n',M4);
```

```
k=1:10;
figure(3)
plot(k,snrxd1,'r-',k,snrxd2,'m-',k,snrxd3,'g-',k,snrxd4,'b-'),grid
on;
legend('sqtwolog阈值','rigrsure阈值','heursure阈值','minimaxi阈值');
xlabel('分解层数');
ylabel('去噪后信噪比/dB');
title('分解层数和去噪后信噪比的关系');

%%改进阈值函数
wname='sym4';
lev=5;
[c,l]=wavedec(s,lev,wname);
a5=appcoef(c,l,wname,lev);
d5=detcoef(c,l,5);
d4=detcoef(c,l,4);
d3=detcoef(c,l,3);
d2=detcoef(c,l,2);
d1=detcoef(c,l,1);
a=4;
cD=[d1,d2,d3,d4,d5];
sigma=median(abs(cD))/0.6745;
thr1=(sigma*sqrt(2*(log(length(y)))))/(log(1+1));
for i=1:length(d1)
if(d1(i)>=thr1)
cD1(i)=d1(i)-thr1/(1+exp(((d1(i)-thr1).^2)/a))-thr1/(2*exp(1/a));
else if(abs(d1(i))<thr1)
        cD1(i)=0;
    else
cD1(i)=d1(i)+thr1/(1+exp(((-d1(i)-thr1).^2)/a))+thr1/(2*exp(1/a));
    end
end
end
thr2=(sigma*sqrt(2*(log(length(y)))))/(log(2+1));
for i=1:length(d2)
if(d2(i)>=thr2)
cD2(i)=d2(i)-thr2/(1+exp(((d2(i)-thr2).^2)/a))-thr2/(2*exp(1/a));
else if(abs(d2(i))<thr2)
        cD2(i)=0;
    else
cD2(i)=d2(i)+thr2/(1+exp(((-d2(i)-thr2).^2)/a))+thr2/(2*exp(1/a));
    end
end
end
```

```
thr3=(sigma*sqrt(2*(log(length(y)))))/(log(3+1));
for i=1:length(d3)
if(d3(i)>=thr3)
cD3(i)=d3(i)-thr3/(1+exp(((d3(i)-thr3).^2)/a))-thr3/(2*exp(1/a));%
else if(abs(d3(i))<thr3)
        cD3(i)=0;
    else
cD3(i)=d3(i)+thr3/(1+exp(((-d3(i)-thr3).^2)/a))+thr3/(2*exp(1/a));
    end
end
end
thr4=(sigma*sqrt(2*(log(length(y)))))/(log(4+1));
for i=1:length(d4)
if(d4(i)>=thr4)
cD4(i)=d4(i)-thr4/(1+exp(((d4(i)-thr4).^2)/a))-thr4/(2*exp(1/a));
else if(abs(d4(i))<thr4)
        cD4(i)=0;
    else
cD4(i)=d4(i)+thr4/(1+exp(((-d4(i)-thr4).^2)/a))+thr4/(2*exp(1/a));
    end
end
end
thr5=(sigma*sqrt(2*(log(length(y)))))/(log(5+1));
for i=1:length(d5)
if(d5(i)>=thr5)
cD5(i)=d5(i)-thr5/(1+exp(((d5(i)-thr5).^2)/a))-thr5/(2*exp(1/a));%
else if(abs(d5(i))<thr5)
        cD5(i)=0;
    else
cD5(i)=d5(i)+thr5/(1+exp(((-d5(i)-thr5).^2)/a))+thr5/(2*exp(1/a));
    end
end
end
cd=[a5,cD5,cD4,cD3,cD2,cD1];
c=cd;
yhs=waverec(cd,l,wname);

%%阈值和阈值函数对去噪效果的影响
xdh1=wden(s,'sqtwolog','h','mln',M1,wavec);
figure(5)
subplot(221);
plot(xdh1);
xlabel('样本序号');
ylabel('幅值');
```

```
title('sqtwolog阈值+硬阈值函数去噪效果图');
xds1=wden(s,'sqtwolog','s','mln',M1,wavec);
subplot(222);
plot(xds1);
xlabel('样本序号');
ylabel('幅值');
title('sqtwolog阈值+软阈值函数去噪效果图');
xdh2=wden(s,'rigrsure','h','mln',M2,wavec);
subplot(223);
plot(xdh2)
xlabel('样本序号');
ylabel('幅值');
title('rigrsure阈值+硬阈值函数去噪效果图');
xds2=wden(s,'rigrsure','s','mln',M2,wavec);
subplot(224);
plot(xds2);
xlabel('样本序号');
ylabel('幅值');
title('rigrsure阈值+软阈值函数去噪效果图');
xdh3=wden(s,'heursure','h','mln',M3,wavec);
figure(6)
subplot(221);
plot(xdh3);
xlabel('样本序号');
ylabel('幅值');
title('heursure阈值+硬阈值函数去噪效果图');
xds3=wden(s,'heursure','s','mln',M3,wavec);
subplot(222);
plot(xds3);
xlabel('样本序号');
ylabel('幅值');
title('heursure阈值+软阈值函数去噪效果图');
xdh4=wden(s,'minimaxi','h','mln',M4,wavec);
%figure(8)
subplot(223);
plot(xdh4);
xlabel('样本序号');
ylabel('幅值');
title('minimaxi阈值+硬阈值函数去噪效果图');
xds4=wden(s,'minimaxi','s','mln',M4,wavec);
subplot(224);
plot(xds4);
xlabel('样本序号');
ylabel('幅值');
```

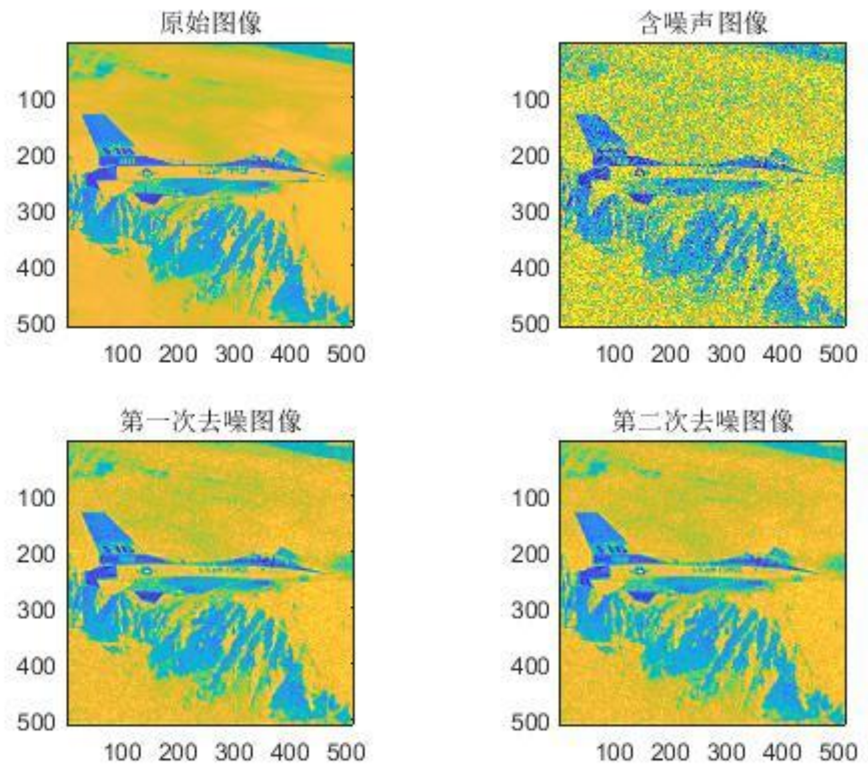
```
title('minimaxi阈值+软阈值函数去噪效果图');
figure(1)
subplot(313);
plot(yhs,'LineWidth',1);
xlabel('样本序号');
ylabel('幅值');
title('改进的阈值去噪效果图');
snrxn=snrr(y,s);
fprintf('原始噪声信号信噪比 %4.4f\n',snrxn);
snrxdh1=snrr(y,xdh1);
fprintf(' sqtwolog阈值+硬阈值函数去噪效果图 %4.4f\n',snrxdh1);
snrxdh1=snrr(y,xds1);
fprintf(' sqtwolog阈值+软阈值函数去噪效果图 %4.4f\n',snrxdh1);
snrxdh2=snrr(y,xdh2);
fprintf(' rigrsure阈值+硬阈值函数去噪效果图 %4.4f\n',snrxdh2);
snrxds2=snrr(y,xds2);
fprintf(' rigrsure阈值+软阈值函数去噪效果图 %4.4f\n',snrxds2);
snrxdh3=snrr(y,xdh3);
fprintf(' heursure阈值+硬阈值函数去噪效果图 %4.4f\n',snrxdh3);
snrxds3=snrr(y,xds3);
fprintf(' heursure阈值+软阈值函数去噪效果图 %4.4f\n',snrxds3);
snrxdh4=snrr(y,xdh4);
fprintf(' minimaxi阈值+硬阈值函数去噪效果图 %4.4f\n',snrxdh4);
snrxds4=snrr(y,xds4);
fprintf(' minimaxi阈值+软阈值函数去噪效果图 %4.4f\n',snrxds4);
snrys=snrr(y,yhs);
fprintf('改进后的信噪比 %4.4f\n',snrys);

%%子程序
function [y,noise] = Gnoisegen(x,snr)
noise=randn(size(x));
Nx=length(x);
signal_power=1/Nx*sum(x.*x);
noise_power=1/Nx*sum(noise.*noise);
noise_variance=signal_power/(10^(snr/10));
noise=sqrt(noise_variance/noise_power)*noise;
y=x+noise;
end

function snrwave=levelandth1(y,s,wave,n)
for j=1:n
    xdh=wden(s,'sqtwolog','s','mln',5,wave(j,:));
    snrwave(j)=snrr(y,xdh);
end
```

```
end

function [z,snrxd]=level(y,s,th,wave)
for k=1:10
xdh=wden(s,th,'s','mln',k,wave);
snrxd(k)=snrr(y,xdh);
end
[~,z]=max(snrxd);
end

function z=snrr(x,y)
y1=sum(x.^2);
y2=sum((y-x).^2);
z=10*log10((y1/y2));
end
```

3、编程实现小波变换在图像去噪、图像复原、图像融合、图像数字水印、图像边缘检测等中的一种应用。

选取小波变换在图像去噪方面的应用。

（1） 结果图



（2） 代码

```
I=imread('airplane.tiff');
gray=rgb2gray(I);
X=double(gray);
subplot(221);image(X);
title('原始图像');
axis square
init=2055615866;randn('seed',init)
x=X+38*randn(size(X));
subplot(222);image(x);
title('含噪声图像');
axis square;
%进行2层小波分解
[c,s]=wavedec2(x,2,'sym4');
%提取小波分解第一层的图像
a1=wrcoef2('a',c,s,'sym4');
subplot(223);image(a1);
title('第一次去噪图像');
axis square;
%提取小波分解第二层的图像
a2=wrcoef2('a',c,s,'sym4',2);
subplot(224);image(a2);title('第二次去噪图像');
axis square;
```