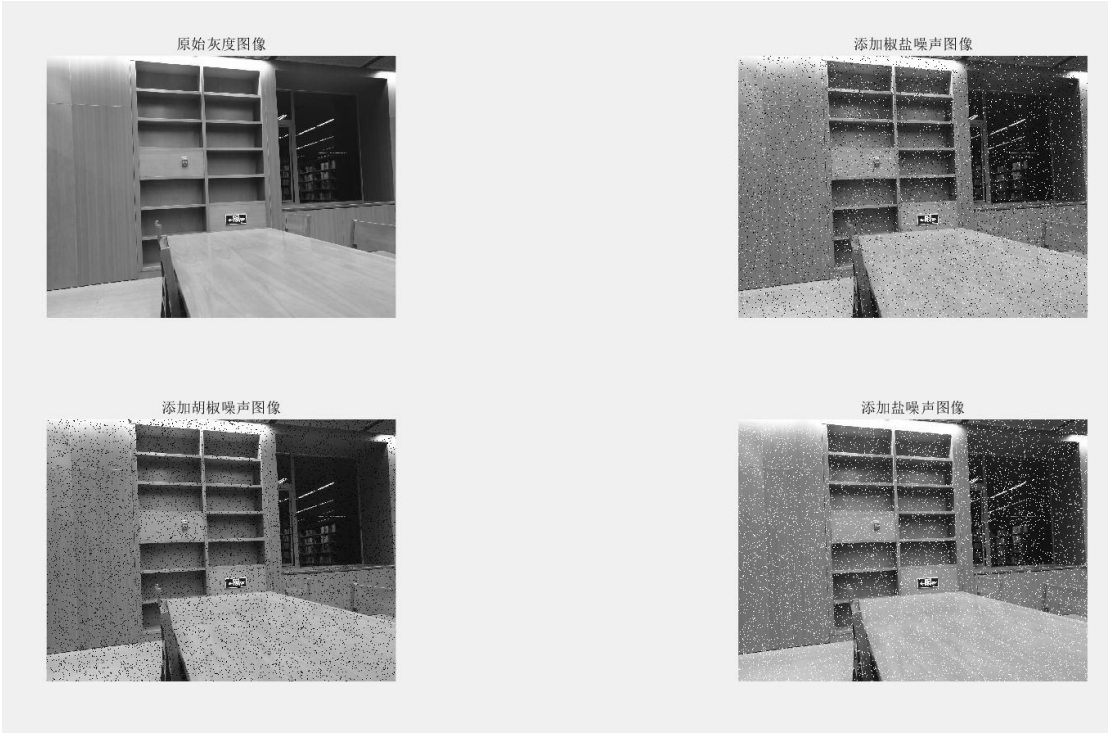
图像处理作业一

软工一班
陈仟雅
3017218053
2019.11.28

一、椒盐噪声

椒盐噪声是由图像传感器，传输信道，解码处理等产生的黑白相间的亮暗点噪声。椒盐噪声往往由图像切割引起。

椒盐噪声是指两种噪声，一种是盐噪声（salt noise），另一种是胡椒噪声（pepper noise）。盐=白色，椒=黑色。前者是高灰度噪声，后者属于低灰度噪声。一般两种噪声同时出现，呈现在图像上就是黑白杂点。



测试代码：

```
I=imread('a.jpg');
G=rgb2gray(I);
J = imnoise(G,'salt & pepper',0.05);
J1= imnoise(G,'salt & pepper',0.05);
J2= imnoise(G,'salt & pepper',0.05);
[M,N]=size(J);
for i=1:M
for j=1:N
if(J1(i,j)==255)
J1(i,j)=0;
else
```

```
J1(i,j)=J1(i,j);
end
end
end
for i=1:M
for j=1:N
if(J2(i,j)==0)
J2(i,j)=255;
else
J2(i,j)=J2(i,j);
end
end
end
A= meshgrid(1:3, 1:3);
A1= meshgrid(1:5, 1:5);
A2= meshgrid(1:8, 1:8);
figure(1)
subplot(221);imshow(G);title('原始灰度图像');
subplot(222);imshow(J);title('添加椒盐噪声图像');
subplot(223);imshow(J1);title('添加胡椒噪声图像');
subplot(224);imshow(J2);title('添加盐噪声图像');
```

二、中值滤波与均值滤波

1、中值滤波法是一种非线性平滑技术，它将每一像素点的灰度值设置为该点某邻域窗口内的所有像素点灰度值的中值。原理是把数字图像或数字序列中一点的值用该点的一个邻域中各点值的中值代替，让周围的像素值接近的真实值，从而消除孤立的噪声点。方法是用某种结构的二维滑动模板，将板内像素按照像素值的大小进行排序，生成单调上升（或下降）的为二维数据序列。但在条纹中心分析方法中作用不大。

实现方法：

（1）通过从图像中的某个采样窗口取出奇数个数据进行排序；

（2）用排序后的中值取代要处理的数据即可。

实际应用：

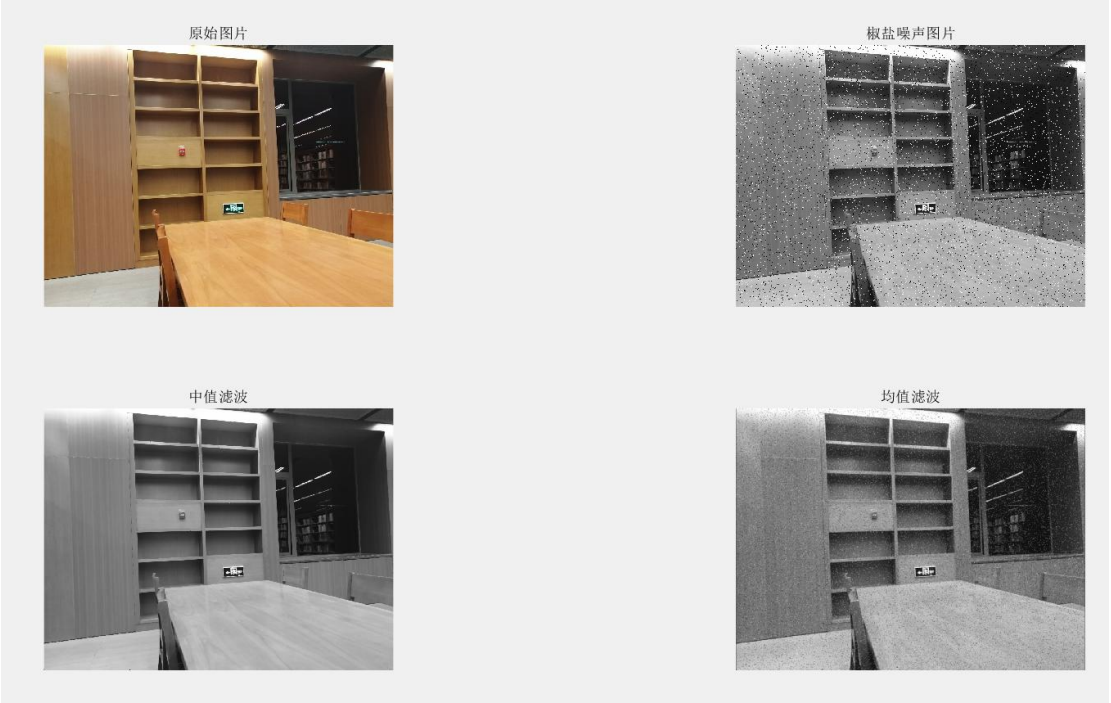
（1）中值滤波法对消除椒盐噪声非常有效，在光学测量条纹图象的相位分析处理方法中有特殊作用，但在条纹中心分析方法中作用不大。

（2）中值滤波在图像处理中，常用于保护边缘信息，是经典的平滑噪声的方法。

2、均值滤波是典型的线性滤波算法，它是指在图像上对目标像素给一个模板，该模板包括了其周围的临近像素（以目标象素为中心的周围 n 个像素，构成一个滤波模板，即去掉目标像素本身），再用模板中的全体像素的平均值来代替原来像素值。

均值滤波也称为线性滤波，其采用的主要方法为邻域平均法。线性滤波的基本原理是用均值代替原图像中的各个像素值，即对待处理的当前像素点（x，y），选择一个模板，该模板由其近邻的若干像素组成，求模板中所有像素的均值，再把该均值赋予当前像素点（x，y），作为处理后图像在该点上的灰度个 g（x，y），即个 g（x，y）=1/m ∑f（x，y） m 为该模板中包含当前像素在内的像素总个数。
不足之处：

均值滤波本身存在着固有的缺陷，即它不能很好地保护图像细节，在图像去噪的同时也破坏了图像的细节部分，从而使图像变得模糊，不能很好地去除噪声点。



明显可以看出，中值滤波消除噪点能力强于均值滤波。

测试代码：（接上面的椒盐噪声，以加了椒盐噪声的图片作为参考）

```
figure(2);
%原始图片
subplot(2,2,1);
imshow(I);
title('原始图片');
data=J;%读入图片，图片复制到当前文件夹
subplot(2,2,2);
imshow(data);
title('椒盐噪声图片');
%中值滤波
mdata=medfilt2(data);
subplot(2,2,3);
imshow(mdata);
title('中值滤波');
%均值滤波
```

```
h=fspecial('average');%创建一个均值模板
fdata=imfilter(data,h);%前面是图片，后面是模板
subplot(2,2,4);
imshow(fdata);
title('均值滤波');
```

三、最大值滤波与最小值滤波

最大值、最小值滤波与中值滤波相类似，先是排序像素值，而后对中心像素值和最大、最小像素值进行比较。若比最小值小，则将最小值替换成中心像素；同样若比最大值大，则将最大值替换成中心像素。

若设输出的图像为 g（x，y）

最大值滤波：g（x，y）max{f（x s，y t），（s，t W）};

最小值滤波：g（x，y） min{f（x s，y t），（s，t W）}。

测试代码：（接上面的椒盐噪声，以加了椒盐噪声的图片作为参考）

```
figure(3);
%原始图片
subplot(2,2,1);
imshow(I);
title('原始图片');
subplot(2,2,2);
imshow(J);
title('椒盐噪声图片');
%中值滤波
Jmin= ordfilt2(J,1,ones(3,3));
subplot(2,2,3)
imshow(Jmin);
title('最小值滤波');
```

```
%均值滤波
Jmax= ordfilt2(J,9,ones(3,3));
subplot(2,2,4)
imshow(Jmax);
title('最大值滤波');
```

四、自适应中值滤波器

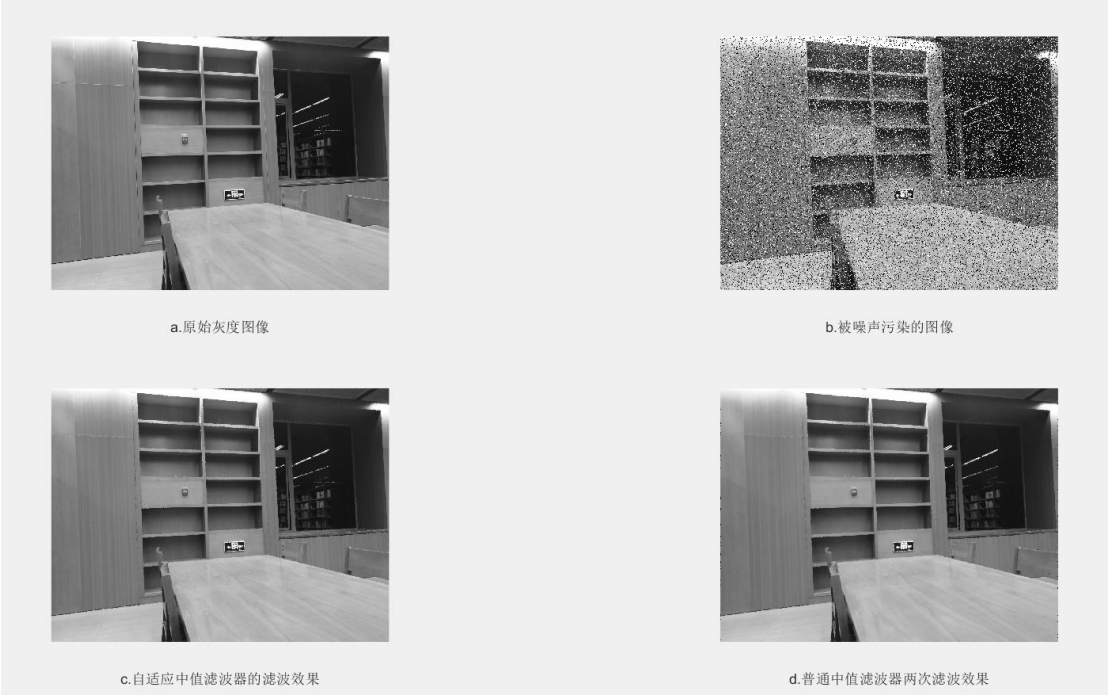
常规的中值滤波器的窗口尺寸是固定大小不变的，就不能同时兼顾去噪和保护图像的细节。这时就要寻求一种改变，根据预先设定好的条件，在滤波的过程中，动态的改变滤波器的窗口尺寸大小，这就是自适应中值滤波器 Adaptive Median Filter。在滤波的过程中，自适应中值滤波器会根据预先设定好的条件，改变滤波窗口的尺寸大小，同时还会根据一定的条件判断当前像素是不是噪声，如果是则用邻域中值替换掉当前像素；不是，则不作改变。

自适应中值滤波器有三个目的：

(1) 滤除椒盐噪声

(2) 平滑其他非脉冲噪声

(3) 尽可能的保护图像中细节信息，避免图像边缘的细化或者粗化。



a.原始灰度图像　　b.被噪声污染的图像

c.自适应中值滤波器的滤波效果　　d.普通中值滤波器两次滤波效果

测试代码：

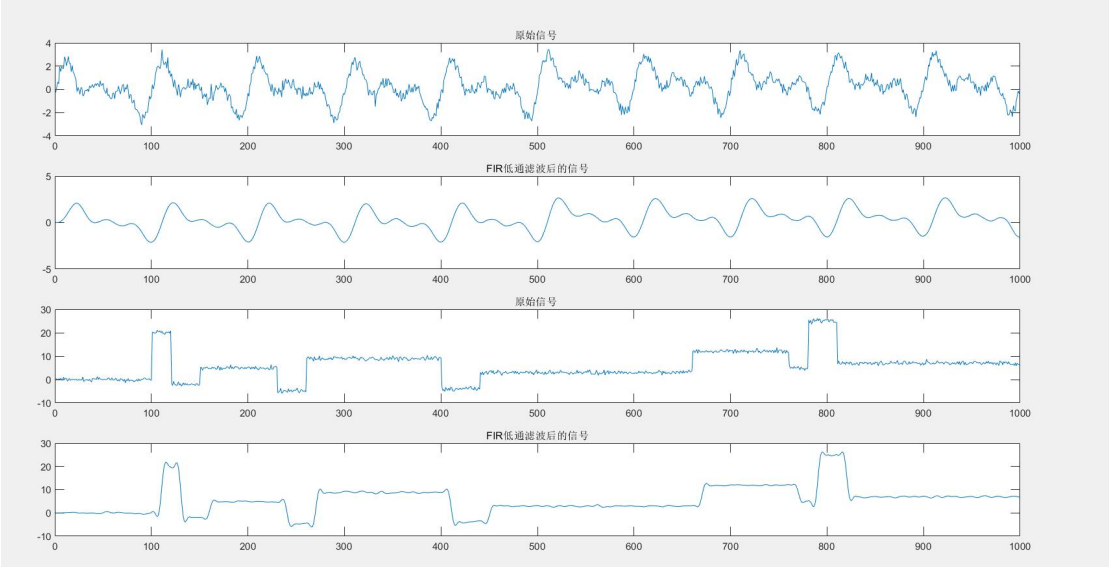
```
%% 读入图像 I
I=imread('a.jpg');
%转化为灰度图 Ig
Ig=rgb2gray(I);
%被密度为 0.2 的椒盐噪声污染的图像 Inoise
Inoise=imnoise(Ig,'salt & pepper',0.2);
%或者是被方差为 0.2 的高斯噪声污染的图像 Inoise
%Inoise=imnoise(Ig,'gaussian',0.2);
%显示原图的灰度图 Ig 和噪声图像 Inoise
```

```
subplot(2,2,1),imshow(Ig);xlabel('a.原始灰度图像');
subplot(2,2,2),imshow(Inoise);xlabel('b.被噪声污染的图像');

%% 定义参数
%获取图像尺寸:Im,In
[Im,In]=size(Inoise);
%起始窗口尺寸:nmin*nmin(窗口尺寸始终取奇数)
nmin=3;
%最大窗口尺寸:nmax*nmax
nmax=9;
%定义复原后的图像 Imf
Imf=Inoise;
%为了处理到图像的边界点，需将图像扩充
%因为窗口尺寸是弹性的，所以将 Inoise 固定扩充到最
大:I_ex[(Im+(nmax-1))*(In+(nmax-1))]
I_ex=[zeros((nmax-1)/2,In+(nmax-1));zeros(Im,(nmax-1)/2),Inoise,zeros
(Im,(nmax-1)/2);zeros((nmax-1)/2,In+(nmax-1))];
%% 自适应滤波过程
%遍历图像 Inoise 中的每一点
for x=1:Im
    for y=1:In
        for n=nmin:2:nmax
                %图像 Inoise 中的某点(x,y)的领域 Sxy，对应在 I_ex 中为
(x+[(nmax-1)/2-(n-1)/2]:x+[(nmax-1)/2-(n-1)/2]+(n-1),y+(nmax-1)/2-(n-
1)/2:y+[(nmax-1)/2-(n-1)/2]+(n-1))

Sxy=I_ex(x+(nmax-1)/2-(n-1)/2:x+(nmax-1)/2+(n-1)/2,y+(nmax-1)/2-(n-1)
/2:y+(nmax-1)/2+(n-1)/2);
                Smax=max(max(Sxy));%求出窗口内像素的最大值
                Smin=min(min(Sxy));%求出窗口内像素的最小值
                Smed=median(median(Sxy));%求出窗口内像素的中值
                %判断中值是否是噪声点
                if Smed>Smin && Smed<Smax
                   %若中值既大于最小值又小于最大值，则不是
                   %是，则退出该 if 语句，增大窗口尺寸，再次判断
                   %不是，则判断该点的原值是不是噪声点
                   if Imf(x,y)<=Smin || Imf(x,y)>=Smax
                   %若该点的原值既大于最小值又小于最大值，则不是
                   %不是，则输出原值，即不作处理
                   %是，则输出中值
                      Imf(x,y)=Smed;
                   end
                   break
                   %有输出则不再进行循环判断
```

```
                end
            end
            %当 n=max 时，输出中值
            Imf(x,y)=Smed;
        end
end
subplot(2,2,3),imshow(Imf);xlabel('c.自适应中值滤波器的滤波效果');
%% 与普通中值滤波器的对比
Imf1=medfilt2(Inoise,[3,3]);
Imf2=medfilt2(Imf1,[3,3]);
subplot(2,2,4),imshow(Imf2);xlabel('d.普通中值滤波器两次滤波效果');
```

五、低通滤波

低通滤波器是容许低于截止频率的信号通过，但高于截止频率的信号不能通过的电子滤波装置。



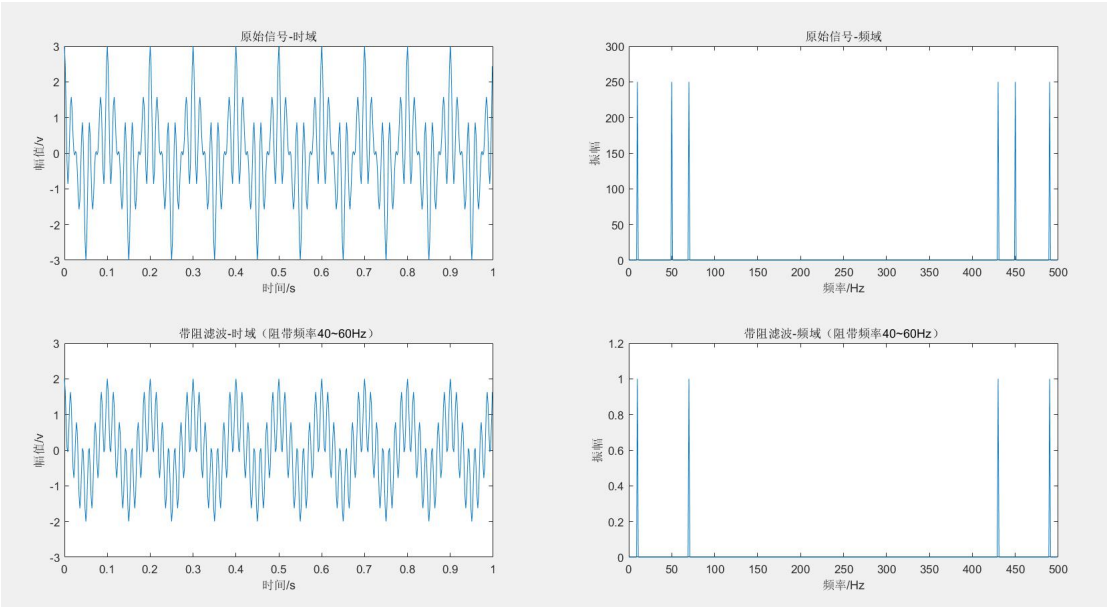
测试代码：

```
%创建两个信号 Mix_Signal_1 和信号 Mix_Signal_2
Fs = 1000;          %采样率
N = 1000;           %采样点数
n = 0:N-1;
t = 0:1/Fs:1-1/Fs;          %时间序列
Signal_Original_1 = sin(2*pi*10*t)+sin(2*pi*20*t)+sin(2*pi*30*t);
Noise_White_1 = [0.3*randn(1,500), rand(1,500)];          %前 500 点高斯分部白噪声，后 500 点均匀分布白噪声
Mix_Signal_1 = Signal_Original_1 + Noise_White_1;          %构造的混合信号

Signal_Original_2 = [zeros(1,100), 20*ones(1,20), -2*ones(1,30), 5*ones(1,80), -5*ones(1,30), 9*ones(1,140), -4*ones(1,40),
```

```
3*ones(1,220), 12*ones(1,100), 5*ones(1,20), 25*ones(1,30), 7
*ones(1,190)];
Noise_White_2 = 0.5*randn(1,1000);        %高斯白噪声
Mix_Signal_2 = Signal_Original_2 + Noise_White_2;        %构造的混合信
号
%混合信号 Mix_Signal_1  FIR 低通滤波
figure(2);
F = [0:0.05:0.95];
A = [1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];
b = firls(20,F,A);
Signal_Filter = filter(b,1,Mix_Signal_1);

subplot(4,1,1);        %Mix_Signal_1 原始信号
plot(Mix_Signal_1);
axis([0,1000,-4,4]);
title('原始信号 ');

subplot(4,1,2);        %Mix_Signal_1 FIR 低通滤波滤波后信号
plot(Signal_Filter);
axis([0,1000,-5,5]);
title('FIR 低通滤波后的信号');

%混合信号 Mix_Signal_2  FIR 低通滤波
F   =  [0:0.05:0.95];
A  =  [1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0] ;
b  =  firls(20,F,A);
Signal_Filter = filter(b,1,Mix_Signal_2);
subplot(4,1,3);        %Mix_Signal_2 原始信号
plot(Mix_Signal_2);
axis([0,1000,-10,30]);
title('原始信号 ');

subplot(4,1,4);        %Mix_Signal_2 FIR 低通滤波滤波后信号
plot(Signal_Filter);
axis([0,1000,-10,30]);
title('FIR 低通滤波后的信号');
```

六、带阻滤波

带阻滤波器（bandstop filters，简称 BSF）是指能通过大多数频率分量、但将某些范围的频率分量衰减到极低水平的滤波器，与带通滤波器的概念相对。其中点阻滤波器（notch filter）是一种特殊的带阻滤波器，它的阻带范围极小，有着很高的 Q 值（Q Factor）。



测试代码：

```
%% 对 周期函数 使用带阻滤波（BSF）
clear all;
close all;
%% 构建原始信号
N = 500; %原始信号长度：点数
Fs = 500; %采样频率：Hz
Dt = 1/Fs; %采样间隔时间：s
t = [0:N-1]*Dt; %时间序列：s
f1 = 10;f2 = 50;f3 = 70; %原始信号频率
y = cos(2*f1*t*pi)+cos(2*f2*t*pi)+cos(2*f3*t*pi);
subplot(2,2,1);plot(t,y); %时域信号图
title('原始信号-时域');xlabel('时间/s');ylabel('幅值/v');
% xlim([0 12]);ylim([-1.5 1.5]);
%% FFT 变换
FN = N; %FFT 执行长度：点数
f0 = 1/(Dt*FN); %基频
Fy = fft(y);  %对时域信号进行 FFT 变换
mag = abs(Fy);
n = 0:FN-1;
Ff = n*f0;  %频率序列
subplot(2,2,2);plot(Ff,mag); %绘制原始信号的振幅图
title('原始信号-频域');xlabel('频率/Hz');ylabel('振幅');
% ylim([0 0.8]);xlim([0 50]);
%% 带阻滤波器 BSF
BN = N;
Bn = 0:BN-1;
fmax = 60;
fmin = 40;
```

```
By = zeros(1, length(y));
for m = Bn
    if (m*Fs/BN>fmin & m*Fs/BN<fmax) | (m*Fs/BN>(Fs-fmax) &
m*Fs/BN<(Fs-fmin));
        By(m+1) = 0;
    else
        if m<BN-1;
            By(m+1) = Fy(m+1);
        end
    end
end
subplot(2,2,4);plot(Ff,abs(By)*2/BN);
title('带阻滤波-频域（阻带频率 40~60Hz）');xlabel('频率/Hz');ylabel('
振幅');
subplot(2,2,3);plot(t,real(ifft(By)));
title('带阻滤波-时域（阻带频率 40~60Hz）');xlabel('时间/s');ylabel('幅
值/v');
```

七、维纳滤波、逆滤波、约束最小二乘方

1、维纳滤波（wiener filtering）一种基于最小均方误差准则、对平稳过程的最优估计器。这种滤波器的输出与期望输出之间的均方误差为最小，因此，它是一个最佳滤波系统。它可用于提取被平稳噪声污染的信号。

从连续的（或离散的）输入数据中滤除噪声和干扰以提取有用信息的过程称为滤波，这是信号处理中经常采用的主要方法之一，具有十分重要的应用价值，而相应的装置称为滤波器。根据滤波器的输出是否为输入的线性函数，可将它分为线性滤波器和非线性滤波器两种。维纳滤波器是一种线性滤波器。

2、逆滤波复原过程：对退化的图像进行二位傅里叶变换；计算系统点扩散函数的二位傅里叶变换；引入 H（fx,fy）计算并且对结果进行逆傅里叶变换。

3、约束最小二乘方滤波（Constrained Least Squares Filtering，aka Tikhonov filtration,Tikhonov regularization）核心是 H 对噪声的敏感性问题。减少噪声敏感新问题的一种方法是以平滑度量的最佳复原为基础的，因此我们可以建立下列约束条件：

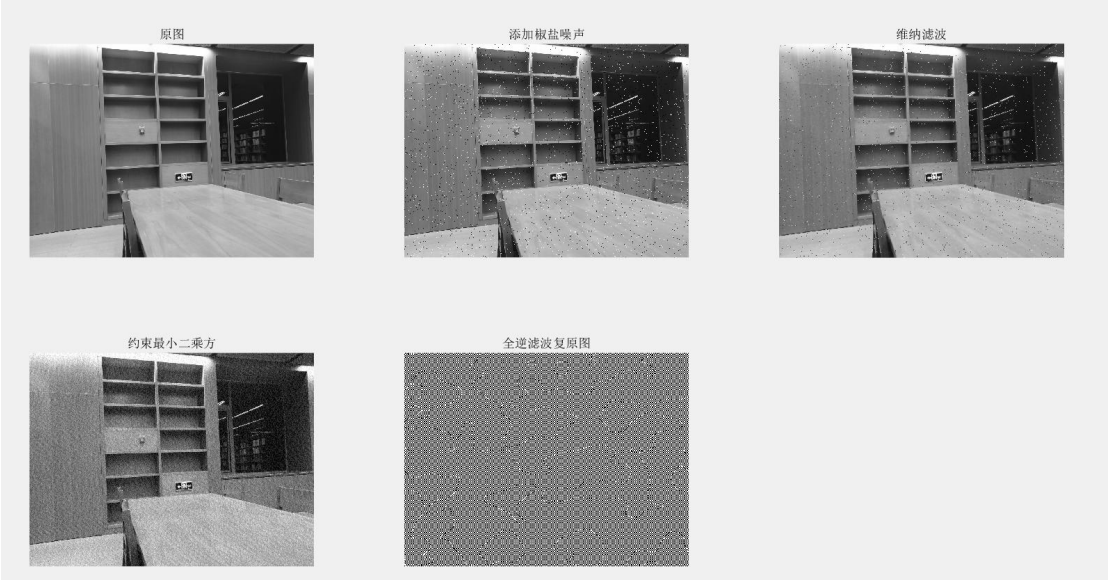
$$C = \sum_{0}^{M-1}\sum_{0}^{N-1}\left[\nabla^2 f(x,y)\right]^2 \quad (1)$$

其约束条件为

$$||G - H\hat{F}||_2^2 = ||N||_2^2 \quad (2)$$

这里，$\hat{F}$是为退化图像的估计，N为加性噪声，拉普拉斯算子∇^2在这里表示平滑程度。

约束最小二乘方滤波要求噪声的方差和均值，这些参数可通过给定的退化图像计算出来，这是约束最小二乘方滤波的一个重要优点。



测试代码：

```
clear;clc;close all
I=imread('a.jpg');  %读入图片
I=rgb2gray(I);  %rgb 图转换成灰度图
subplot(2,3,1);imshow(I);
title('原图')
I1=imnoise(I,'salt & pepper',0.02);  %原图添加椒盐噪声，得到添加噪声之后的图 I1
subplot(2,3,2);imshow(I1)
title('添加椒盐噪声')

%维纳滤波
I4=wiener2(double(I1),[5 5]);  %对 I1 进行维纳滤波
subplot(2,3,3);imshow(uint8(I4))
title('维纳滤波')

%约束最小二乘方
[hei,wid,~] = size(I);
LEN = 21;
THETA = 11;
PSF = fspecial('motion', LEN, THETA);
blurred = imfilter(I, PSF, 'conv', 'circular');
If = fft2(blurred);
Pf = psf2otf(PSF,[hei,wid]);
```

```
% Simulate additive noise.
noise_mean = 0;
noise_var = 0.00001;
blurred_noisy = imnoise(blurred, 'gaussian',noise_mean, noise_var);
%subplot(2,3,7), imshow(blurred_noisy); title('模拟模糊和噪声')
% Try restoration using  Home Made Constrained Least Squares Filtering.
p = [0 -1 0;-1 4 -1;0 -1 0];
P = psf2otf(p,[hei,wid]);
gama = 0.001;
If = fft2(blurred_noisy);
numerator = conj(Pf);
denominator = Pf.^2 + gama*(P.^2);
subplot(2,3,4); imshow(deconvreg(blurred_noisy, PSF,0)); title('约束最
小二乘方');


%逆滤波
[m,n]=size(I);
F=fftshift(fft2(I));
k=0.0025;
for u=1:m
    for v=1:n
        H(u,v)=exp((-k)*(((u-m/2)^2+(v-n/2)^2)^(5/6)));
    end
end
G=F.*H;
I0=real(ifft2(fftshift(G)));
I1=imnoise(uint8(I0),'gaussian',0,0.001);
F0=fftshift(fft2(I1));
F1=F0./H;
I2=ifft2(fftshift(F1));
subplot(2,3,5);imshow(uint8(I2));title('全逆滤波复原图');
```