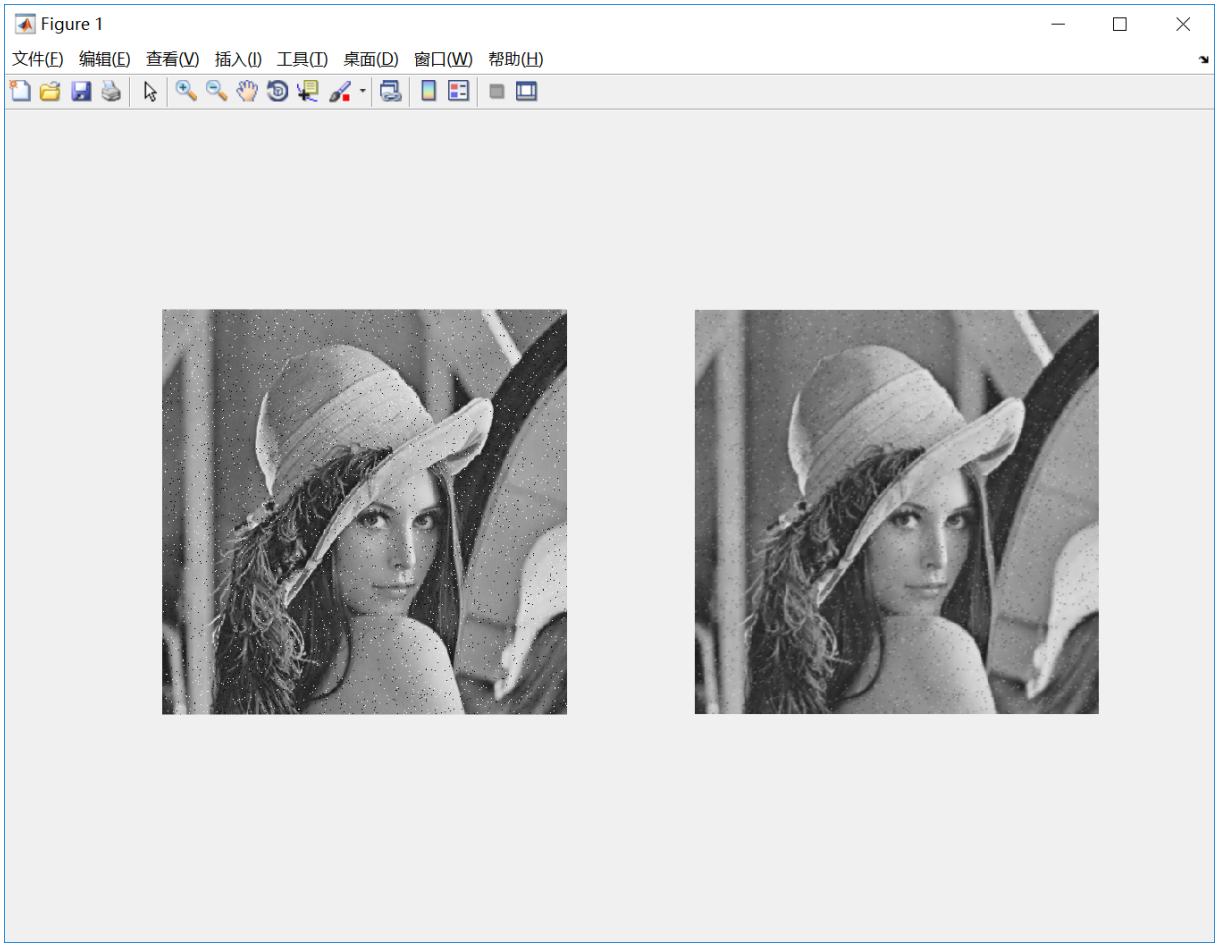
图像处理第三次作业

- 平滑滤波
  - 代码

```
img = imread('lena.jpg');
img = imnoise(img,'salt & pepper',0.02);
figure;
subplot(1,2,1);
imshow(img);
[h, w] = size(img);

for i = 1 : h
    for j = 1 : w
        up = max(i - 1, 1);
        down = min(i + 1, h);
        left = max(j - 1, 1);
        right = min(j + 1, w);
        img(i, j) = mean(mean(img(up : down, left : right)));
    end
end
subplot(1,2,2);
imshow(img);
```

  - 运行结果



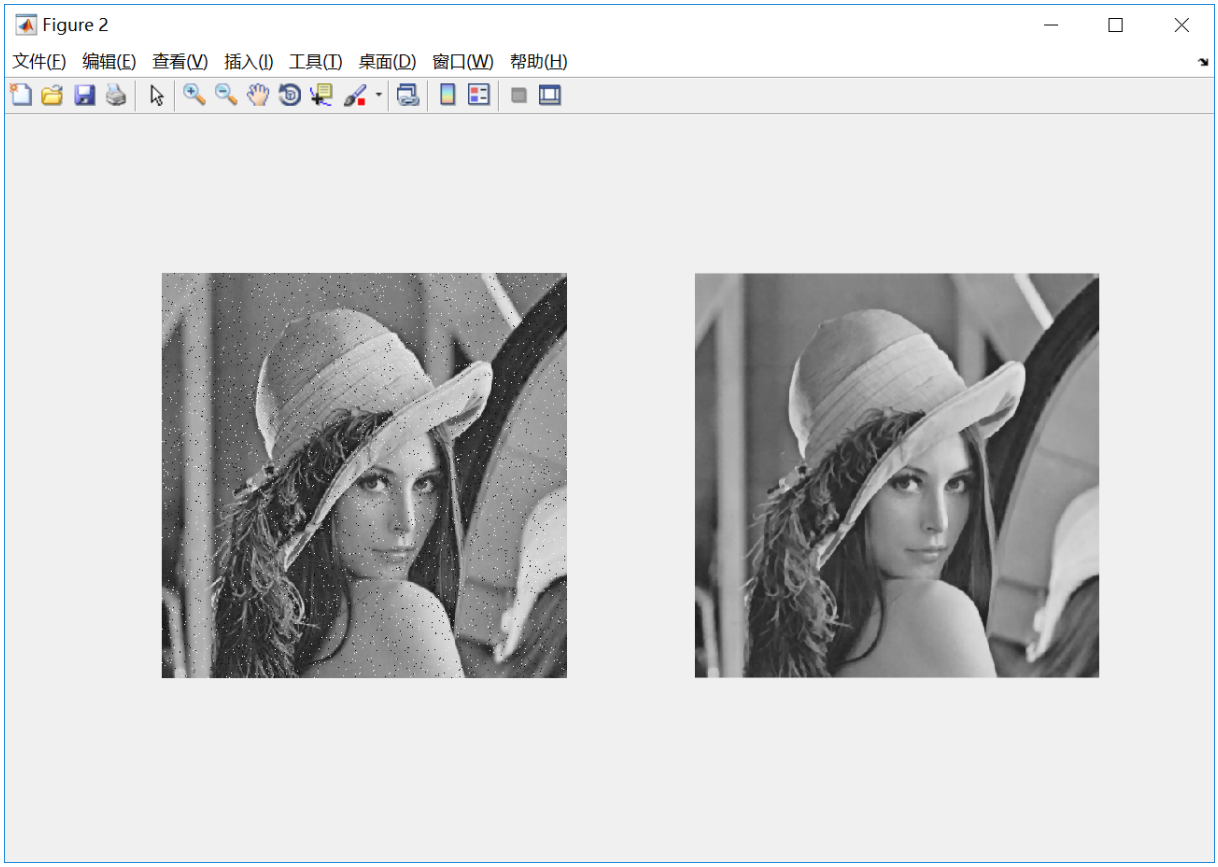
- 中值滤波
  - 代码

```
img = imread('lena.jpg');
img = imnoise(img,'salt & pepper',0.02);
figure;
subplot(1,2,1);
imshow(img);
[h, w] = size(img);

for i = 1 : h
    for j = 1 : w
        up = max(i - 1, 1);
        down = min(i + 1, h);
        left = max(j - 1, 1);
        right = min(j + 1, w);
        sub = img(up : down, left : right);
        sub = sub(:);
        img(i, j) = median(sub);
    end
end

subplot(1,2,2);
imshow(img);
```

- 运行结果

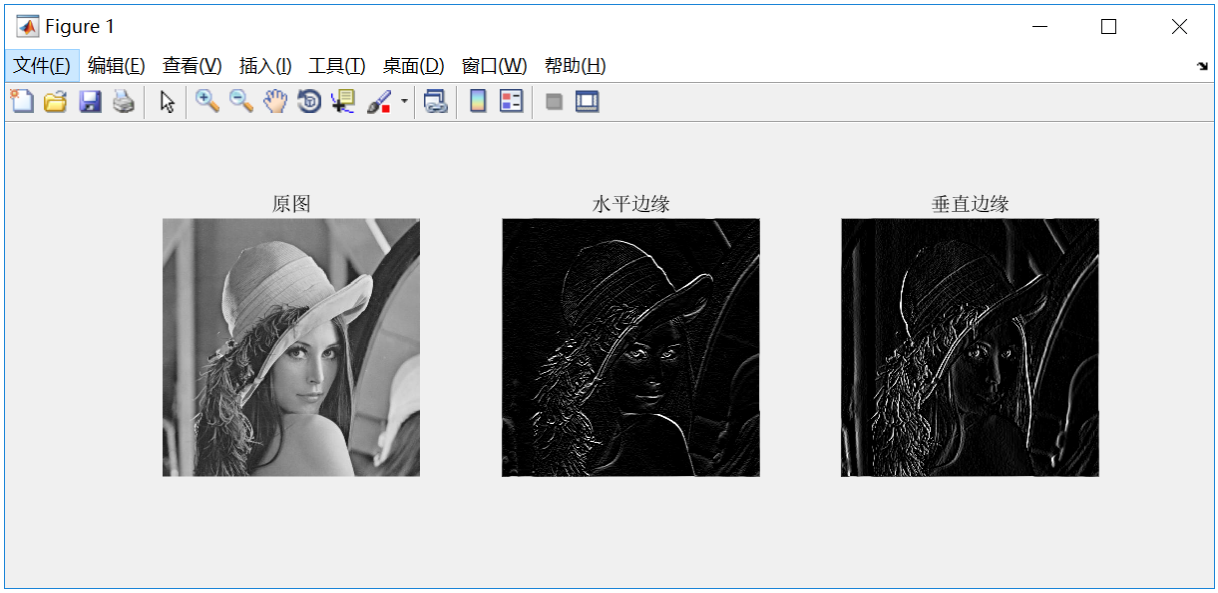


- sobel算子
  - 代码

```
img=imread('lena.jpg');
subplot(131);imshow(img);title('原图');
[high,width] = size(img);
F2 = double(img);
U = double(img);
img_x = img;
img_y = img;
for i = 2:high - 1
    for j = 2:width - 1
        Gx = (U(i+1,j-1) + 2*U(i+1,j) + F2(i+1,j+1)) - (U(i-1,j-1) + 2*U(i-1,j) +
F2(i-1,j+1));
        Gy = (U(i-1,j+1) + 2*U(i,j+1) + F2(i+1,j+1)) - (U(i-1,j-1) + 2*U(i,j-1) +
F2(i+1,j-1));
        img_x(i,j) = Gx;
        img_y(i,j) = Gy;
    end
end

subplot(132);imshow(im2uint8(img_x));title('水平边缘');
subplot(133);imshow(im2uint8(img_y));title('垂直边缘');
```

  - 运行结果

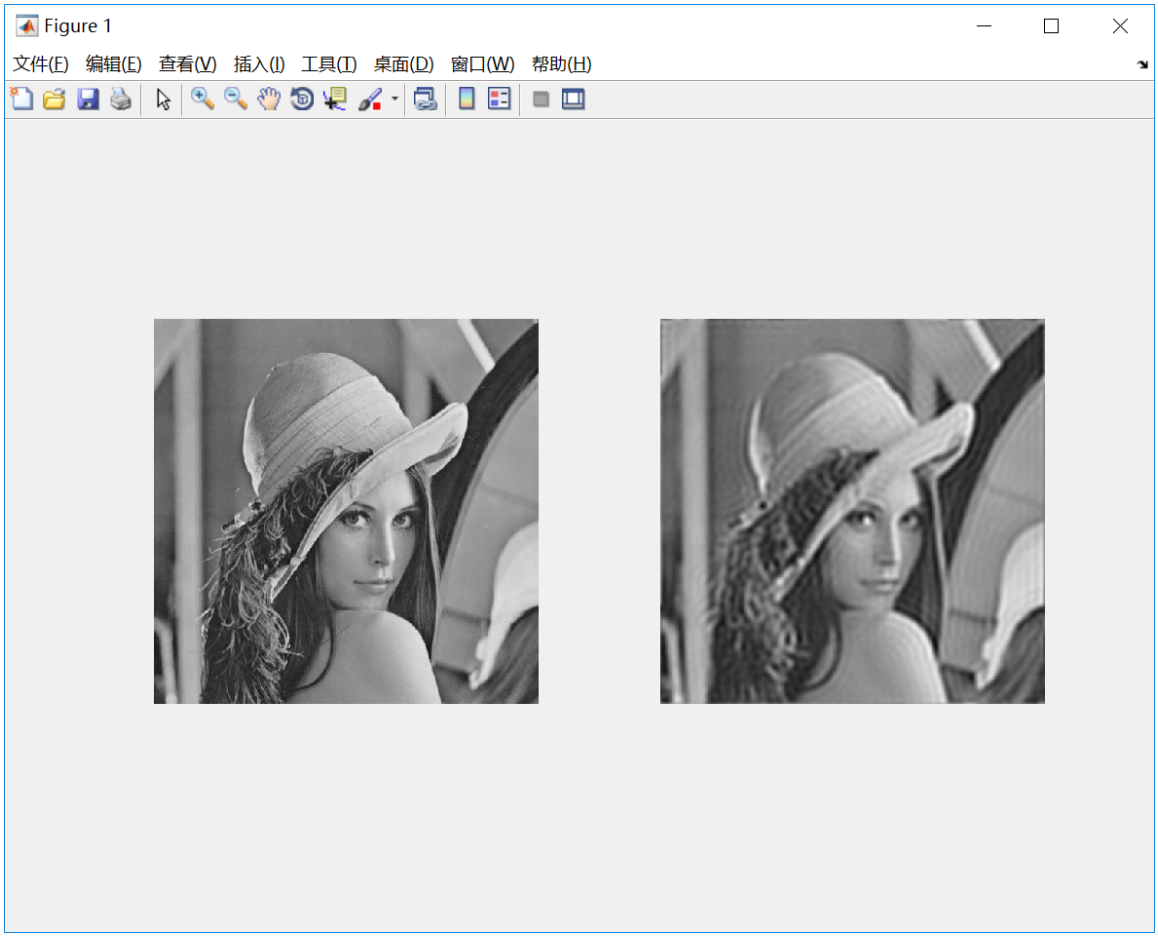


- 低通滤波
  - 理想情况
    - 代码

```
img = imread('lena.jpg');
subplot(121);
imshow(img);
img = double(img);
f = fft2(img);
g = fftshift(f);

[M, N] = size(f);
n1 = floor(M/2);
n2 = floor(N/2);
d0 = 45;
for i=1:M
    for j=1:N
        d=sqrt((i-n1)^2+ (j-n2)^2);
        if d<=d0
            h=1;
        else
            h=0;
        end
        g(i,j)=h*g(i,j);
    end
end
g = ifftshift(g);
g = uint8(real(ifft2(g)));
subplot(122);
imshow(g);
```

    - 运行结果

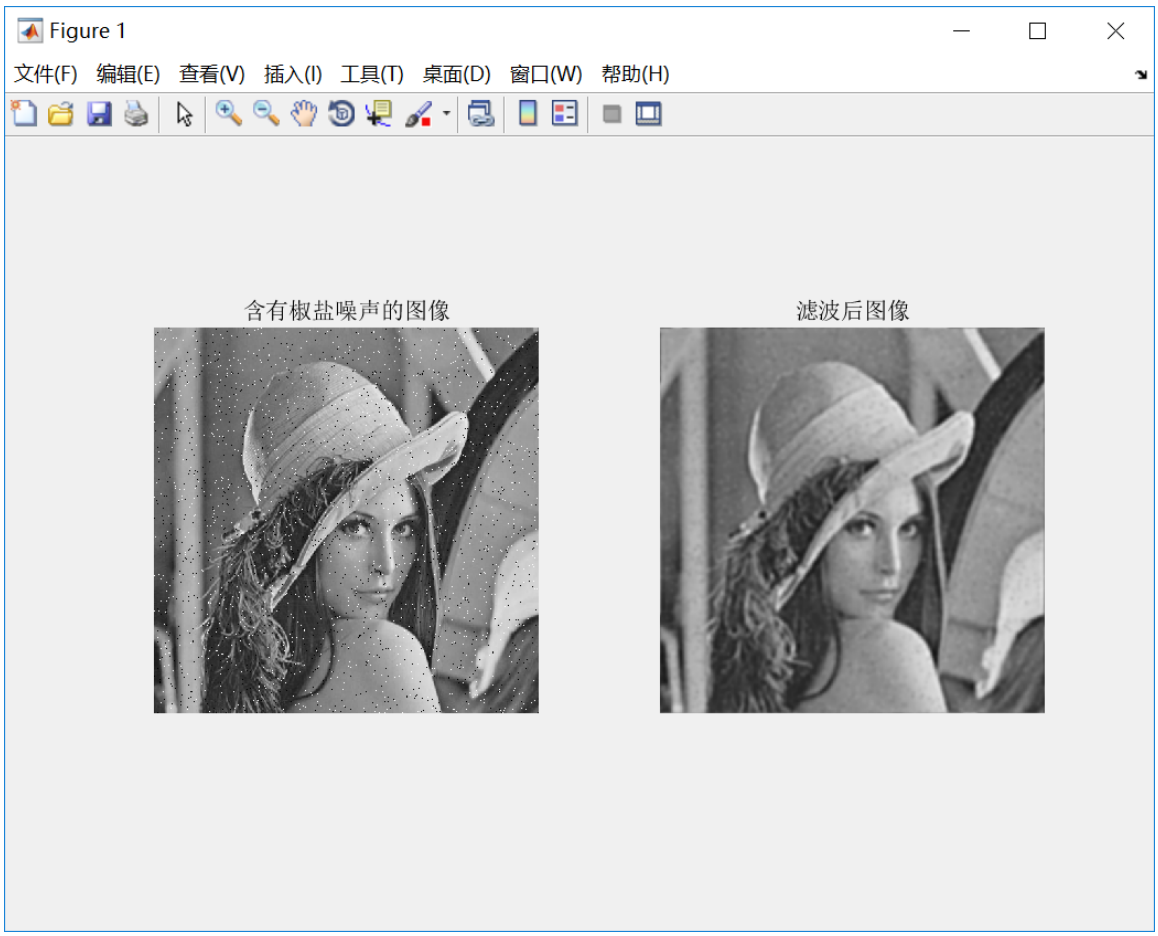


- 巴特沃斯
  - 代码

```
img = imread('lena.jpg');
img = imnoise(img, 'salt & pepper',0.02);
subplot(121);
imshow(img);
title('含有椒盐噪声的图像');
img = double(img);
f = fft2(img);
g= fftshift(f);
[M, N]=size(f);
n=3;
d0=55;
n1=floor(M/2);
n2=floor(N/2);
for i=1:M
    for j=1:N
        d = sqrt((i-n1)^2+(j-n2)^2);
        h=1/(1+(d/d0)^(2*n));
        g(i,j)=h*g(i,j);
    end
end
g=ifftshift(g);
g=uint8(real(ifft2(g)));
subplot(122);
```

```
imshow(g);
title('滤波后图像');
```

- 运行结果



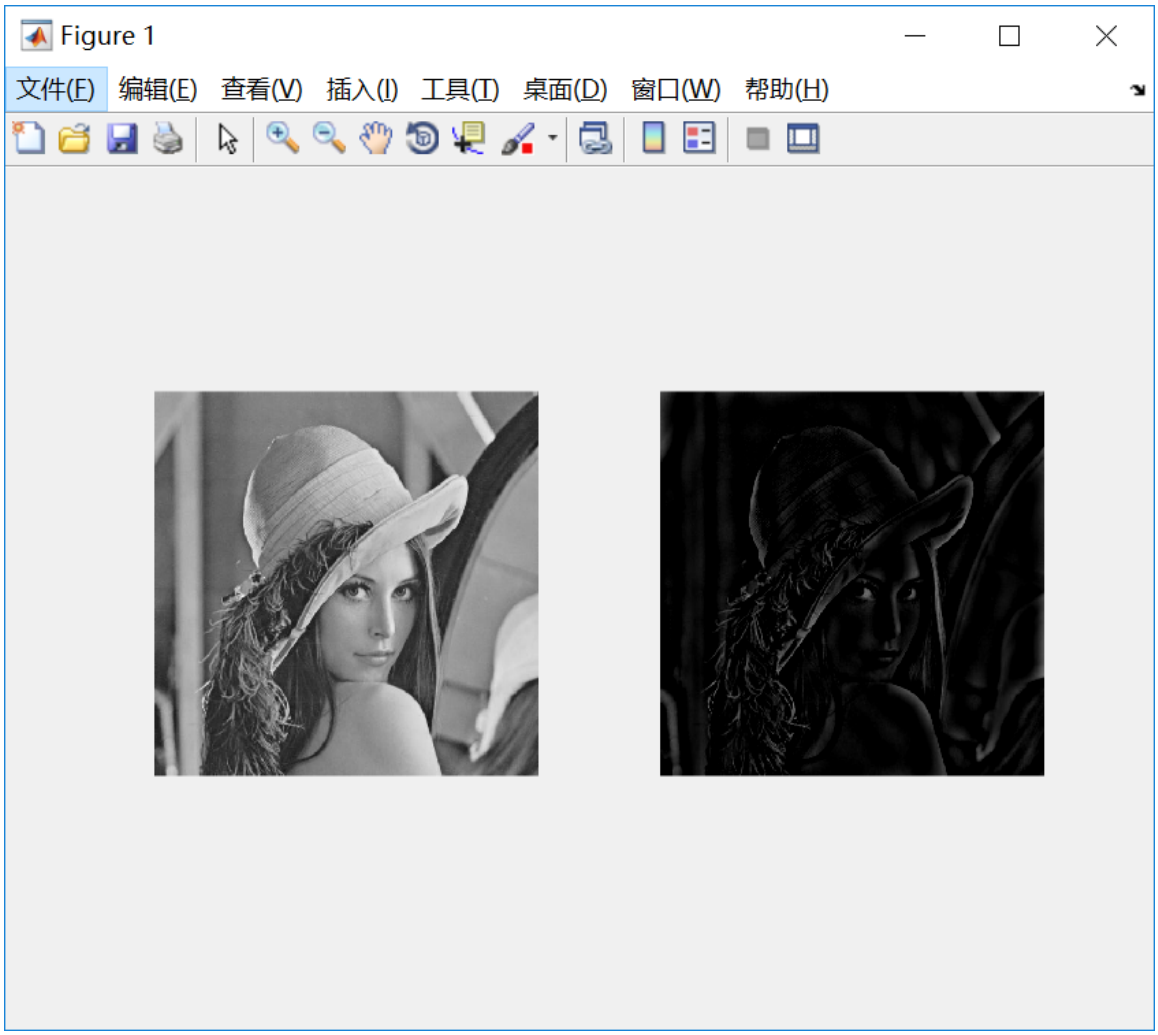
- 高通滤波
  - 理想情况
    - 代码

```
img = imread('lena.jpg');
subplot(121);
imshow(img);
img = double(img);
f = fft2(img);
g = fftshift(f);

[M, N] = size(f);
n1 = floor(M/2);
n2 = floor(N/2);
d0 = 10;
for i=1:M
    for j=1:N
        d=sqrt((i-n1)^2+ (j-n2)^2);
        if d<=d0
            h=0;
```

```
        else
            h=1;
        end
        g(i,j)=h*g(i,j);
    end
end
g = ifftshift(g);
g = uint8(real(ifft2(g)));
subplot(122);
imshow(g);
```

- 运行结果



- 巴特沃斯
  - 代码

```
img = imread('lena.jpg');
subplot(121);
imshow(img);
title('含有椒盐噪声的图像');
img = double(img);
f = fft2(img);
g= fftshift(f);
```

```
[M, N]=size(f);
n=3;
d0=10;
n1=floor(M/2);
n2=floor(N/2);
for i=1:M
    for j=1:N
        d = sqrt((i-n1)^2+(j-n2)^2);
        h=1/(1+(d0/d)^(2*n));
        g(i,j)=h*g(i,j);
    end
end
g=ifftshift(g);
g=uint8(real(ifft2(g)));
subplot(122);
imshow(g);
title('滤波后图像');
```

- 运行结果