数字图像处理第六次作业

姓名：吴志朋
班级：自动化 62
学号：2160504050

1、在测试图像上产生高斯噪声 lena 图-需能指定均值和方差；并用滤波器恢复图像；

（1）问题分析：

图像退化模型：

图像退化过程被建模为一个退化函数和一个加性噪声项，对一幅输入图像 f(x,y)进行处理，产生一副退化后的图像 g(x,y)。给定 g(x,y)和关于退化函数 H 的一些知识以及关于加性噪声项 η(x,y)的一些知识后，图像复原的目的就是获得原始图像的一个估计。

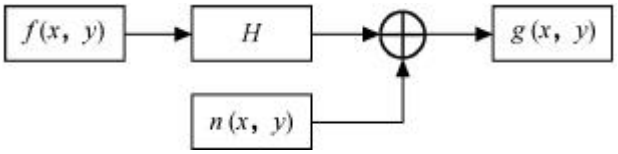
如果 H 是一个线性的、位置不变的过程，那么空间域中的退化图像可由下式给出：

$$g(x,y)=h(x,y)\bigstar f(x,y)+\eta(x,y)$$

其中，h(x,y)是退化函数的空间表示；符号'★"表示空间卷积.

等价的频率域表示：

$$G(u,v)=H(u,v)F(u,v)+N(u,v)$$

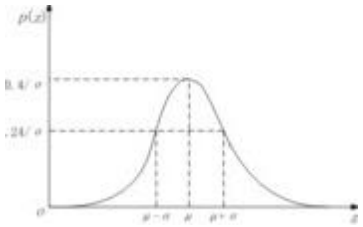


高斯噪声：

所谓高斯噪声是指它的概率密度函数服从高斯分布（即正态分布）的一类噪声。一个高斯随机变量 z 的 PDF 可表示为：

$$P(z)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left[\frac{(z-u)^2}{2\sigma^2}\right]$$

其中 z 代表灰度，u 是 z 的均值，是 z 的标准差。高斯噪声的灰度值多集中在均值附近。



算术均值滤波器：

令 Sxy 表示中心在点(x,y)处，大小为 m×n 的矩形子图像窗口的一组坐标。算术均值滤波器在 Sxy 定义的区域中计算被污染的图像 g(x,y)的平均值。在点(x,y)处复原图像的值：

$$\hat{f}(x,y)=\frac{1}{mn}\sum_{(s,t)\in S_{xy}}g(s,t)$$

这个操作可以使用大小为 m×n 的一个空间滤波器来实现，其所有系数均为其值的 1/mn。均值滤波器平滑一幅图像中的局部变化，虽然模糊了结果，但降低了噪声。

中值滤波器：

统计排序滤波器是空间域滤波器，空间滤波器的响应是基于由该滤波器包围的图像区域中的像素值的排序。中值滤波器使用一个像素邻域中灰度级的中值来替代该像素值，即：

$$\hat{f}(x,y)=\underset{(s,t)\in S_{xy}}{median}\{g(s,t)\}$$

编程思路：

首先利用 imnoise 函数给图像添加高斯噪声，之后分别利用算术平均值滤波器和中值滤波器进行滤波并观察效果。

（2）MATLAB 函数：

g=imnoise(f,type,parameters)

函数功能：使用函数 imnoise 来用噪声污染一幅图像、

调用格式：

J = imnoise(I,type)

J = imnoise(I,type,parameters)

参数 Type 对应的噪声类型如下：

'gaussian'高斯白噪声

'localvar'0 均值白噪声

'poisson'泊松噪声

'salt & pepper'盐椒噪声

'speckle'乘性噪声

（3）处理结果：

1）添加高斯噪声

lena.bmp 原始图像

lena 添加高斯噪声（u=0，s^2=0.01）

lena 添加高斯噪声（u=0，s^2=0.05）

lena 添加高斯噪声（u=0，s^2=0.1）

lena 添加高斯噪声（u=0.1，s^2=0.01）

lena 添加高斯噪声（u=0.5，s^2=0.01）


2）图像恢复（选取被均值为 0，方差为 0.01 的高斯噪声污染的图像为例）

利用算术均值滤波器恢复图像(5x5 模板)

lena 添加高斯噪声（u=0，s^2=0.01）

算术均值滤波的结果

利用中值滤波器恢复图像 (5x5 模板)

lena 添加高斯噪声（u=0，s^2=0.01）

中值滤波的结果

（4）结果分析及总结：

①首先通过 imnoise 函数分别产生了被不同均值和方差的高斯噪声污染的图像。当高斯噪声均值不变为 0 时，随着方差增加，图像噪声越严重；当高斯噪声方差不变时，均值会影响到整个图像的灰度值，使整个图像变亮。与理论上均值和方差对图像的影响一致。
②分别使用算术均值滤波器和中值滤波器对加噪图像进行恢复。两种方法在一定程度上都可以降低噪声。算术均值滤波器降低噪声的同时也模糊了图像。

2、推导维纳滤波器并实现下边要求；

(a) 实现模糊滤波器如方程 Eq. (5.6-11).

(b) 模糊 lena 图像：45 度方向，T=1；

(c) 再模糊的 lena 图像中增加高斯噪声，均值= 0 ，方差=10 pixels 以产生模糊图像；

(d)分别利用方程 Eq. (5.8-6)和(5.9-4)，恢复图像。

（1）问题分析：

1）维纳滤波器的推导：

图像的退化模型为：

$$x(\mathrm{n}_1,\mathrm{n}_2)=b(\mathrm{n}_1,\mathrm{n}_2)*s(\mathrm{n}_1,\mathrm{n}_2)+\mathrm{w}(\mathrm{n}_1,\mathrm{n}_2) \tag{1}$$

其中，s(n1,n2)为原始图像，b(n1,n2)为退化函数，w(n1,n2)为噪声函数，x(n1,n2)为退化的图像。并假设 s 与 w 不相关，w 为 0 均值的平稳随机过程。

图像的复原模型为：

$$\hat{s}(\mathrm{n}_1,\mathrm{n}_2)=\mathrm{h}(\mathrm{n}_1,\mathrm{n}_2)*x(\mathrm{n}_1,\mathrm{n}_2)=\sum_{l_1}\sum_{l_2}h(\mathrm{l}_1,\mathrm{l}_2)\times\mathrm{x}(\mathrm{n}_1-\mathrm{l}_1,\mathrm{n}_2-\mathrm{l}_2) \tag{2}$$

其中，$\hat{s}(\mathrm{n}_1,\mathrm{n}_2)$ 为恢复的图像，$\mathrm{h}(\mathrm{n}_1,\mathrm{n}_2)$ 为恢复滤波器。

误差度量为：

$$e^2=E\{(s(\mathrm{n}_1,\mathrm{n}_2)-\hat{s}(\mathrm{n}_1,\mathrm{n}_2))^2\} \tag{3}$$

基于正交性原理，若要求误差最小，则必有下式成立：

$$E\{\mathrm{e}(\mathrm{n}_1,\mathrm{n}_2)\times x^*(\mathrm{m}_1,\mathrm{m}_2)\}=0 \tag{4}$$

将（3）式带入（4）式有：

$$E\{\mathrm{s}(\mathrm{n}_1,\mathrm{n}_2)\times x^*(\mathrm{m}_1,\mathrm{m}_2)\}=E\{\hat{\mathrm{s}}(\mathrm{n}_1,\mathrm{n}_2)\times x^*(\mathrm{m}_1,\mathrm{m}_2)\} \tag{5}$$

即

（6）

换元得：

$$R_{sx}(\mathrm{n}_1,\mathrm{n}_2)=h(\mathrm{n}_1,\mathrm{n}_2)*R_{xx}(\mathrm{n}_1,\mathrm{n}_2) \tag{7}$$

等式两端同时取傅里叶变换得：

$$P_{sx}(w_1,w_2)=H(w_1,w_2)\times P_x(w_1,w_2) \tag{8}$$

即

$$H(w_1,w_2)=\frac{P_{sx}(w_1,w_2)}{P_x(w_1,w_2)} \tag{9}$$

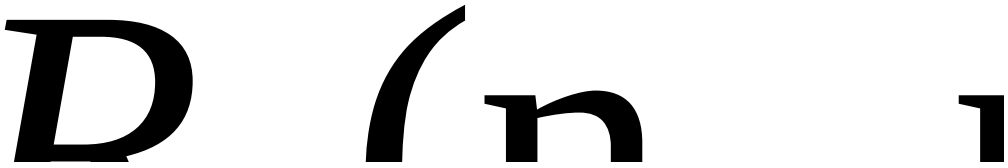
公式（8）中

（10）

公式（10）两端同时取傅里叶变换得：

$$P_{sx}(w_1,w_2)=B^*(w_1,w_2)\times P_s(w_1,w_2) \tag{11}$$

公式（8）中



（12）

公式（12）两端同时取傅里叶变换：

$$P_x(w_1,w_2)=\left|B(w_1,w_2)\right|^2\times P_s(w_1,w_2)+P_w(w_1,w_2) \tag{13}$$

将（11）式和（13）式带入（8）式得

$$H(w_1,w_2)=\frac{B^*(w_1,w_2)\times P_s(w_1,w_2)}{\left|B(w_1,w_2)\right|^2\times P_s(w_1,w_2)+P_w(w_1,w_2)} \tag{14}$$

将符号化成与书中一致的表示

（15）

故表达式由下式给出

（16）

2）约束最小二乘方滤波

对于约束最小二乘方滤波，期望是找一个最小准则函数 C，定义如下：

$$C=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[\nabla^2 \mathrm{f}(\mathrm{x},\mathrm{y})]^2 \tag{17}$$

其约束为

$$\left\|g-H\hat{f}\right\|^2=\|\eta\|^2 \tag{18}$$

其中，$\|w\|^2$ □ w^Tw 是欧几里得向量范数，$\hat{f}$ 是未退化图像的估计。

这个最佳问题在频率域中的解决由下面的表达式给出：

$$\hat{F}(\mathrm{u},\mathrm{v})=[\frac{H^*(\mathrm{u},\mathrm{v})}{|H(\mathrm{u},\mathrm{v})|^2+\gamma|P(\mathrm{u},\mathrm{v})|^2}]\mathrm{G}(\mathrm{u},\mathrm{v}) \tag{19}$$

其中，γ 是一个参数，必须对它进行调整以满足式（18）的条件，P(u,v)是函数

$$p(\mathrm{x},\mathrm{y})=\begin{bmatrix}0&-1&0\\-1&4&-1\\0&-1&0\end{bmatrix} \tag{20}$$

的傅里叶变换。

（2）MATLAB 函数：

1）imfilter

功能：对任意类型数组或多维图像进行滤波。

用法：B = imfilter(A,H)

B = imfilter(A,H,option1,option2,...)

或写做 g = imfilter(f, w, filtering_mode, boundary_options, size_options)

其中，f 为输入图像，w 为滤波掩模，g 为滤波后图像。filtering_mode 用于指定在滤波过程中是使用“相关”还是“卷积”。boundary_options 用于处理边界充零问题，边界的大小由滤波器的大小确定。具体参数选项见下表：

	选项	描述
filtering_mode	‘corr’	通过使用相关来完成，该值为默认。
	‘conv’	通过使用卷积来完成。
boundary_options	‘X’	输入图像的边界通过用值 X（无引号）来填充扩展，其默认值为 0。
	‘replicate’	图像大小通过复制外边界的值来扩展。
	‘symmetric’	图像大小通过镜像反射其边界来扩展。
	‘circular’	图像大小通过将图像看成是一个二维周期函数的一个周期来扩展
size_options	‘full’	输出图像的大小与被扩展图像的大小相同
	‘same’	输出图像的大小与输入图像的大小相同。这可通过将滤波掩模的中心点的偏移限制到原图像中包含的点来实现，该值为默认值。

2）fspecial

功能：fspecial 函数用于建立预定义的滤波算子。

用法：h = fspecial(type)

h = fspecial(type，para)

其中 type 指定算子的类型，para 指定相应的参数；

	选项	描述
type	'average'	averaging filter 为均值滤波，参数为 hsize 代表模板尺寸，默认值为[3 3]
	'disk'	circular averaging filter 为圆形区域均值滤波，参数为 radius 代表区域半径，默认值为 5.
	'gaussian'	Gaussian lowpass filter 为高斯低通滤波，有两个参数，hsize 表示模板尺寸，默认值为[3 3]，sigma 为滤波器的标准值，单位为像素，默认值为 0.5.
	'laplacian'	laplacian filter 为拉普拉斯算子，参数 alpha 用于控制算子形状，取值范围为[0，1]，默认值为 0.2。
	'log'	Laplacian of Gaussian filter 为拉普拉斯高斯算子，有两个参数，hsize 表示模板尺寸，默认值为[3 3]，sigma 为滤波器的标准差，单位为像素，默认值为 0.5.

	'motion'	motion filter 为运动模糊算子，有两个参数，表示摄像物体逆时针方向以 theta 角度运动了 len 个像素，len 的默认值为 9，theta 的默认值为 0；
	'prewitt'	Prewit thorizontal edge-emphasizing filter 用于边缘增强，大小为[3 3]，无参数
	'sobel'	Sobel horizontal edge-emphasizing filter 用于边缘提取，无参数
	'unsharp'	unsharp contrast enhancement filter 为对比度增强滤波器。参数 alpha 用于控制滤波器的形状，范围为[0，1]，默认值为 0.2.

（3）处理结果：

（a）实现模糊滤波器如方程 Eq. (5.6-11).

模糊滤波器的频域表达式为：

$$H(u,v)=\frac{T}{\pi(ua+vb)}\sin[\pi(ua+vb)]e^{-j\pi(ua+vb)}$$

故实现该滤波器，只需先将输入图像进行傅里叶变换并移至图像中心，之后将图像的傅里叶变换和模糊滤波器的傅里叶变换进行阵列相乘，将得到的结果经过傅里叶反变换返回到空间域即可实现该滤波器。具体程序见附件 mohu_filter.m

（b）模糊 lena 图像：45 度方向，T=1；（a=0.1，b=0.1；T=1）

lena.bmp 原始图像　　lena 运动模糊的结果（a=0.1，b=0.1；T=1）

lena.bmp 原始图像

lena 运动模糊的结果（调用 MATLAB 中的函数）

（c）再模糊的 lena 图像中增加高斯噪声，均值=0 ，方差=10 pixels 以产生模糊图像；

lena 运动模糊的结果（a=0.1，b=0.1；T=1）

添加高斯噪声的结果（均值为 0，方差为 0.01）

lena 运动模糊的结果（MATLAB 版）

添加高斯噪声的结果（MATLAB 版）

（d）分别利用方程 Eq. (5.8-6)和(5.9-4)，恢复图像。

维纳滤波的结果：

lena 运动模糊+高斯噪声.bmp

维纳滤波结果（K=0.06）

维纳滤波结果（K=0.01）

维纳滤波结果（K=0.5）

lena 运动模糊+高斯噪声（MATLAB 版）

维纳滤波结果（MATLAB 版）

约束最小二乘方滤波的结果：

lena 运动模糊+高斯噪声（MATLAB 版）

约束最小二乘滤波结果（MATLAB 版）

（3）结果分析及总结：

①首先分别通过自己编写的模糊函数和 MATLAB 中提供的 imfilter 和 fspecial 函数配合使用对图像 lena 进行了模糊滤波。发现套用书上的公式图像是斜向下 45 度运动模糊，而 MATLAB 中函数模糊的结果是斜向上 45 度运动模糊，不过这不是重点可以接受，模糊的基本效果还是一致的。之后调用 imnoise 函数对两幅图像加入高斯噪声，得到第二问的结果。

②之后分别使用自己编写的函数和 MATLAB 中提供的 deconvwnr 函数进行维纳滤波。调用 MATLAB 中函数滤波后的图像得到了一定的改善，运动模糊的影响基本被消除，但噪声的影响仍然较大，导致图像质量下降；对于自己编写的维纳滤波函数，难点在于寻找令信噪比最大的 K 值，报告中显示了部分 K 值对应的滤波结果，其中 K=0.06，为信噪比最大时的滤波结果，从结果看，视觉上的效果并不是很理想，要想达到更好的效果可能需要寻找更加合适的 K 值。

③最后采用 MATLAAB 提供的 deconvreg 函数进行约束最小二乘方滤波。从滤波后的结果看，约束最小二乘方滤波得到了比维纳滤波更好的结果，尤其是对噪声的滤除。

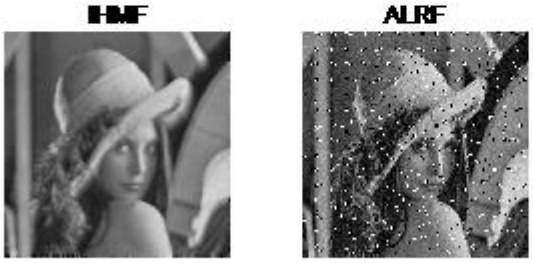
3.在测试图像 lena 图加入椒盐噪声（椒和盐噪声密度均是 0.1）；用学过的滤波器恢复图像；再使用反谐波分析 Q 大于 0 和小于 0 的作用；**

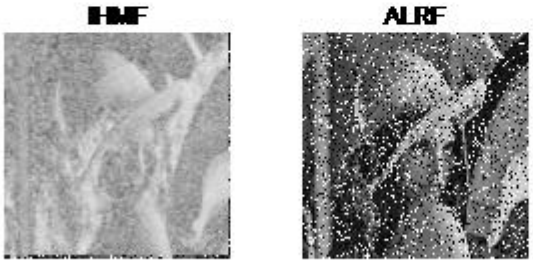
实现思路：通过 imnoise 函数添加椒盐噪声，选用中值滤波器、高斯空域滤波器、高斯频域滤波器对图像进行恢复对比，通过课本上的实例，添加 imfilt、charmean、tofloat 函数，建立逆谐波滤波器并对图像进行滤波。

1. 实验原理

椒盐噪声是指图像中的一些分散的黑点和白点。

这个时候就可以用中值滤波器进行滤波，当取中值时，该区域中的全为 1 或 0 的点都会被去除 。





3. 结果总结

因为椒盐噪声会造成很多很多的零值，这些零值会造成其他像几何均值滤波器等利用乘法的滤波器产生无穷大的结果。

分析

综合对比可看出，中值滤波器效果最好，完全消除了椒盐噪声同时基本保留了原图像的细节，空域高斯滤波效果稍好于频域高斯滤波；

对逆谐波滤波器，Q 决定的是对椒噪声和盐噪声的选择过滤程度，当 Q 大于 0 时，过滤椒噪声，当 Q 小于 0 时，过滤盐噪声。

附录

【参考文献】

[1] 冈萨雷斯.数字图像处理（第三版）北京：电子工业出版社，2011

[2] 周品.MATLAB 数字图像处理 北京：清华大学出版社，2012

[3] 杨杰.数字图像处理及 MATLAB 实现 北京：电子工业出版社，2010

【源代码】

add_gaussian_noise.m(题目 1 添加高斯噪声)

```
I=imread('lena.bmp');
figure(1);
imshow(I);
title('源图像 lena.bmp');
imwrite(I,'lena 原始图像.bmp');
I2=imnoise(I,'gaussian',0.5,0.01);
figure(2);
imshow(I2);
title('加入 gaussian 噪声后的 lena.bmp');
imwrite(I2,'lena 加入 gaussian 噪声后(u=0.5，s^2=0.01).bmp');
```

suanshujunzhi_filter.m(题目 1 算术均值滤波恢复图像)

```
I=imread('lena_gaussian_noise.bmp');
figure(1);
imshow(I);
title('lena 加入 gaussian 噪声后(u=0，s^2=0.01).bmp');
imwrite(I,'lena 加入 gaussian 噪声后(u=0，s^2=0.01).bmp');
n=5;
```

```
h=1/n^2.*ones(n,n);
I2=conv2(I,h,'same');
I2=uint8(I2);
figure(2);
imshow(I2);
title('算术均值滤波的结果（5x5）');
imwrite(I2,'算术均值滤波的结果（5x5）.bmp');
```

median_filter.m(题目 1 中值滤波恢复图像)

```
I=imread('lena_gaussian_noise.bmp');
figure(1);
imshow(I);
title('lena 加入 gaussian 噪声后(u=0，s^2=0.01).bmp');
imwrite(I,'lena 加入 gaussian 噪声后(u=0，s^2=0.01).bmp');
figure(2);
n=5;
a=ones(n,n);
p=size(I);
x1=double(I);
x2=x1;
for i=1:p(1)-n+1
    for j=1:p(2)-n+1
        c=x1(i:i+(n-1),j:j+(n-1));
        e=c(1,:);
        for u=2:n
            e=[e,c(u,:)];
        end
        mm=median(e);
        x2(i+(n-1)/2,j+(n-1)/2)=mm;
    end
end
I2=uint8(x2);
imshow(I2);
title('中值滤波的结果（5x5）');
imwrite(I2,'中值滤波的结果（5x5）.bmp');
```

mohu_filter.m(题目 2 运动模糊滤波器)

```
I=imread('lena.bmp');
figure(1);
imshow(I);
title('lena.bmp 原始图像');
imwrite(I,'lena 原始图像.bmp');
f=double(I);
F=fft2(f);
```

```
F=fftshift(F);
[M,N]=size(F);
a=0.1;b=0.1;T=1;
for u=1:M
    for v=1:N
H(u,v)=(T/(pi*(u*a+v*b)))*sin(pi*(u*a+v*b))*exp(-sqrt(-1)*pi*(u*a+v*b
));
     G(u,v)=H(u,v)*F(u,v);
    end
end
G=ifftshift(G);
g=ifft2(G);
g=256.*g./max(max(g));
g=uint8(real(g));
figure(2);
imshow(g);
title('运动模糊化 lena.bmp');
imwrite(g,'lena 运动模糊的结果.bmp');
```

Wiener_filter.m(题目 2 维纳滤波器自编版)

```
I=imread('lena 运动模糊+高斯噪声.bmp');
figure(1);
imshow(I);
title('lena 运动模糊+高斯噪声');
imwrite(I,'lena 运动模糊+高斯噪声.bmp');
g=double(I);
G=fft2(g);
G=fftshift(G);
[M,N]=size(G);
a=0.1;b=0.1;T=1;K=0.0259;
for u=1:M
    for v=1:N
H(u,v)=(T/(pi*(u*a+v*b)))*sin(pi*(u*a+v*b))*exp(-sqrt(-1)*pi*(u*a+v*b
));
     F(u,v)=1/H(u,v)*(abs(H(u,v)))^2/((abs(H(u,v)))^2+K)*G(u,v);
    end
end
F=ifftshift(F);
f=ifft2(F);
f=256.*f./max(max(f));
f=uint8(real(f));
figure(2);
imshow(f);
title('维纳滤波的将结果');
```

```
imwrite(f,'维纳滤波的结果(K=0.0259).bmp');
```

wiener_filter_k.m(题目 2 维纳滤波器寻找信噪比最大的 K 值)

```
I=imread('lena 运动模糊+高斯噪声.bmp');
figure(1);
imshow(I);
title('lena 运动模糊+高斯噪声');
imwrite(I,'lena 运动模糊+高斯噪声.bmp');
g=double(I);
G=fft2(g);
G=fftshift(G);
[M,N]=size(G);
a=0.1;b=0.1;T=1;i=1;
format long
for k=0.01:0.01:0.11
    for u=1:M
        for v=1:N
H(u,v)=(T/(pi*(u*a+v*b)))*sin(pi*(u*a+v*b))*exp(-sqrt(-1)*pi*(u*a+v*b
));
F(u,v)=(1/H(u,v))*((abs(H(u,v)))^2/((abs(H(u,v)))^2+k))*G(u,v);
        end
    end
    F=ifftshift(F);
    f=ifft2(F);
    f=256.*f./max(max(f));
    f=uint8(real(f));
    figure;
    imshow(f);
    title('维纳滤波的结果');
    e=f-uint8(g);
    SNR(i)=sum(sum(g.^2))/sum(sum(e.^2));
    i=i+1;
end
idx=find(max(SNR))
```

wiener_filter_matlab.m(题目 2 维纳滤波器 MATLAB 版)

```
I=imread('lena.bmp');
H=fspecial('motion',50,45);
I1=imfilter(I,H,'circular','conv');
figure(1);
imshow(I1);
title('运动模糊后的 lena.bmp(角度为 45 度)');
imwrite(I1,'lena 运动模糊（调用 matlab 中的函数）.bmp');
I2=imnoise(I1,'gaussian',0,0.01);
```

```
figure(2)
imshow(I2);
title('加噪并模糊的 lena.bmp');
imwrite(I2,'lena 运动模糊+高斯噪声（调用 matlab 中的函数 0.bmp');
figure(3);
noise=imnoise(zeros(size(I)),'gaussian',0,0.01);
NSR=sum(noise(:).^2)/sum(im2double(I(:)).^2);
I3=deconvwnr(I2,H,NSR);
imshow(I3);
title('维纳滤波的结果');
imwrite(I3,'lena 维纳滤波的结果（调用 MATLAB 中的函数）.bmp');
```

yueshuzuixiaercheng_filter_matlab.m（题目 2 约束最小二乘滤波（MATLAB 版））

```
I=imread('lena.bmp');
h=fspecial('motion',50,45);
I1=imfilter(I,h,'circular','conv');
I2=imnoise(I1,'gaussian',0,0.01);
figure(1);
imshow(I2);
title('lena 运动模糊+高斯噪声');
imwrite(I2,'lena 运动模糊+高斯噪声（MATLAB 版）.bmp');
V=0.0001;
NoisePower=V*prod(size(I));
[g,LAGRA]=deconvreg(I1,h,NoisePower);
figure(2);
imshow(g);
title('约束最小二乘滤波的结果（MATLAB 版）');
imwrite(g,'约束最小二乘滤波的结果（MATLAB 版）.bmp');
```