1.1 我们假设傅里叶正变换前常数项为 1，即 F(u,v) = $\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y) e^{-j2pi(ux/M+vy/N)}$，此时，

F(0,0) = $\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)$；可以看出零频率项与 f(x,y)的平均值成正比，即

F(0,0) = MN$\overline{f}$ (x,y)。所以，通过函数接口以及输入图像矩阵，我们可以直接计算出 MN、$\overline{f}$ (x,y)以及 F(0,0)的值。如果满足 F(0,0) = MN$\overline{f}$ (x,y)，说明假设成立，即傅里叶正变换前常数项为 1，相对的，反变换前常数项为$\frac{1}{MN}$。如果满足 F(0,0) = $\overline{f}$ (x,y)，说明傅里叶正变换前常数项为$\frac{1}{MN}$，反变换常数项为 1。如果满足 F(0,0) = $\sqrt{MN}\,\overline{f}$ (x,y)，说明傅里叶正变换和反变换前常数项都为$\frac{1}{\sqrt{MN}}$。

1.2 一样。根据二维 DFT 的平移特性有

f(x-x0,y-y0)$\Leftrightarrow$F(u,v)$e^{-j2pi(x0u/M+y0v/N)}$;

我们可以假设图 b 是 f(x,y),由于图 b 与图 c 仅仅是补 0 位置不同，所以图 c 可以看作是由 f(x,y)平移得到的，为 f(x-x0,y-y0)。根据上述公式，图 c 的傅里叶谱为

|F(u,v)$e^{-j2pi(x0u/M+y0v/N)}$|=|F(u,v)|*|$e^{-j2pi(x0u/M+y0v/N)}$|;

根据欧拉公式，|$e^{-j2pi(x0u/M+y0v/N)}$| = 1,所以|F(u,v)$e^{-j2pi(x0u/M+y0v/N)}$|=|F(u,v)|。

故，图 b、图 c 的傅里叶谱相同。

1.3

1. g(x,y) = 1/4[f(x,y+1)+f(x+1,y)+f(x-1,y)+f(x,y-1)].

   G(u,v) = 1/4[$e^{j2pi*v/N}+e^{j2pi*u/M}+e^{-j2pi*u/M}+e^{-j2pi*v/N}$] * F(u,v)

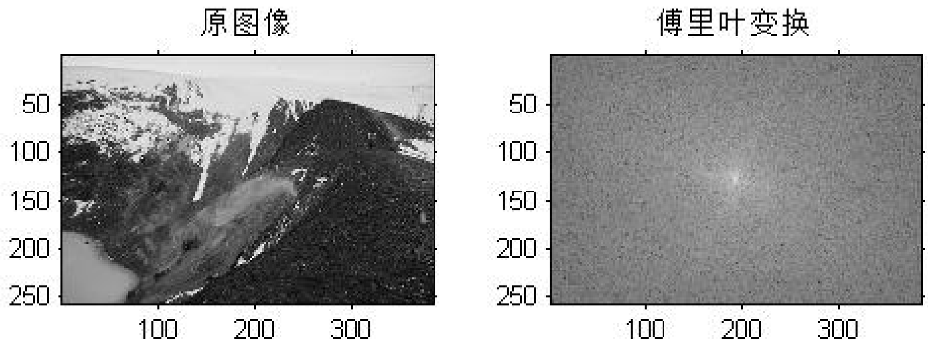
   = H(u,v) * F(u,v)

   H(u,v) = 1/2[cos(2pi u/M)+cos(2pi v/N)] 为频率域滤波器。

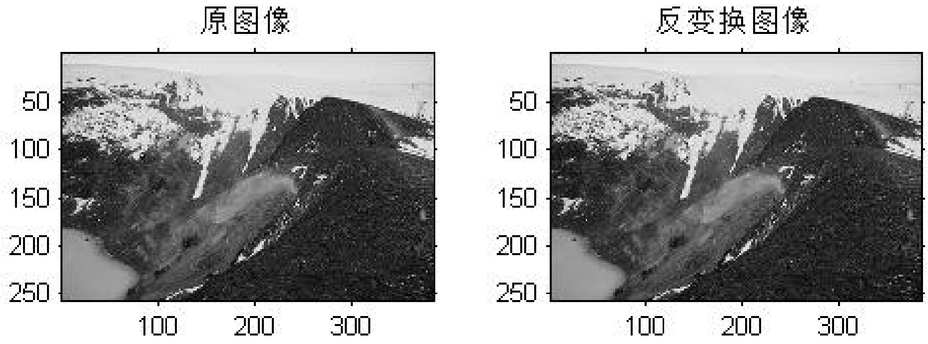
2. 低通滤波器。将 H(u,v)平移到频率矩形的中心(M/2,N/2),
   H(u,v) = 1/2[cos(2pi (u-M/2)/M)+cos(2pi (v-N/2)/N)].
   先仅考虑 u 变量，当 u 的值从 0 增加到 M-1 时，H(u,v)的值从-1 增加到 1 再减小为-1。在 u=0 和 M 处，H(u,v) = -1。在 u=M/2 处，H(u,v) = 1。1 为最大值。该滤波器的振幅在滤波器中心处取，越远离中心，H(u,v)取值越小。这种特性与低通滤波器一样。这种情况对 v 变量也一样。所以，该滤波器为低通滤波器。

2.2

1.这里，flag 取 0 得到傅里叶正变换频谱图，flag 取 1 进行反傅里叶变换。



2.



3. 算法实现：

傅里叶正变换和反变换都是采用一样的方法，就是利用二维 DFT 的可分性，将二维变换转为一维变换。按照书本所给公式，实现两个三重循环即可。公式如下：

$$F(u,v) = \sum_{x=0}^{M-1} e^{-j2pi*ux/M} \sum_{y=0}^{N-1} f(x,y)e^{-j2pi*vy/N} = \sum_{x=0}^{M-1} F(x,v)e^{-j2pi*ux/M},$$

其中，$F(x,v) = \sum_{y=0}^{N-1} f(x,y)e^{-j2pi*vy/N}$。

代码实现如下：

傅里叶正变换：

```
%傅里叶变换，采用的是将二维 DFT 转为两个一维 DFT
%变换的第一部分，temp2 为中间变量矩阵
for x=1:M
    for v=1:N
        for y=1:N
            s = s + temp1(x,y) * exp((-1i) * 2 * pi * v * y / N);
        end
        temp2(x,v) = s;
        s = 0;
    end
end
%变换的第二部分，temp 为未进行中心化的生成矩阵，temp3 为中心化后并通过傅里叶变换产生的频谱图
```

```
for u=1:M
    for v=1:N
        for x=1:M
            s = s + temp2(x,v) * exp((-1i) * 2 * pi * u * x / M);
        end
        temp(u,v) = s;
        temp3(u,v) = log(abs(s));
        s = 0;
    end
End
```

傅里叶反变换：

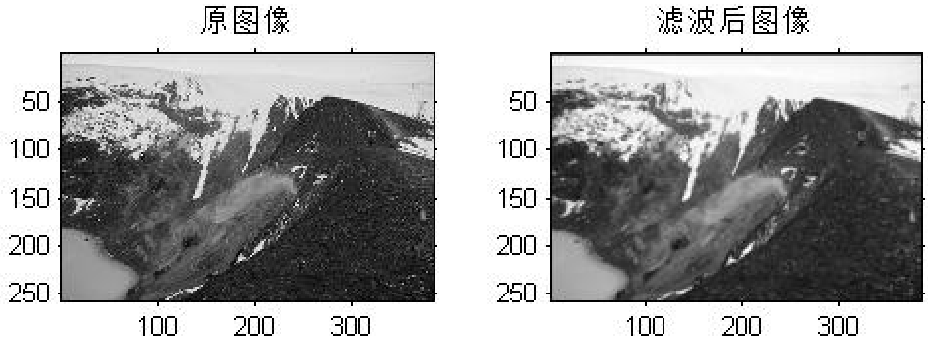
```
%作傅里叶反变换，方法跟傅里叶正变换一样，都是采用二维转一维
if (flag == 1)
for u=1:M
    for y=1:N
        for v=1:N
            s = s + temp(u,v) * exp((1i) * 2 * pi * v * y / N);
        end
        temp4(u,y) = s;
        s = 0;
    end
end

for x=1:M
    for y=1:N
        for u=1:M
            s = s + temp4(u,y) * exp((1i) * 2 * pi * u * x / M) / M;
        end
        temp5(x,y) = real(s); %temp5 为反变换后的矩阵
        s = 0;
    end
end
```

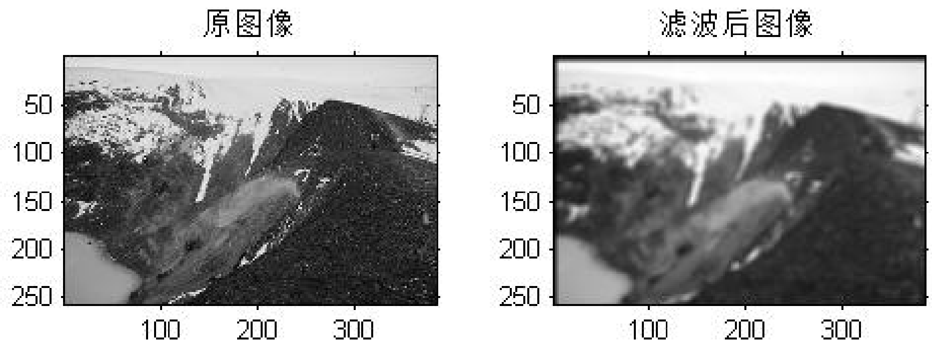
2.4

1.

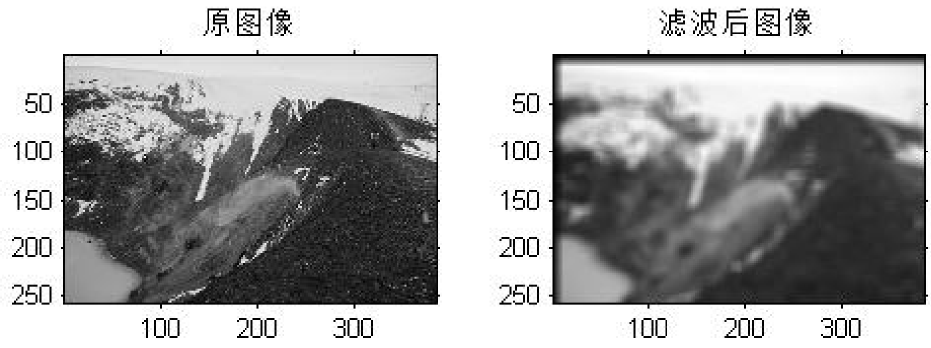
3*3 滤波结果


原图像
滤波后图像
50
100
150
200
250
100
200
300

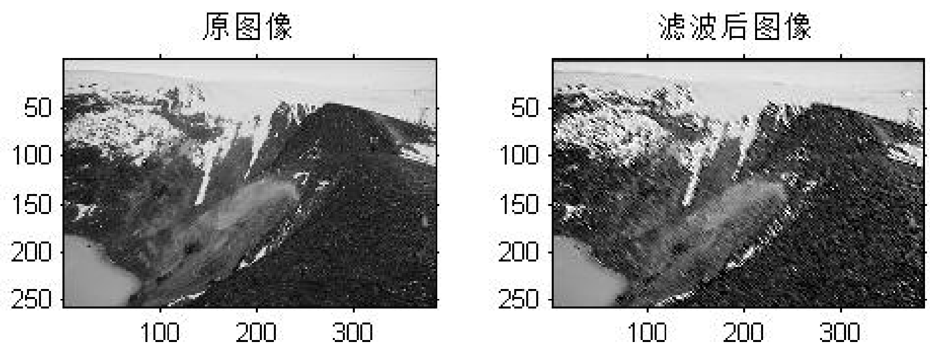
7*7 滤波结果


原图像
滤波后图像
50
100
150
200
250
100
200
300

11*11 滤波结果



2.拉普拉斯算子滤波后结果：我选择的是[1 1 1;1 -8 1;1 1 1]算子。



3.算法说明：

我是根据书本 4.7.3 节内容按照它所描述的步骤一步步实现频率域滤波算法的。具体步骤如下：

1. 选择填充参数 P=2M，Q=2N；
2. 对输入图像和滤波器补 0，生成 P*Q 大小的图像矩阵和滤波器；
3. 中心化，用 $(-1)^{x+y}$ 乘以图像矩阵以及滤波器移到其变换的中心；
4. 计算图像跟滤波器的 DFT，生成 F(u,v)和 H(u,v)；
5. 用阵列相乘形成乘积 G(u,v) = H(u,v) * F(u,v);
6. 作傅里叶反变换，公式为：

   g(x,y) = {real[3-1[G(u,v)]]} * $(-1)^{x+y}$

7. 从 g(x,y)的左上限提取 M*N 区域，得到最终处理结果 g(x,y)。

相关代码：

矩阵补 0：

```
% 0 填充图像矩阵和滤波器
P = 2*M;
Q = 2*N;
f = zeros(P,Q);
h = zeros(P,Q);
f(1:M,1:N) = img;
f(M+1:P,1:Q) = 0;
f(1:M,N+1:Q) = 0;
h(1:m,1:n) = filter;
h(m+1:P,1:Q) = 0;
h(1:m,n+1:Q) = 0;
```

中心化，傅里叶正变换和反变换跟上一个代码基本类似，就不再附上；

频率域乘积：

```
%作频率域乘积
for u=1:P
    for v=1:Q
        G(u,v) = H(u,v) * F(u,v);
    end
end
```

截取 M*N 大小图像作为输出

```
%截取 M*N 大小矩阵
for x=1:M
    for y=1:N
        g(x,y) = g2(x,y);
    end
end
```