1.问题描述：编写一个图像直方图均衡化程序，g=histequal4e(I)，其中 I 是 8 比特图像

对应代码：histequal.py 文件

整体思路方法及代码编写：

8 比特图像灰度级的数量为 256

Imhist 求其概率值

imhist.cumsum()为累加

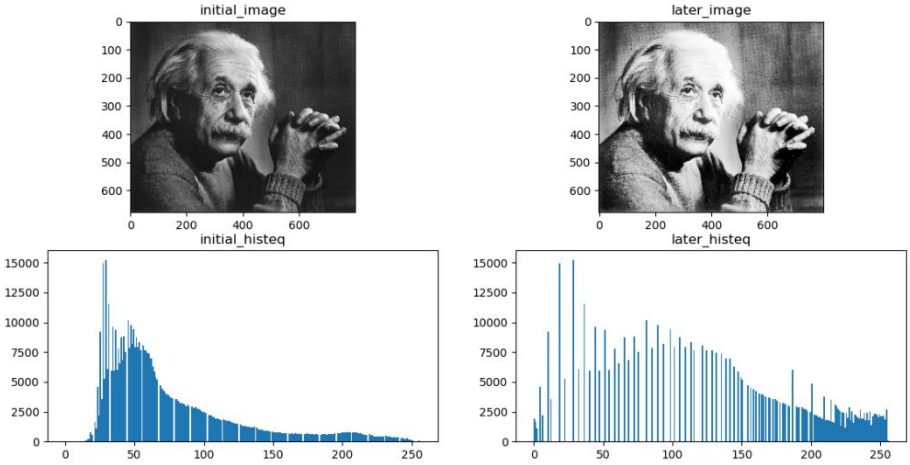
再乘以 255 得到像素值

通过 interp 把新得到的值移到原始值上

plt.hist(f.ravel())显示直方图，ravel 是把多维降维一维

结果展示：

明显看到图片的直方图均衡化了很多。



2.问题描述：编写一个程序完成如下功能：读入清晰图像，加上椒盐噪声，采用有选择保边缘平滑法对图像进行平滑。

对应代码：smothness.py 文件

整体思路方法：（1）读入图像，为图像添加椒盐噪声，即对某些像素值随机赋值为 0 或 255。一共添加 proportion=0.05*总数个椒盐噪声的像素点，对这些点选择随机生成 0 或 1.若是 0 则赋值为 0，若是 1 则赋值为 255。

（2）有选择保边缘平滑法。首先 padding，两种方式：选择补 0 或者复制边界像素值。其次写出 9 种掩模，分别与每个像素的 5*5 邻域相乘，计算方差和均值，注意这里计算方差，不应该算上掩模中值为 0 的地方，算均值也不应该算上，不能直接用 var 和 mean。最小方差对应的灰度均值为像素输出值。

代码编写：

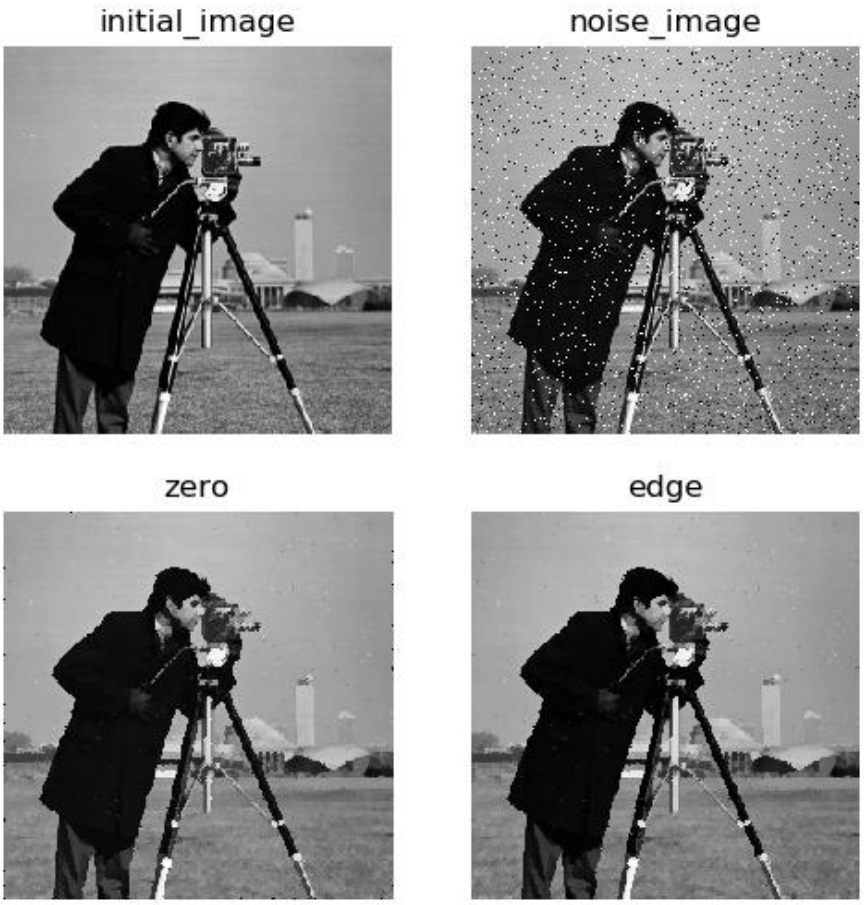
- ✧ 一共有 num 个添加椒盐噪声的像素点 num = int(height*width*proportion)
- ✧ 选择随机生成 0or1 并赋相应的值 random.randint(0,1)
- ✧ 噪声图 padding 方式 1：补 0 mode='constant', constant_values=0
- ✧ padding 方式 2：edge

- ✧ np.array9 个掩模算子
- ✧ 滑动算子与图片像素的每 5 个邻域分别相乘 w = ff[i-2:i+3,j-2:j+3]*a[k]
- ✧ 找出最小方差对应的索引 np.where(U==np.min(U))
- ✧ 该索引对应的算子再次滑动图片求得均值 np.mean

结果展示：

左上为原图，右上为噪声图片。左下为补 0 后的有选择保边缘平滑法，右下为复制 edge 值的有选择保边缘平滑法。

两种进行对比，感觉除了 zero 的边缘有些不平整，以及图片边缘得到的不太一样，其他的内容没有任何区别，有选择保边缘平滑法恢复的图片很好。



3. 问题描述：编写一个程序完成拉普拉斯增强。

对应代码：Laplacian.py 文件

整体思路方法：

$$\nabla^2 f = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y)$$

拉普拉斯算子还可以表示成模板的形式

0	1	0
1	-4	1
0	1	0
Laplace算子		

0	-1	0
-1	5	-1
0	-1	0
增强算子		

本题采用增强算子：np.array([[0,-1,0], [-1,5,-1], [0,-1,0]])
图像先 padding，与题 2 一样两种方式
其次用增强算子与其卷积
(ff[i - 1:i + 2, j - 1:j + 2] * a).sum().astype('uint8')
结果展示：