**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

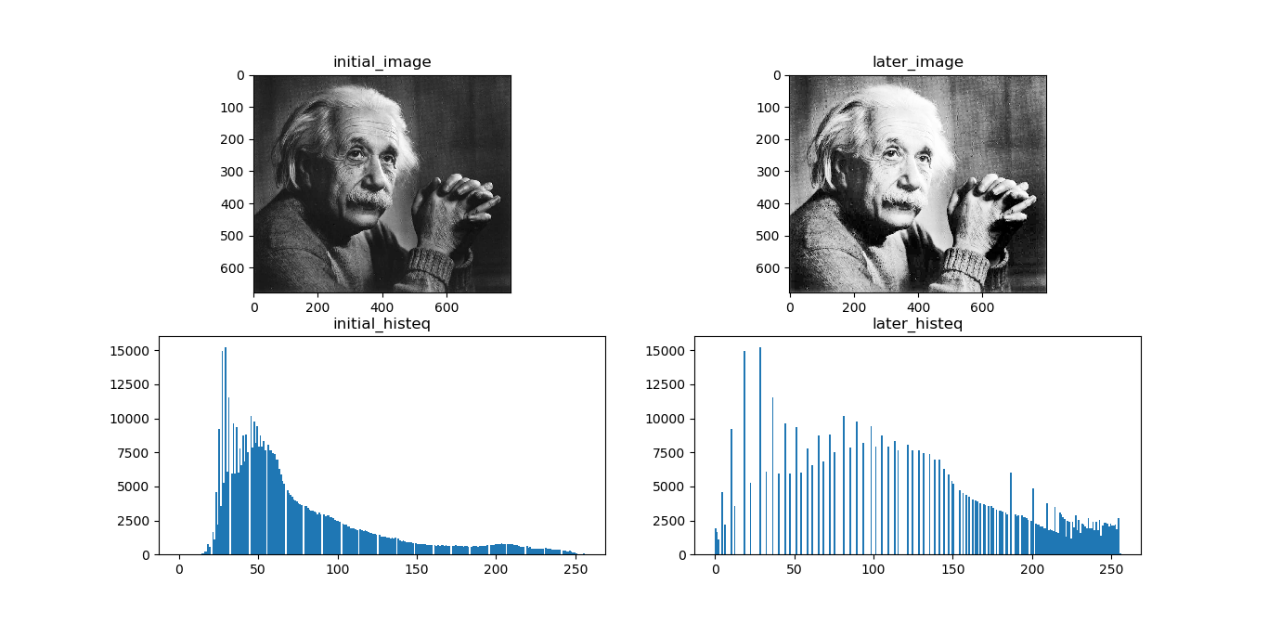
再乘以255得到像素值

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。



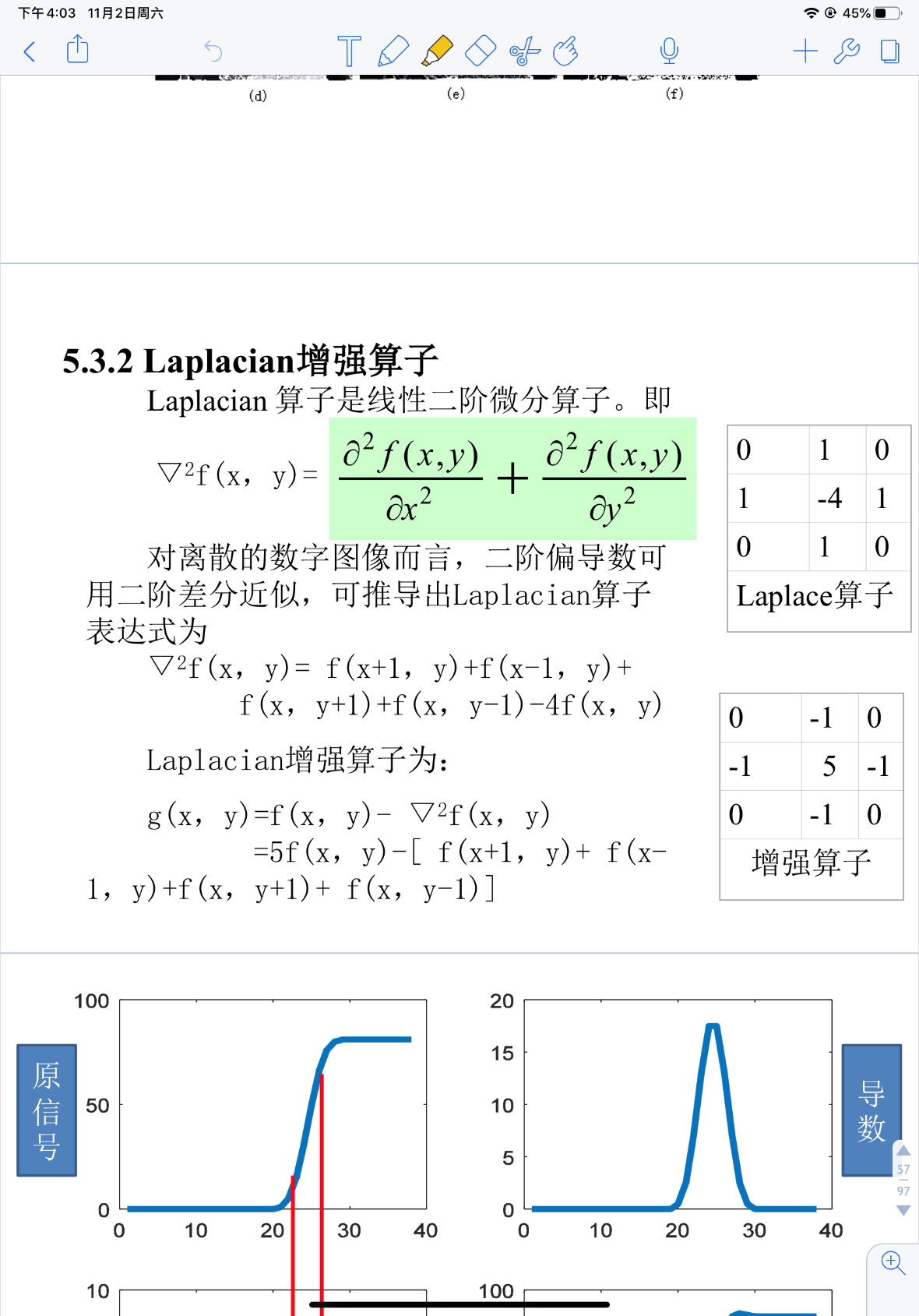
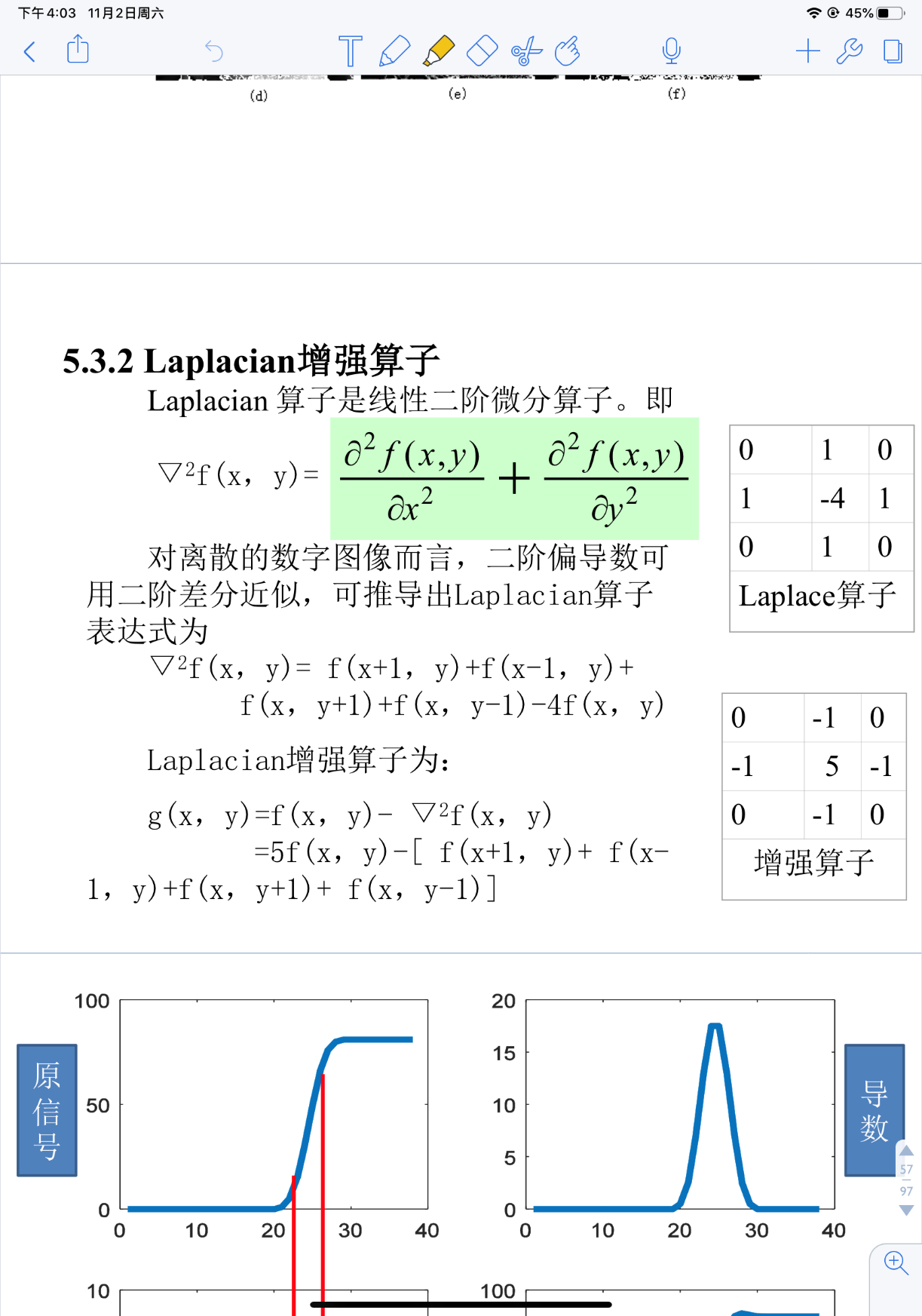
**3.问题描述：**编写一个程序完成拉普拉斯增强。

**对应代码：**Laplacian.py文件

**整体思路方法：**

http://www.itdaan.com/imgs/8/8/9/7/12/36d7deec6a0e1256c74cbe16a9ff81ac.jpe

拉普拉斯算子还可以表示成模板的形式



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：**

****