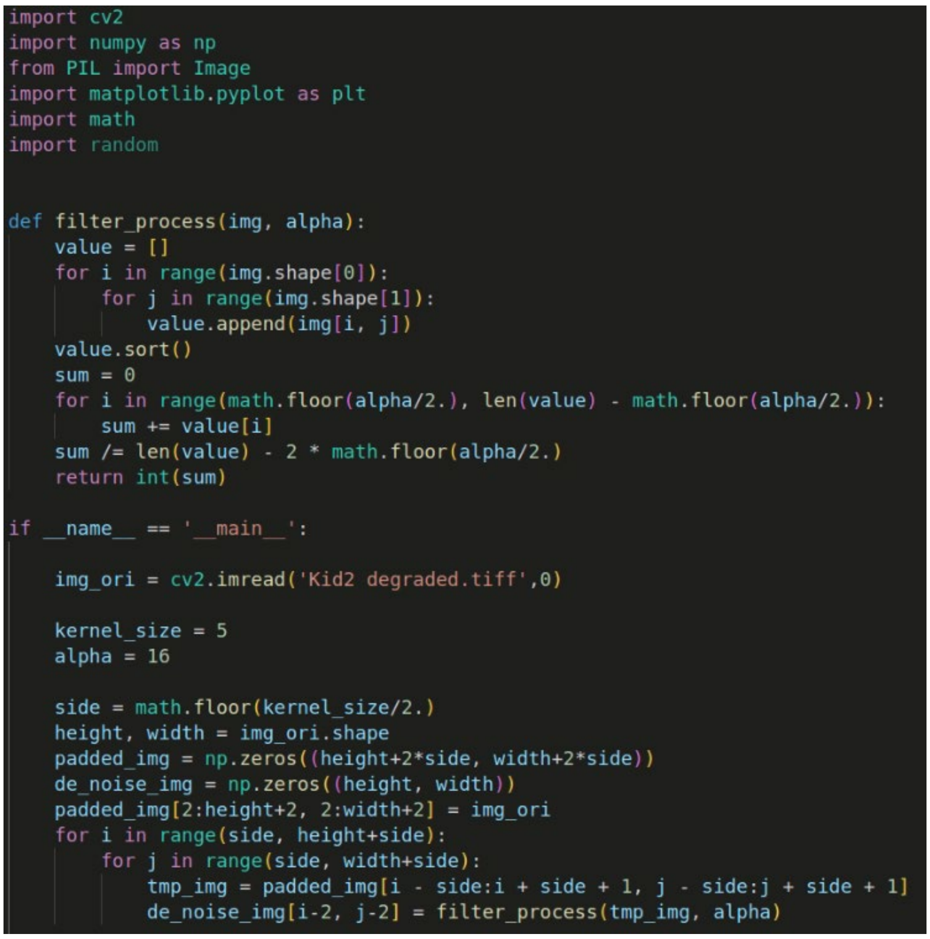
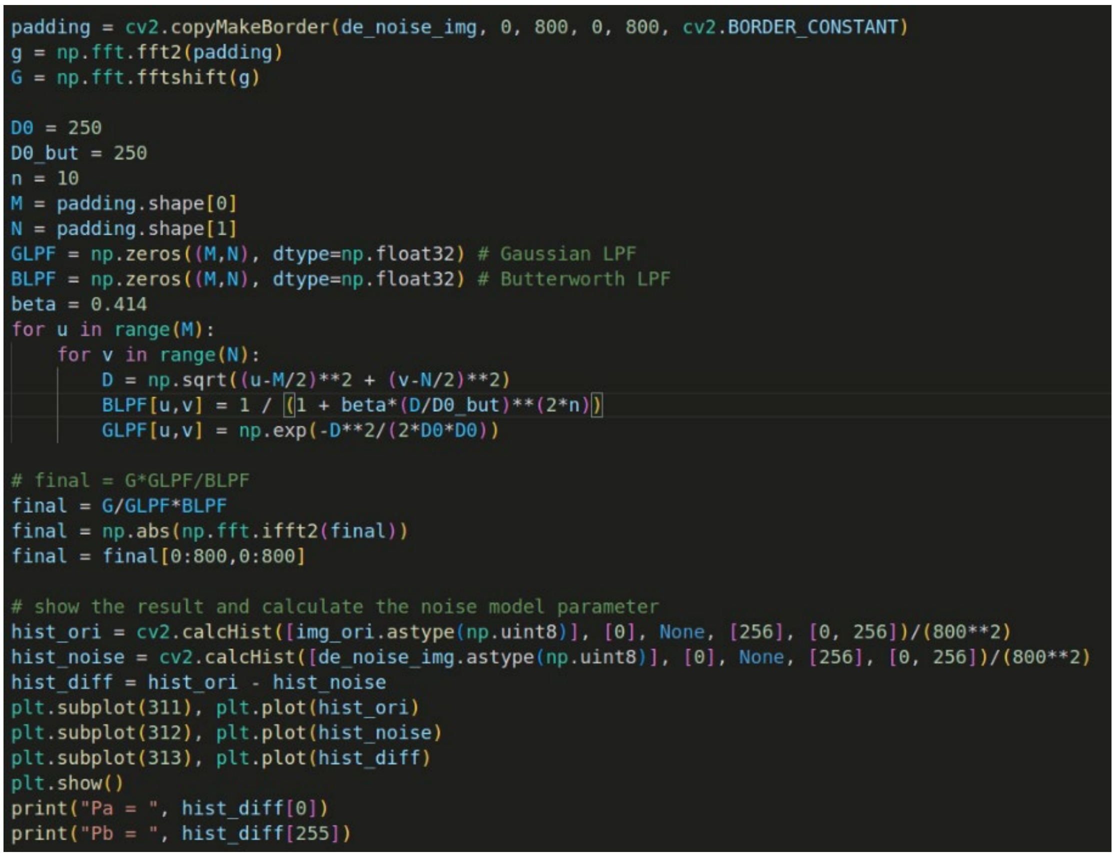
Project 2

ID：311512015　　　name：謝元碩

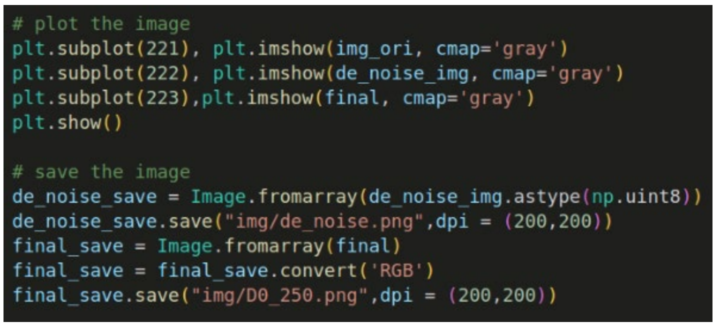
(a) Source codes:



```python
import cv2
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import math
import random


def filter_process(img, alpha):
    value = []
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            value.append(img[i, j])
    value.sort()
    sum = 0
    for i in range(math.floor(alpha/2.), len(value) - math.floor(alpha/2.)):
        sum += value[i]
    sum /= len(value) - 2 * math.floor(alpha/2.)
    return int(sum)

if __name__ == '__main__':

    img_ori = cv2.imread('Kid2 degraded.tiff',0)

    kernel_size = 5
    alpha = 16

    side = math.floor(kernel_size/2.)
    height, width = img_ori.shape
    padded_img = np.zeros((height+2*side, width+2*side))
    de_noise_img = np.zeros((height, width))
    padded_img[2:height+2, 2:width+2] = img_ori
    for i in range(side, height+side):
        for j in range(side, width+side):
            tmp_img = padded_img[i - side:i + side + 1, j - side:j + side + 1]
            de_noise_img[i-2, j-2] = filter_process(tmp_img, alpha)
```

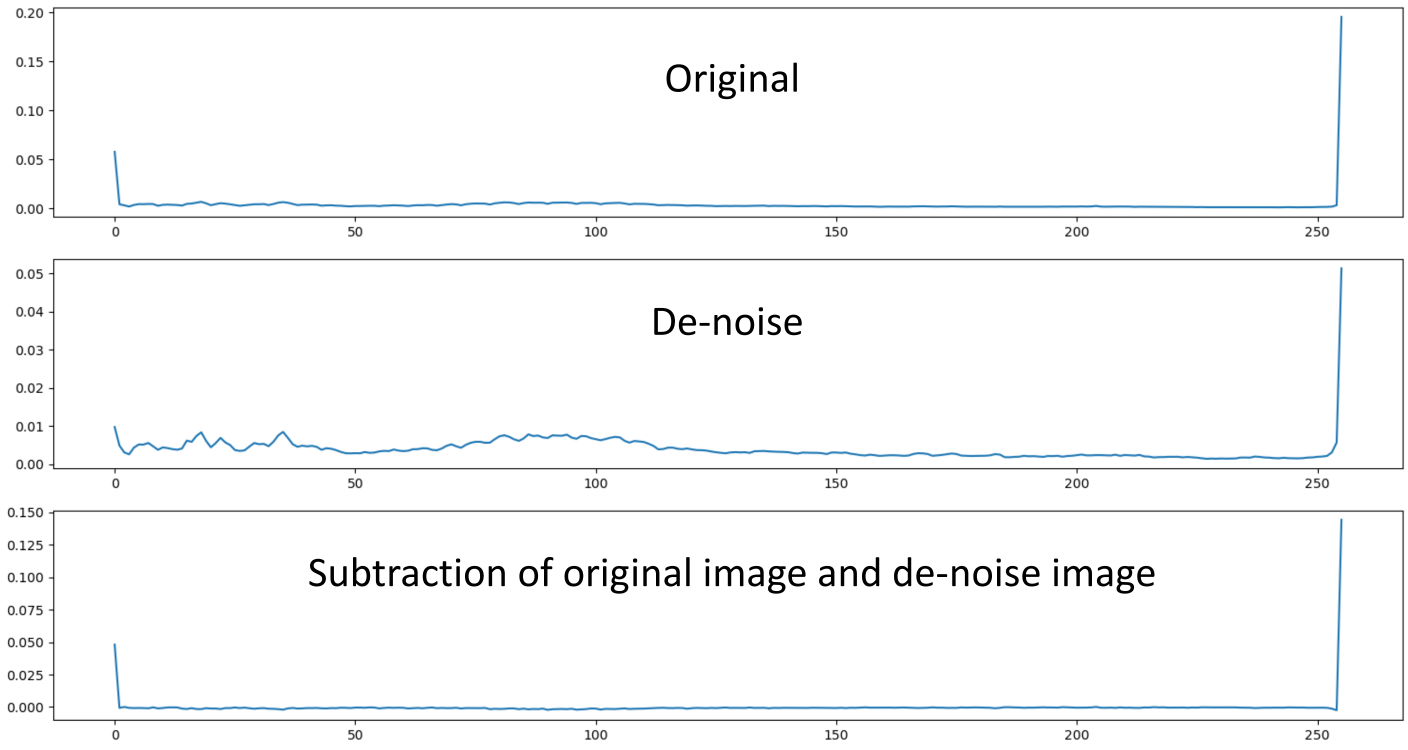


```python
padding = cv2.copyMakeBorder(de_noise_img, 0, 800, 0, 800, cv2.BORDER_CONSTANT)
g = np.fft.fft2(padding)
G = np.fft.fftshift(g)

D0 = 250
D0_but = 250
n = 10
M = padding.shape[0]
N = padding.shape[1]
GLPF = np.zeros((M,N), dtype=np.float32) # Gaussian LPF
BLPF = np.zeros((M,N), dtype=np.float32) # Butterworth LPF
beta = 0.414
for u in range(M):
    for v in range(N):
        D = np.sqrt((u-M/2)**2 + (v-N/2)**2)
        BLPF[u,v] = 1 / (1 + beta*(D/D0_but)**(2*n))
        GLPF[u,v] = np.exp(-D**2/(2*D0*D0))

# final = G*GLPF/BLPF
final = G/GLPF*BLPF
final = np.abs(np.fft.ifft2(final))
final = final[0:800,0:800]

# show the result and calculate the noise model parameter
hist_ori = cv2.calcHist([img_ori.astype(np.uint8)], [0], None, [256], [0, 256])/(800**2)
hist_noise = cv2.calcHist([de_noise_img.astype(np.uint8)], [0], None, [256], [0, 256])/(800**2)
hist_diff = hist_ori - hist_noise
plt.subplot(311), plt.plot(hist_ori)
plt.subplot(312), plt.plot(hist_noise)
plt.subplot(313), plt.plot(hist_diff)
plt.show()
print("Pa = ", hist_diff[0])
print("Pb = ", hist_diff[255])
```

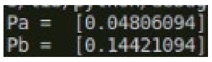


```python
# plot the image
plt.subplot(221), plt.imshow(img_ori, cmap='gray')
plt.subplot(222), plt.imshow(de_noise_img, cmap='gray')
plt.subplot(223),plt.imshow(final, cmap='gray')
plt.show()

# save the image
de_noise_save = Image.fromarray(de_noise_img.astype(np.uint8))
de_noise_save.save("img/de_noise.png",dpi = (200,200))
final_save = Image.fromarray(final)
final_save = final_save.convert('RGB')
final_save.save("img/D0_250.png",dpi = (200,200))
```

(b) Results of noise model and model parameters:

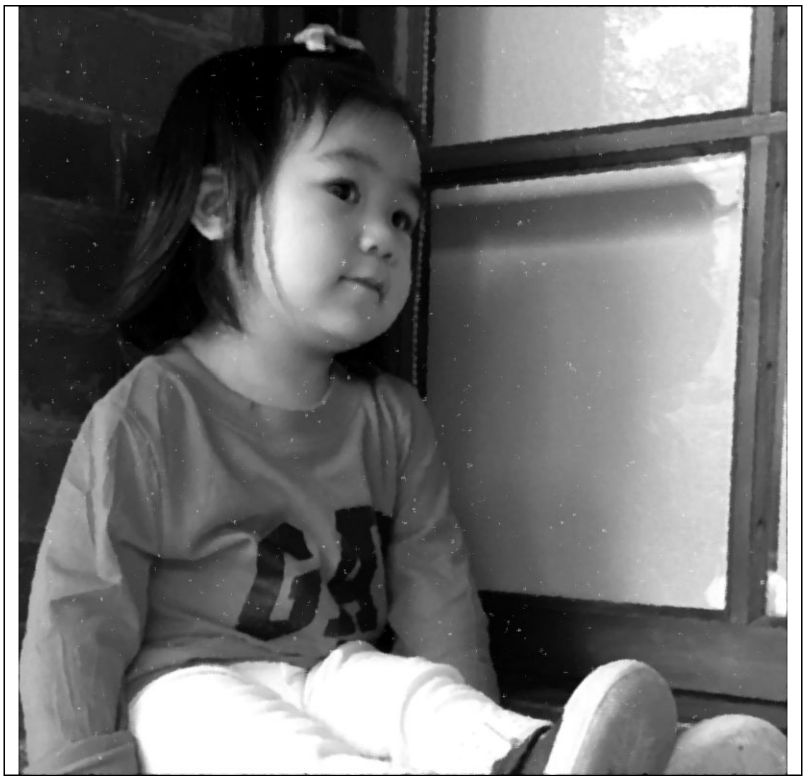


- Compare the original and de-noise histograms, and calculate the probability difference between them to find Pa, Pb. The results show that there are more different in the salt part.

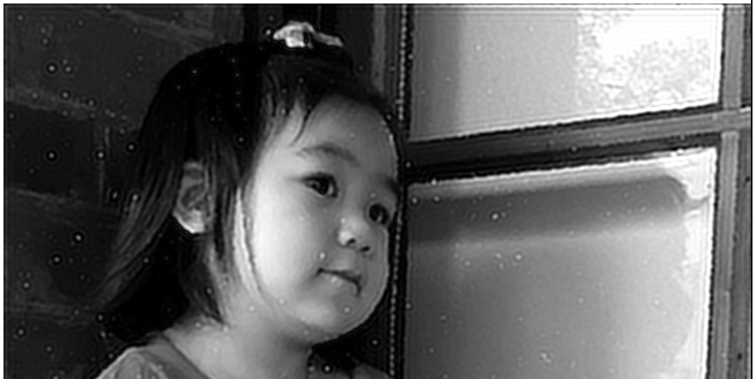


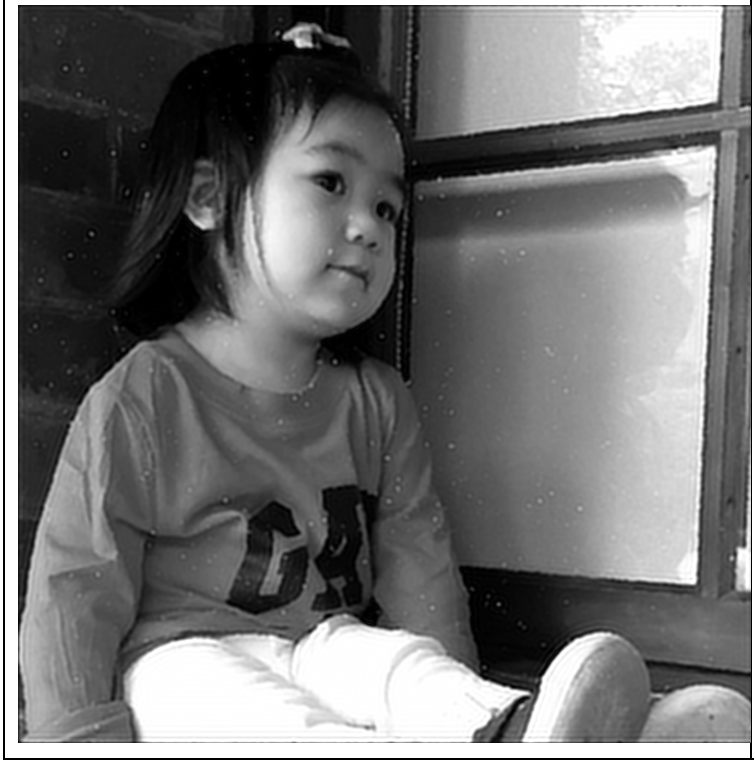
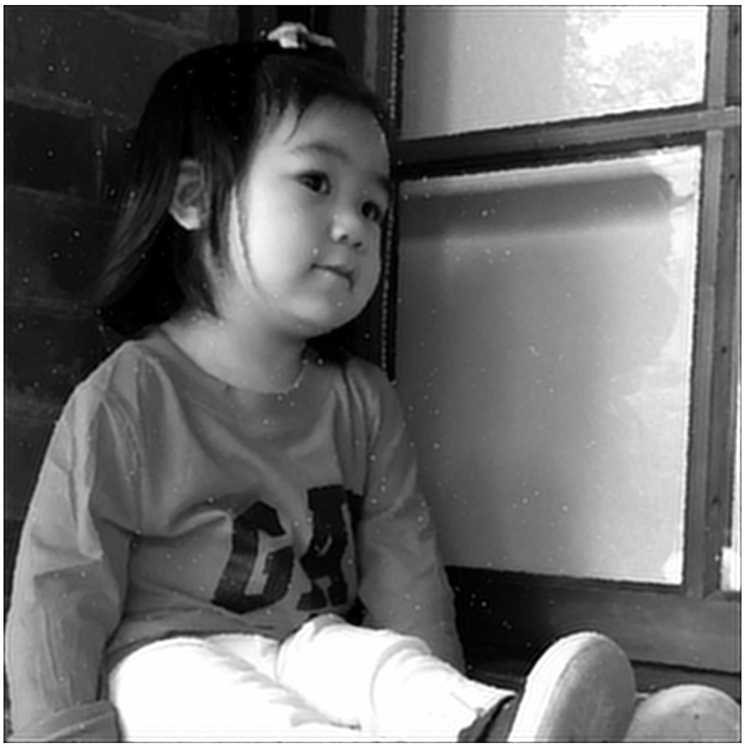
```
Pa =  [0.04806094]
Pb =  [0.14421094]
```

(c) De-noised image by alpha-trimmed mean filter:



(d) Output image、parameters:

D0 = 100 (因處理後為白色，故圖片不明顯)	D0 = 150
	

D0 = 200	D0 = 250
	

- The purpose of the Butterworth LPF is to confine the region of applying inverse filtering scheme. That is, to let the Gaussian LPF deblurring more visible and observable, we need to modify the D0 and power n of Butterworth LPF. The smaller the D0 is, the fewer frequencies are kept to do the deblurring, and that means the image will become blurrier before blurring, so I set D0 to be 250. Next, change of n will influence the quality of images. When n is much smaller, some white noises generate unexpectedly, so I set it to be 10.
- D0 of Gaussian LPF will affect sharpness as well as blurring directly. The picture above showed that when D0 becomes larger, the image becomes a little blurrier. A special situation is that D0=100 makes the image not visible. It’s all white. I think this is because the frequency of too few ranges appears in the range limited by Butterworth LPF, resulting in the loss of the deblur function, which will make the picture change.