Assignment 6

- Self-study and summarize about How to decompose 2D filter into 1D filter
- Apply Max, Min, Median, and Weighted Median filters to two images and discuss effects of these filters

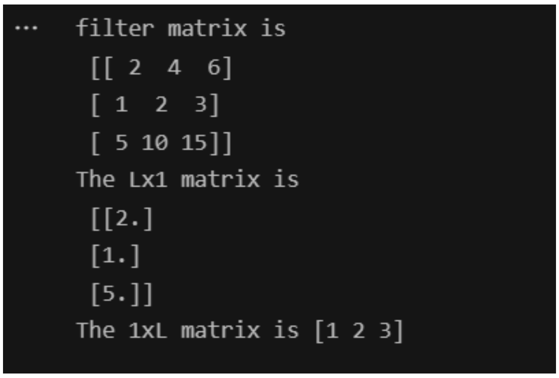
Seperable Filter

According to the common filters; for sharpening an image, blur, or detect edges, the filter will be a 2D matrix which size M x N independent. Therefore, the computational size is also M x N. The larger the filter size, the more memory is taken for processing the image. Moreover, it can be quadratic scaling with the filter's spatial extension due to the multiply-add operation from M x N. For example, image M x N size with a 2D filter L x L size, the result will be L^2MN due to multiply-accumulation from convolution.

Then, the separable filters are the solution to these problems. When the filter is separable, we can decompose it from a 2D filter to two 1D filters in different directions; horizontal (row) and vertical (column). For example, image M x N size with a 2D filter L x L size, the result will be 2LMN which means multiply first and then sum two 1D convolution results. It's obviously a different memory which is taken.

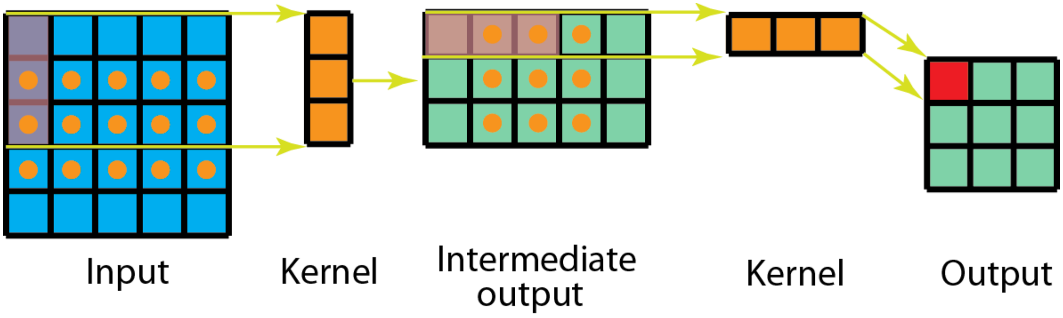
<u>The way to separate 2D filter to two 1D filters</u>

1. separate a filter from L x L into L x 1 and 1 x L



```
...   filter matrix is
       [[ 2  4  6]
       [ 1  2  3]
       [ 5 10 15]]
      The Lx1 matrix is
       [[2.]
       [1.]
       [5.]]
      The 1xL matrix is [1 2 3]
```

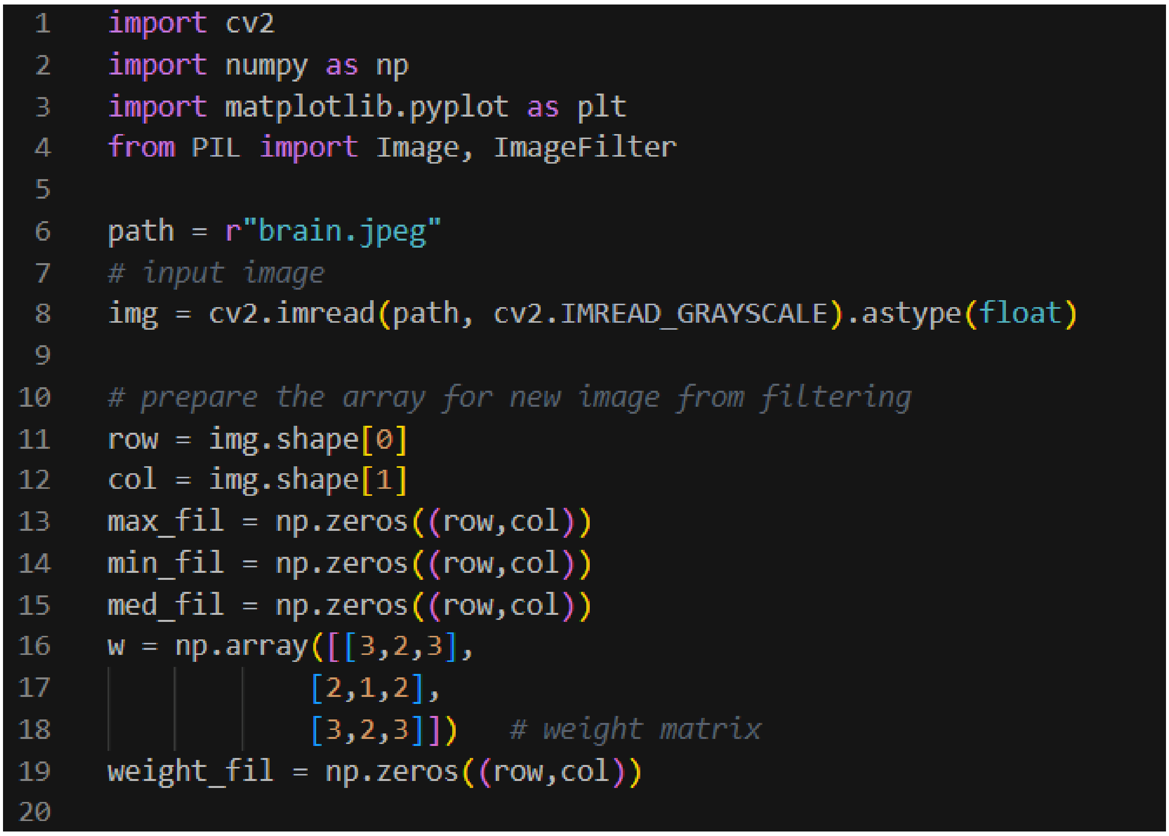
2. convolution matrix Lx1 with the image first, then the result will be called "intermediate image"
3. convolution matrix 1xL with the intermediate image from previous one and get the final filtered image



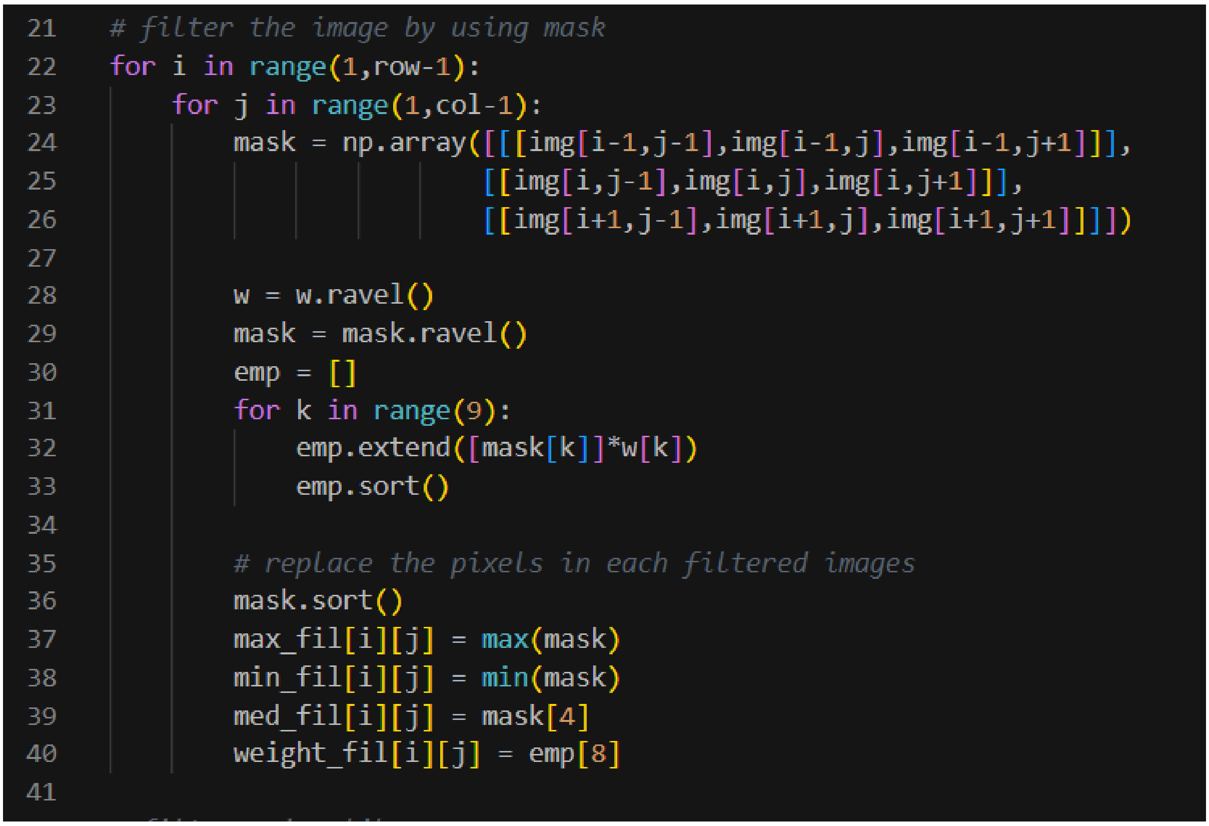
Ganesh, Prakhar. "Types of Convolution Kernels : Simplified | by Prakhar Ganesh." *Towards Data Science*, 18 October 2019, https://towardsdatascience.com/types-of-convolution-kernels-simplified-f040cb307c37. Accessed 21 October 2022.

Non-linear filters

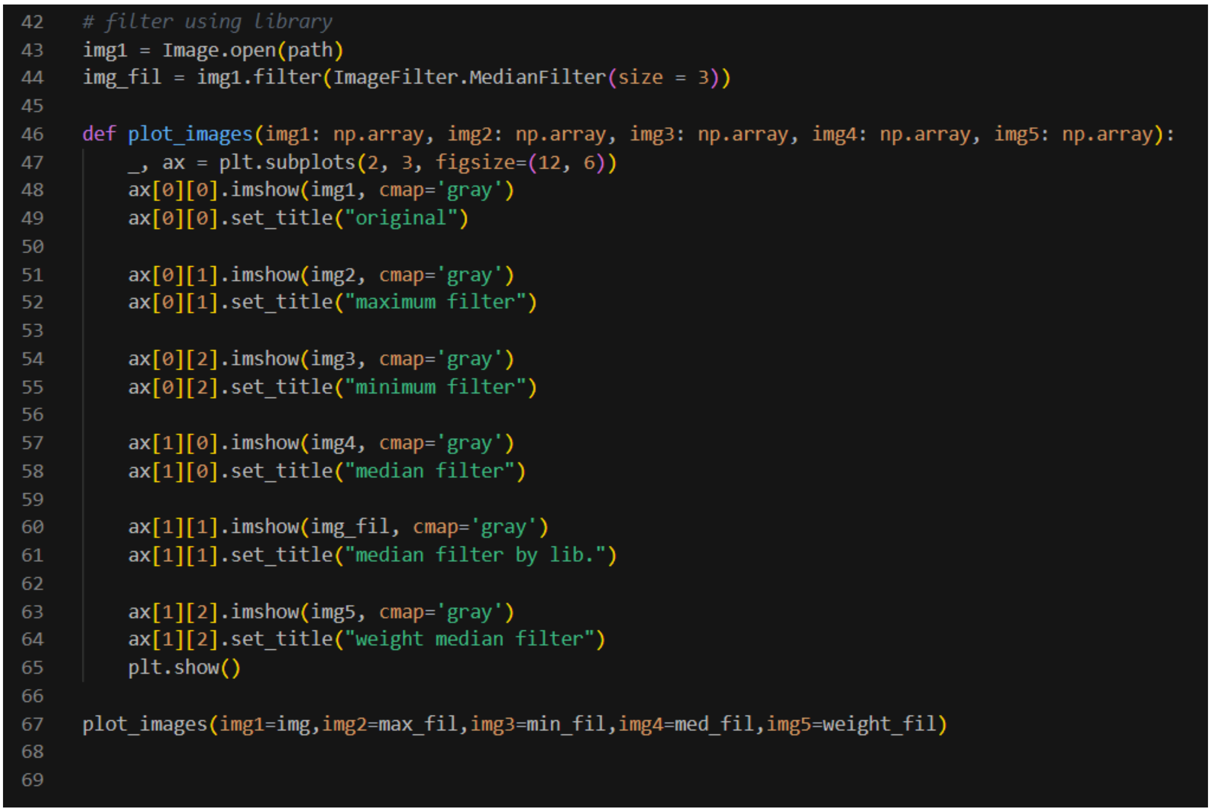
There are maximum filter which reduces pepper noise (black pixels), minimum filter which reduces salt noise (white noise), median filter which solves the pepper and salt noise by using the median value to filter the image instead of max or min values, and weight median filter which adds the weight for the filter. I provide them in the same code and plot the results for comparison as below.



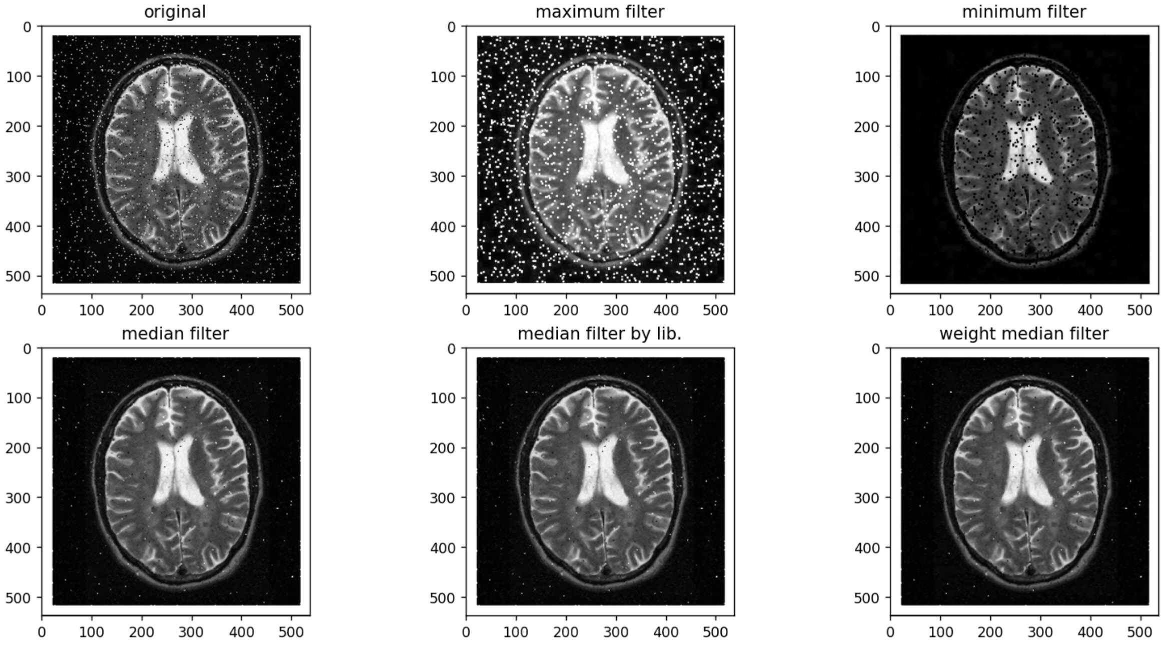
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image, ImageFilter

path = r"brain.jpeg"
# input image
img = cv2.imread(path, cv2.IMREAD_GRAYSCALE).astype(float)

# prepare the array for new image from filtering
row = img.shape[0]
col = img.shape[1]
max_fil = np.zeros((row,col))
min_fil = np.zeros((row,col))
med_fil = np.zeros((row,col))
w = np.array([[3,2,3],
            [2,1,2],
            [3,2,3]])   # weight matrix
weight_fil = np.zeros((row,col))

```



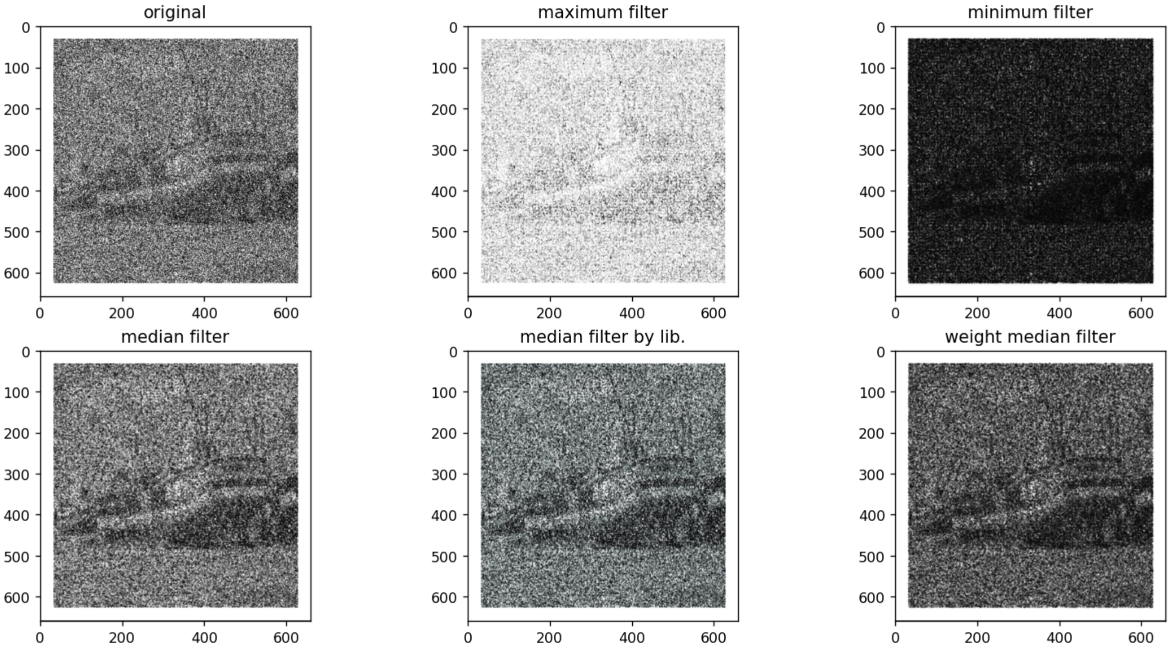
```
# filter the image by using mask
for i in range(1,row-1):
    for j in range(1,col-1):
        mask = np.array([[[img[i-1,j-1],img[i-1,j],img[i-1,j+1]]],
                        [[img[i,j-1],img[i,j],img[i,j+1]]],
                        [[img[i+1,j-1],img[i+1,j],img[i+1,j+1]]]])

        w = w.ravel()
        mask = mask.ravel()
        emp = []
        for k in range(9):
            emp.extend([mask[k]]*w[k])
            emp.sort()

        # replace the pixels in each filtered images
        mask.sort()
        max_fil[i][j] = max(mask)
        min_fil[i][j] = min(mask)
        med_fil[i][j] = mask[4]
        weight_fil[i][j] = emp[8]

```



```
# filter using library
img1 = Image.open(path)
img_fil = img1.filter(ImageFilter.MedianFilter(size = 3))

def plot_images(img1: np.array, img2: np.array, img3: np.array, img4: np.array, img5: np.array):
    _, ax = plt.subplots(2, 3, figsize=(12, 6))
    ax[0][0].imshow(img1, cmap='gray')
    ax[0][0].set_title("original")

    ax[0][1].imshow(img2, cmap='gray')
    ax[0][1].set_title("maximum filter")

    ax[0][2].imshow(img3, cmap='gray')
    ax[0][2].set_title("minimum filter")

    ax[1][0].imshow(img4, cmap='gray')
    ax[1][0].set_title("median filter")

    ax[1][1].imshow(img_fil, cmap='gray')
    ax[1][1].set_title("median filter by lib.")

    ax[1][2].imshow(img5, cmap='gray')
    ax[1][2].set_title("weight median filter")
    plt.show()

plot_images(img1=img,img2=max_fil,img3=min_fil,img4=med_fil,img5=weight_fil)


```



From the results, the maximum filter picks the maximum values which are white. Then, it will show the white area and delete the black ones. On the other hand, the minimum filter picks the lowest values which are black to filter, then the black area is shown more. For the median filter, it uses the median values to filter which can reduce both white and black noise. Finally, the weight median filter is similar to the median filter but weighing the values. Then, we can adjust the weight which affects the black or white tone of the image.



Another set of results, the input image is unknown. Then, I apply the filter to see more detail of the image. From the median and weight median filters, we can guess that it may be a boat at the quay. To be more specific, it needs more adjustment.