Batuhan Gülten
504181290

BBL 588E

HW 1

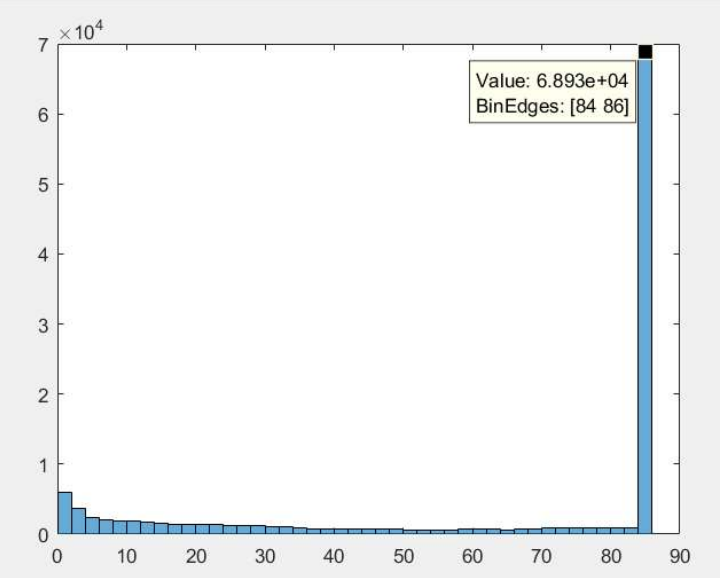
In order to complete given HW, MATLAB environment is used. Therefore, comments are made on matlab codes. All main code and functions are uploaded.

From HW part 1 to 6, given tasks are performed on same code block;

```
%% HW part 1-6
clc;
clear all;
close all;
img= imread('SunnyLake.bmp');
I=(img(:,:,1)+img(:,:,2)+img(:,:,3))/3;
imshow(I);
figure;
h=histogram(I);
[i_1,j_1]=size(I);
for i = 1: i_1
    for j= 1: j_1
    if I (i,j) > 83
      BI(i,j)=1;
    else
         BI(i,j)=0;
    end
    end
end
figure;
imshow(BI);
```

Greyscale image is shown below;

In order to achieve a threshold for binary imaging, we observe the histogram:



Obviously, most of the pixels are around pixel value 84 which corresponds to bright pixels. Therefore we pick the threshold value as 83. Resulting image is shown below;

For HW part 7, following codes are used;

```
%%  Noise Adding part 7
sig=1; V=(sig/256)^2     %
I=imnoise(img,'gaussian',0,V);
I_1=(I(:,:,1)+I(:,:,2)+I(:,:,3))/3;

sig=5; V=(sig/256)^2     %
```

```
I=imnoise(img,'gaussian',0,V);
I_5=(I(:,:,1)+I(:,:,2)+I(:,:,3))/3;

sig=10; V=(sig/256)^2     %
I=imnoise(img,'gaussian',0,V);
I_10=(I(:,:,1)+I(:,:,2)+I(:,:,3))/3;

sig=20; V=(sig/256)^2     %
I=imnoise(img,'gaussian',0,V);
I_20=(I(:,:,1)+I(:,:,2)+I(:,:,3))/3;

figure;
imshow(I_1);
title('SD=1');
figure;
imshow(I_5);
title('SD=5');
figure;
imshow(I_10);
title('SD=10');
figure;
imshow(I_20);
title('SD=20');
```

Noisy images are shown below (SD stands for standard deviation for the gaussian noise);

SD=5

SD=10

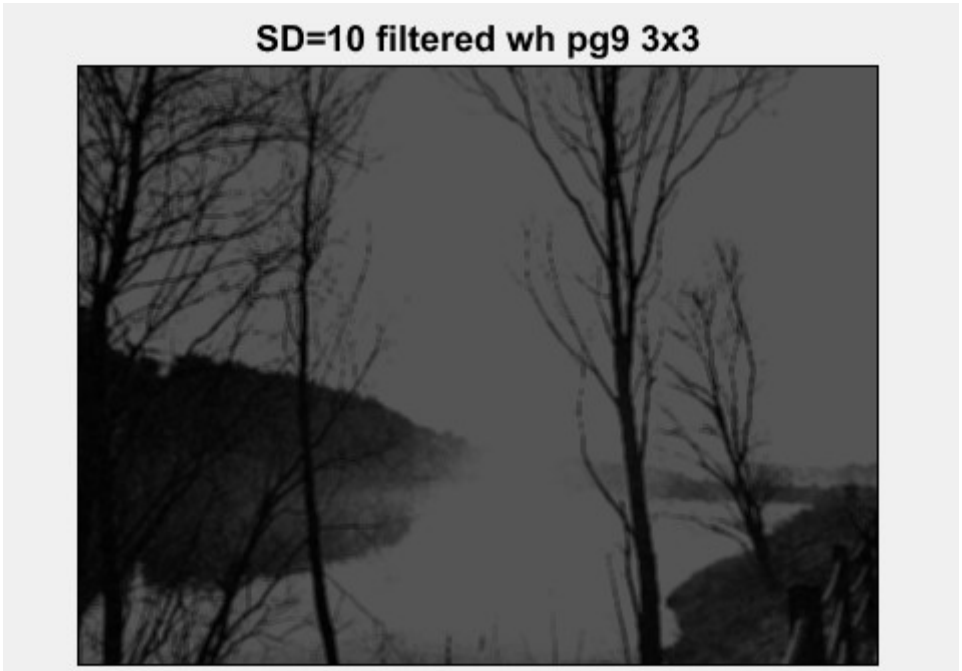
Filtered images and codes are shown below;

```
%% part 8-2
filter_9_2=ones(5)/25;
filtered_I_1 = filter_img(double(I_1),filter_9_2);
filtered_I_5 = filter_img(double(I_5),filter_9_2);
filtered_I_10 = filter_img(double(I_10),filter_9_2);
filtered_I_20 = filter_img(double(I_20),filter_9_2);

figure;
imshow(uint8(filtered_I_1));
title('SD=1 filtered wh pg9 5x5');
figure;
imshow(uint8(filtered_I_5));
title('SD=5 filtered wh pg9 5x5');
figure;
imshow(uint8(filtered_I_10));
title('SD=10 filtered wh pg9 5x5');
figure;
imshow(uint8(filtered_I_20));
title('SD=20 filtered wh pg9 5x5');
```

SD=1 filtered wh pg9 3x3

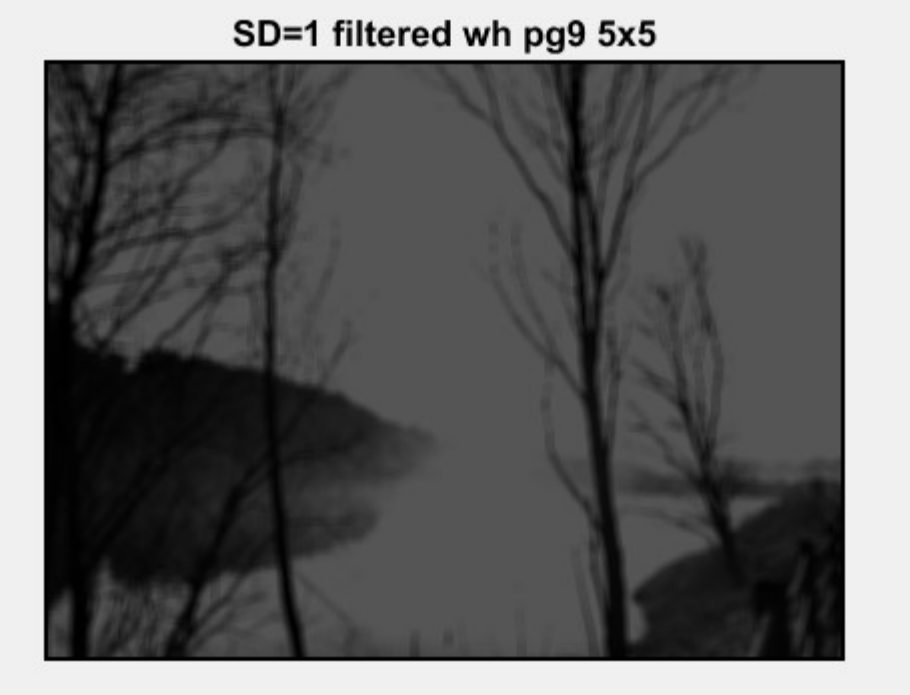
SD=5 filtered wh pg9 3x3

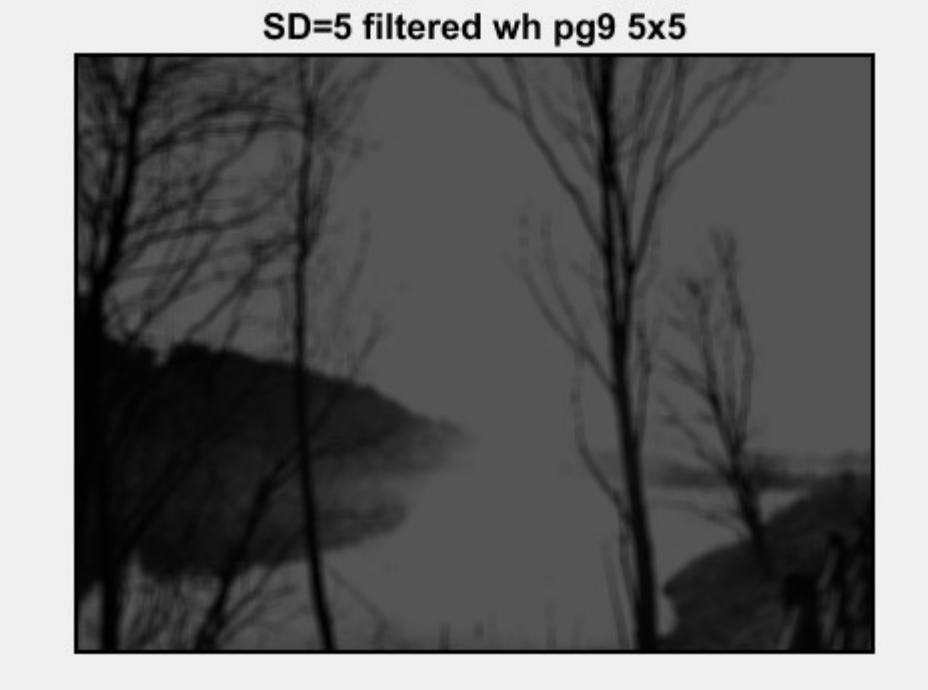
,

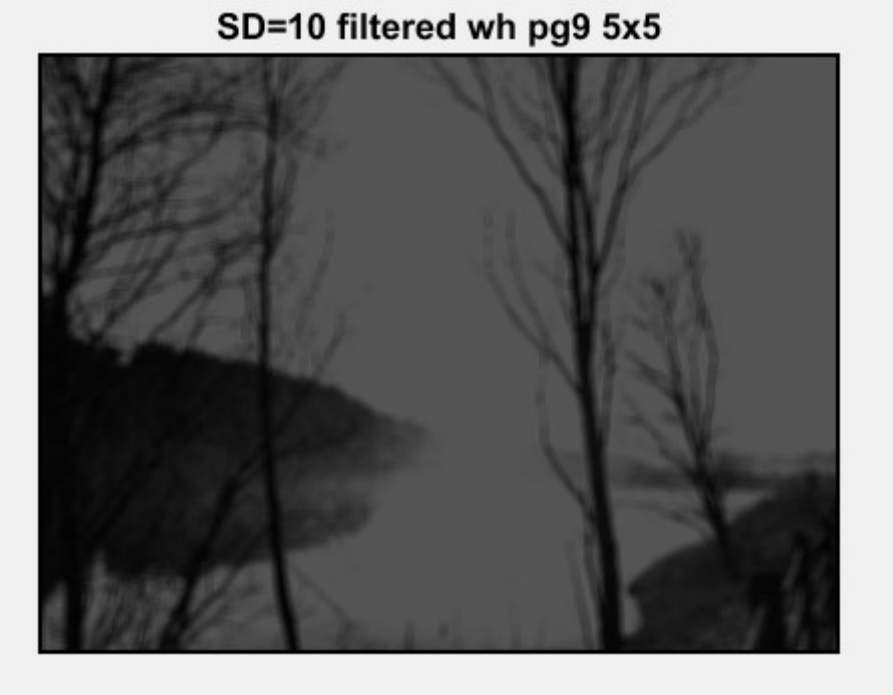



```
%% part 8-3
filter_12=[1,2,1;2,4,2;1,2,1];
filter_12=filter_12/24;
filtered_I_1 = filter_img(double(I_1),filter_12);
filtered_I_5 = filter_img(double(I_5),filter_12);
filtered_I_10 = filter_img(double(I_10),filter_12);
filtered_I_20 = filter_img(double(I_20),filter_12);

figure;
imshow(uint8(filtered_I_1));
title('SD=1 filtered wh pg12');
figure;
```

```
imshow(uint8(filtered_I_5));
title('SD=5 filtered wh pg12');
figure;
imshow(uint8(filtered_I_10));
title('SD=10 filtered wh pg12');
figure;
imshow(uint8(filtered_I_20));
title('SD=20 filtered wh pg12');
```

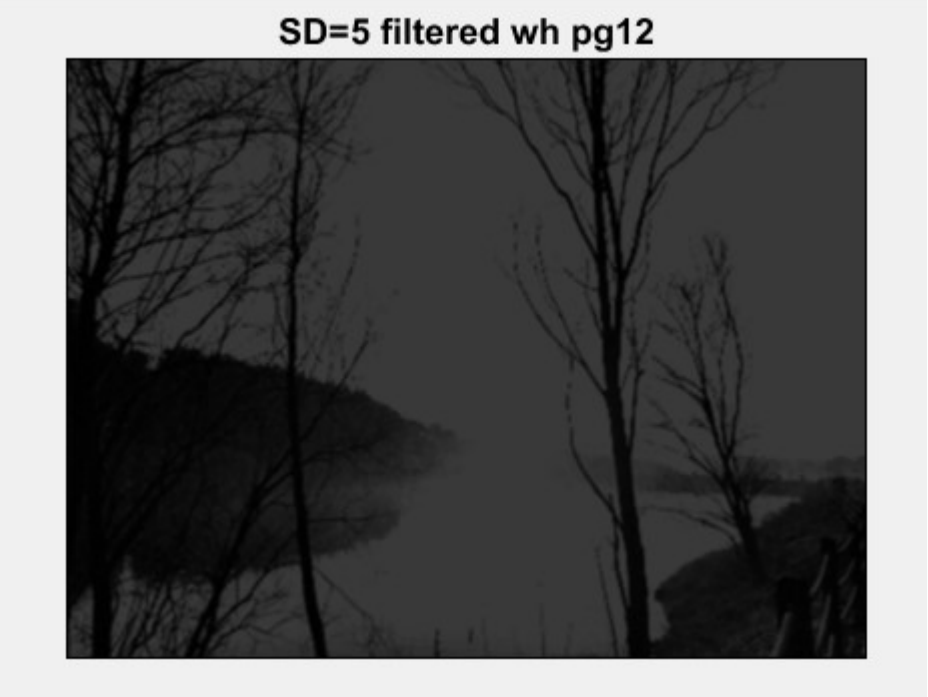


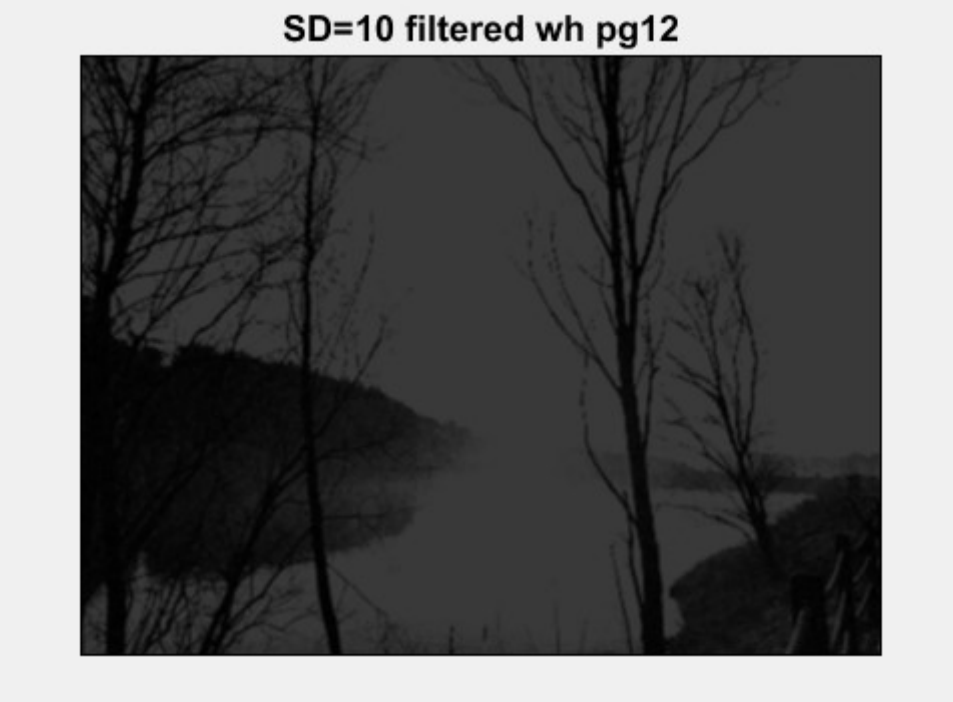


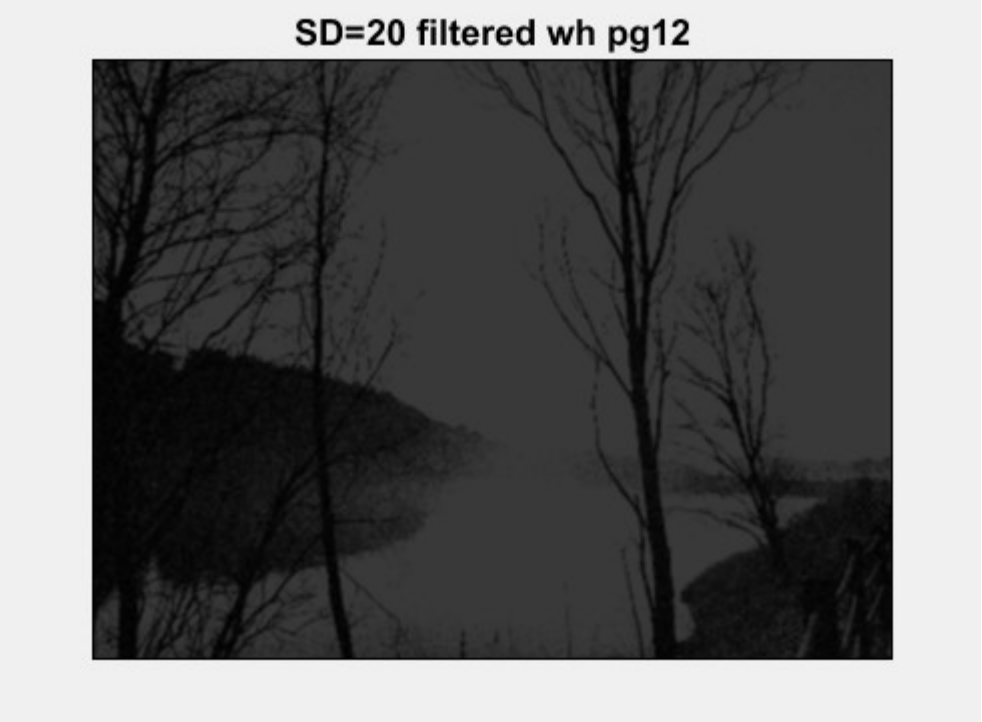



```
%% part 8-3
filter_12=[1,2,1;2,4,2;1,2,1];
filter_12=filter_12/24;
filtered_I_1 = filter_img(double(I_1),filter_12);
filtered_I_5 = filter_img(double(I_5),filter_12);
filtered_I_10 = filter_img(double(I_10),filter_12);
filtered_I_20 = filter_img(double(I_20),filter_12);

figure;
imshow(uint8(filtered_I_1));
title('SD=1 filtered wh pg12');
figure;
```

```
imshow(uint8(filtered_I_5));
title('SD=5 filtered wh pg12');
figure;
imshow(uint8(filtered_I_10));
title('SD=10 filtered wh pg12');
figure;
imshow(uint8(filtered_I_20));
title('SD=20 filtered wh pg12');
```







Both filters can be used for given type of noise, they are both low pass filters. However, gaussian filters weight distribution (weight more to the center pixel rather than equal distribution over all neighborhoods) causes more informative image and more appealing blurriness to the eye.

For part 9, following results have been achieved:

```
%% part 9-1
filter_17_1=[-1,-1,-1;-1,8,-1;-1,-1,-1];

filtered_I_1 = filter_img(double(I_1),filter_17_1);
filtered_I_5 = filter_img(double(I_5),filter_17_1);
```

```
filtered_I_10 = filter_img(double(I_10),filter_17_1);
filtered_I_20 = filter_img(double(I_20),filter_17_1);

figure;
imshow(uint8(filtered_I_1));
title('SD=1 filtered wh pg17_1');
figure;
imshow(uint8(filtered_I_5));
title('SD=5 filtered wh pg17_1');
figure;
imshow(uint8(filtered_I_10));
title('SD=10 filtered wh pg17_1');
figure;
imshow(uint8(filtered_I_20));
title('SD=20 filtered wh pg17_1');
```

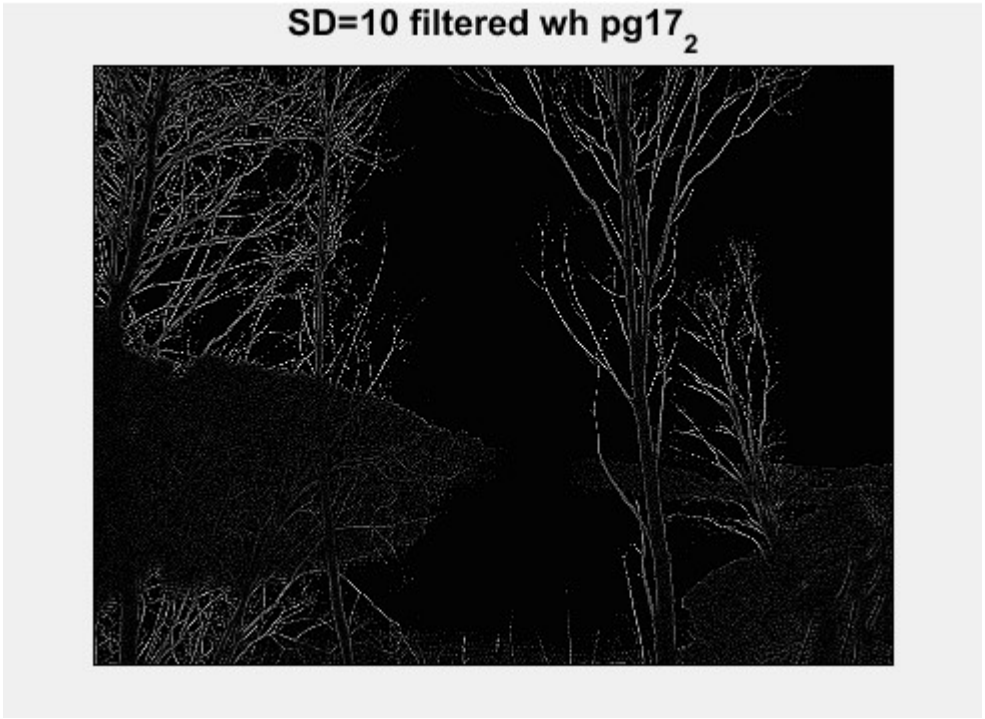
SD=5 filtered wh pg17$_1$

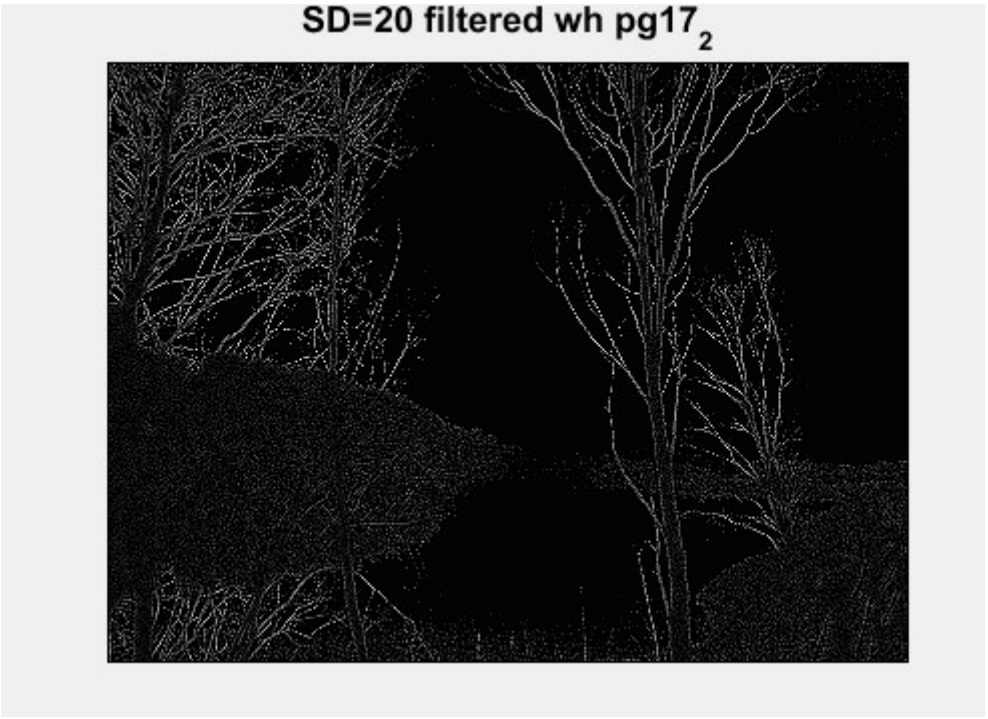
SD=10 filtered wh pg17$_1$


```
%% part 9-2
filter_17_2=[.17,.67,.17;.67,-3.33,.67;.17,.67,.17];

filtered_I_1 = filter_img(double(I_1),filter_17_2);
filtered_I_5 = filter_img(double(I_5),filter_17_2);
filtered_I_10 = filter_img(double(I_10),filter_17_2);
filtered_I_20 = filter_img(double(I_20),filter_17_2);

figure;
imshow(uint8(filtered_I_1));
title('SD=1 filtered wh pg17_2');
figure;
imshow(uint8(filtered_I_5));
title('SD=5 filtered wh pg17_2');
figure;
imshow(uint8(filtered_I_10));
title('SD=10 filtered wh pg17_2');
figure;
imshow(uint8(filtered_I_20));
title('SD=20 filtered wh pg17_2');
```

SD=1 filtered wh $pg17_2$

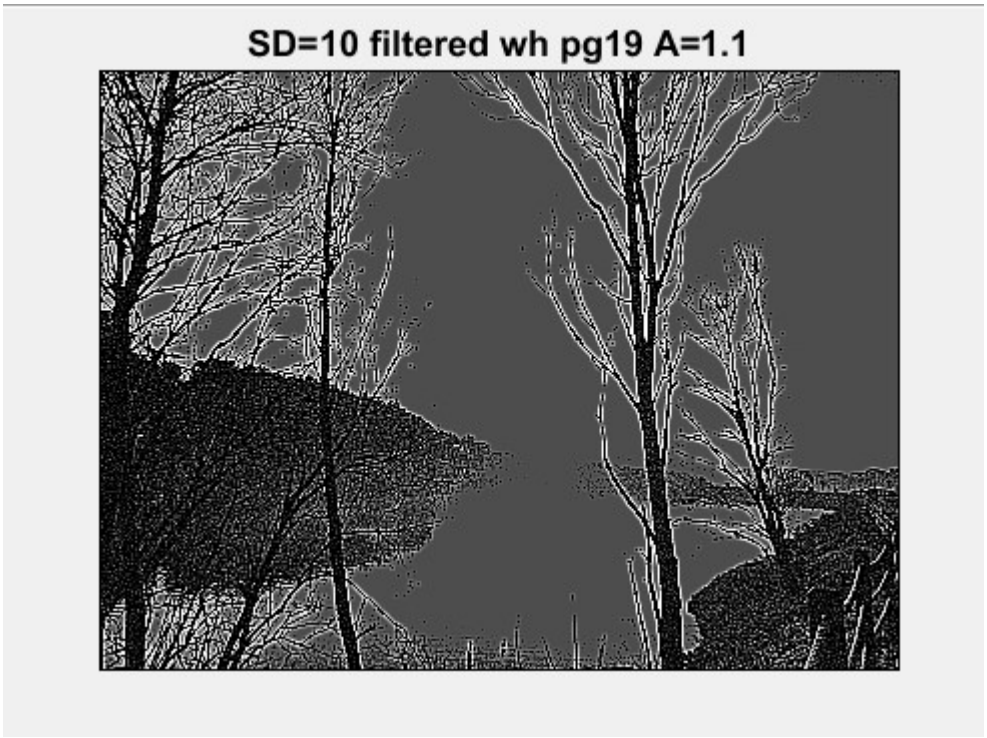
SD=5 filtered wh $pg17_2$

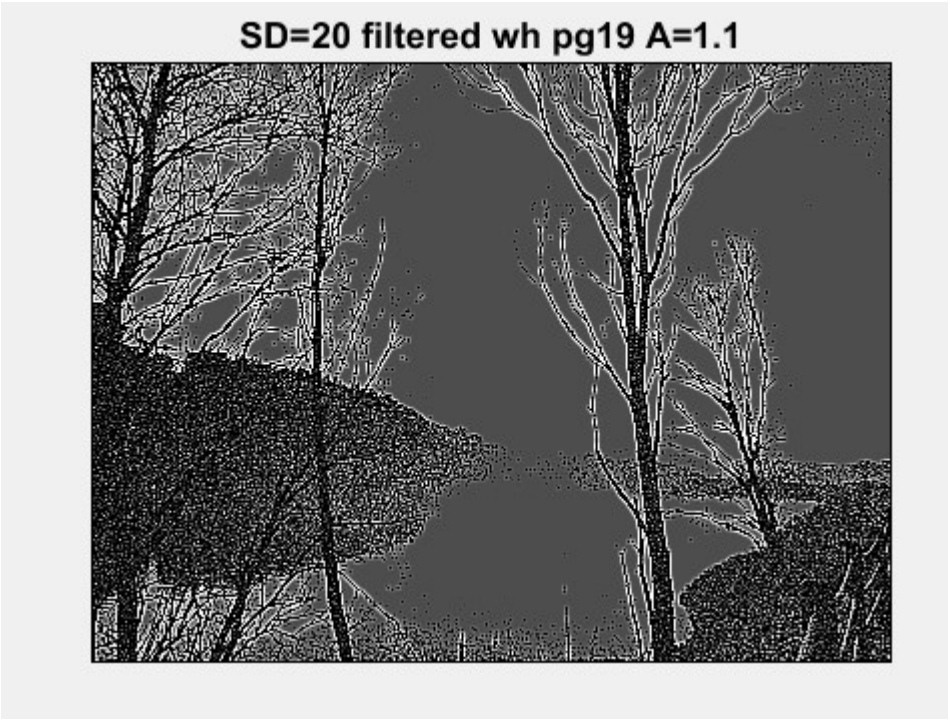





```
A=1.1;
filter_19=[-1,-1,-1;-1,9*A-1,-1;-1,-1,-1];

filtered_I_1 = filter_img(double(I_1),filter_19);
filtered_I_5 = filter_img(double(I_5),filter_19);
filtered_I_10 = filter_img(double(I_10),filter_19);
filtered_I_20 = filter_img(double(I_20),filter_19);

figure;
imshow(uint8(filtered_I_1));
title('SD=1 filtered wh pg19 A=1.1');
```

```
figure;
imshow(uint8(filtered_I_5));
title('SD=5 filtered wh pg19 A=1.1');
figure;
imshow(uint8(filtered_I_10));
title('SD=10 filtered wh pg19 A=1.1');
figure;
imshow(uint8(filtered_I_20));
title('SD=20 filtered wh pg19 A=1.1');
```





When we observe the outcome images of given high pass filters, we can see the high frequency components of the noise. For example, if these images would be used in edge detection problem, the algorithm would fail due to noisy elements. The second kernel in pg 17 might have been slightly better than the other kernels in this matter, but hardly enough. Thus, to overcome such a problem, noisy image should be filtered with low pass filter first.

For the last part of the HW, we are given an impulsive noisy image. We can use median filters for given problem. The **median** is calculated by first sorting all the pixel values from the window into numerical order, and then replacing the pixel being considered with the middle (**median**) pixel value.

We have following codes and results:

```
noise_img = imread('SP_Noisy_SunnyLake.png');
figure;
imshow(noise_img);
title('impulsive noisy image');
filtered_image=medfilt3(noise_img);
figure;
imshow(filtered_image);
title('median filtered image');
```

median filtered image