Assignment 4

Neeraj Kumar (2020BSZ8607) bsz208607@iitd.ac.in
Krishna Chaitanya Reddy Tamataam(2018MT60785) mt6180785@maths.iitd.ac.in
Chayan Kumar Paul(2020EEZ7528) eez207528@iitd.ac.in
IIT Delhi

5th, November 2020

1 Question 1

1.1 Adding Noise to the image

1.1.1 MATLAB Code

```
x = imread('lenna.jpg');
I_lenna = rgb2gray(x);

out1_gaussian = uint8(awgn(double(I_lenna) , 25 , 'measured'));
d = 1/(1+(10^(30/10)));
out1_salt =  imnoise(I_lenna,'salt & pepper',d);

I_camera = imread('cameraman.jpg');


out2_gaussian = uint8(awgn(double(I_camera) , 20 , 'measured'));
d = 1/(1+(10^(25/10)));
out2_salt =  imnoise(I_camera,'salt & pepper',d);


out1_poission = imnoise(I_lenna,'poisson');
out2_poission = imnoise(I_camera,'poisson');



figure
imshow(out1_gaussian)
figure
imshow(out1_salt)
figure
imshow(out1_poission)
figure
imshow(out2_gaussian)
figure
imshow(out2_salt)
figure
imshow(out2_poission)
```

1.1.2 Results

Figure 1: **From left to right ; original lenna image,image and Gaussian noise of 25 db added , image and salt and pepper noise of 30 db added , image and Poisson noise added**

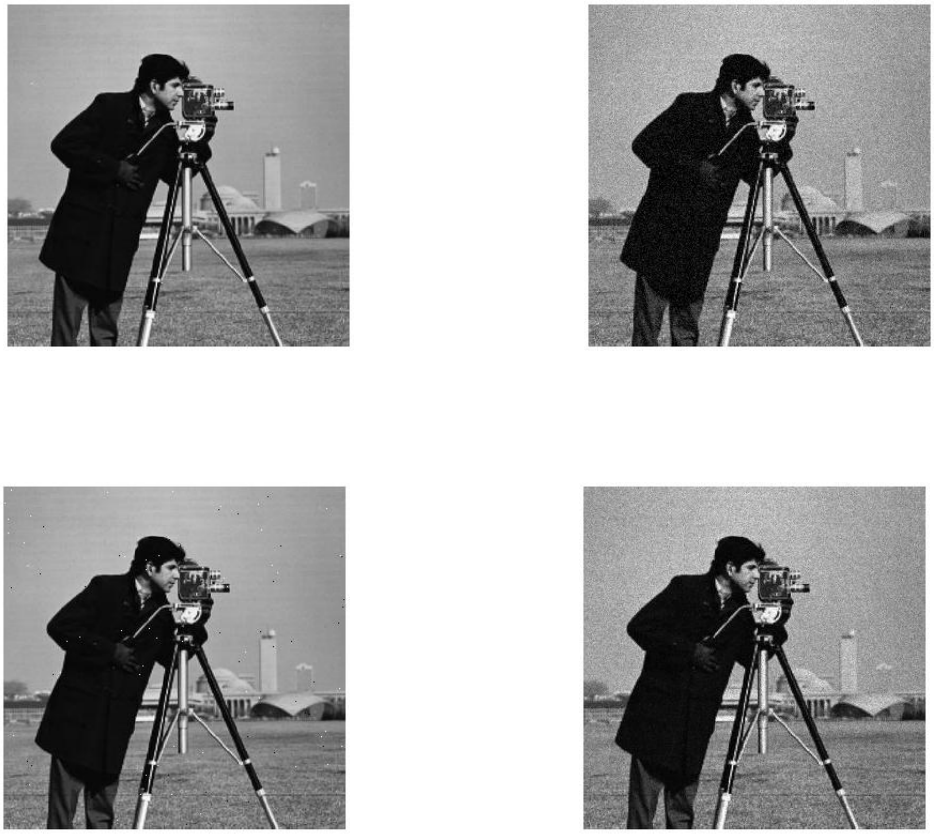

Figure 2: **From left to right ; original cameraman image,image and Gaussian noise of 20 db added , image and salt and pepper noise of 25 db added , image and Poisson noise added**

1.1.3 Comments

In case of salt and pepper noise and Gaussian noise the parameters like the mean and standard deviation is fixed throughout the image.Poisson noise is added by a Poisson process where the output comes from a Poisson distribution whose mean is the input pixel value.since Poisson distribution gives only discrete value the output is quantized. It is a signal dependent noise.

1.2 Removing Noise using Wavelet Coding

1.2.1 MATLAB Code

```
show = I_camera;
[thr,sorh,keepapp] = ddencmp('den','wv',out2_gaussian);
gau = wdencmp('gbl',out2_gaussian,'Haar',2,thr,sorh,keepapp);
figure
imshow(uint8(gau))

psnr_gau = 20*log10(max(max(double(show)))/double(mean(mean((show - uint8(gau))
    .^2))))

[thr,sorh,keepapp] = ddencmp('den','wv',out2_salt);
salt = wdencmp('gbl',out2_salt,'Haar',2,thr,sorh,keepapp);
figure
imshow(uint8(salt))
psnr_sal = 20*log10(max(max(double(show)))/double(mean(mean((show - uint8(salt))
    .^2))))

[thr,sorh,keepapp] = ddencmp('den','wv',out2_poission);
poi = wdencmp('gbl',out2_poission,'Haar',2,thr,sorh,keepapp);
figure
imshow(uint8(poi))
psnr_poi = 20*log10(max(max(double(show)))/double(mean(mean((show - uint8(poi))
    .^2))))
```

1.2.2 Best Images Restored

Figure 3: **Best picture restores, left lenna with salt and pepper noise restored by Symlets(Sym4) Wavelet and Right cameraman with salt and pepper noise restored by Daubechies(db4) Wavelet**

1.2.3 Best Images with Gaussian Noise Restored

Figure 4: **Best picture restores, left lenna restored by Symlets(Sym4) Wavelet and Right cameraman restored by Haar Wavelet**

1.2.4 Best Images with SaltPepper Noise Restored

Figure 5: **Best picture restores, left lenna restored by Symlets(Sym4) Wavelet and Right cameraman restored by Daubechies(db4) Wavelet**

1.2.5 Best Images with Poisson Noise Restored

Figure 6: **Best picture restores, left lenna restored by Symlets(Sym4) Wavelet and Right cameraman restored by Symlets(Sym4) Wavelet**

1.2.6 Best Images restored by Haar Transform

Figure 7: **Best picture restores, left lenna restored by Haar Wavelet and Right cameraman restored by Haar Wavelet**

1.2.7 Best Images restored by db4 Transform


Figure 8: **Best picture restores, left lenna restored by Daubechies(db4) Wavelet and Right cameraman restored by Daubechies(db4) Wavelet**

1.2.8 Best Images restored by Sym4 Transform


Figure 9: **Best picture restores, left lenna restored by Symlets(Sym4) Wavelet and Right cameraman restored by Symlets(Sym4) Wavelet**

1.2.9 Observations

Table 1: PSNR values for different restored lenna images

Image	**Haar**	**db4**	**sym4**
gaussian noise(25 db)	16.8881	17.7561	17.8010
SaltPepper noise(30 db)	21.9346	23.2764	23.2929
Poisson noise	16.0500	16.8606	16.9449

Table 2: PSNR values for different restored Cameraman images

Noise Image	**Haar**	**db4**	**sym4**
gaussian noise(20 db)	18.6257	18.5259	18.5984
SaltPepper noise(25 db)	30.8202	31.1735	31.1603
Poisson noise	19.055	19.0928	19.2098

1.2.10 Comments

- All the wavelets transform methods perform similarly with all the noise with little variation as seen in table 1 and 2 .
- Image restored from the images having salt and pepper noise are being best restored as compared to image with other noise such as Gaussian and Poisson noise as the low wavelet coefficients are capturing such noise components which get removed after removing these and reconstructing the inverse wavelet operation.
- Gaussian Noise kind of smoothen the image and higher frequency component gets lost , so in wavelets transform it did not gets captured , so less SNR.

2 Question 2

2.1 Wavelet Coding

2.1.1 Python Code

```
#for haar wavelet

import numpy as np
import matplotlib.pyplot as plt

import pywt
import pywt.data
import cv2

k = 20

original = cv2.imread('lena.jpeg',0)
# original = cv2.cvtColor(original, cv2.COLOR_BGR2GRAY)


coeffs2 = pywt.dwt2(original, 'haar') #db1
LL, (LH, HL, HH) = coeffs2



plt.imshow(LL,cmap='gray')
```

```
import numpy as np
import matplotlib.pyplot as plt

import pywt
import pywt.data
import cv2

k = 20

original = cv2.imread('lenna.jpg',0)
# original = cv2.cvtColor(original, cv2.COLOR_BGR2GRAY)


coeffs2 = pywt.dwt2(original, 'db1') #db1
LL, (LH, HL, HH) = coeffs2
```

```
plt.imshow(LL,cmap='gray')
```

2.1.2 Results

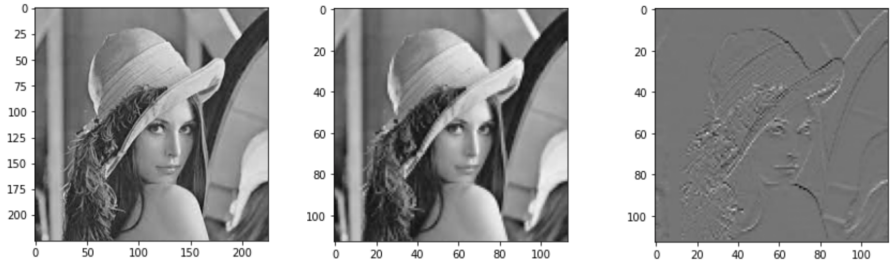

Figure 10: **Haar wavelet transform on lena image. Left : original , Centre:LL wavelet image , Right : LH wavelet image**

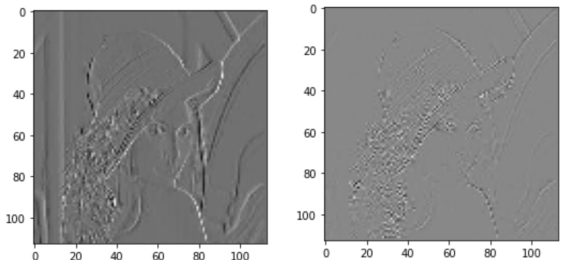

Figure 11: **HAAR wavelet transform, Left:HL wavelet image , Right : HH wavelet image**

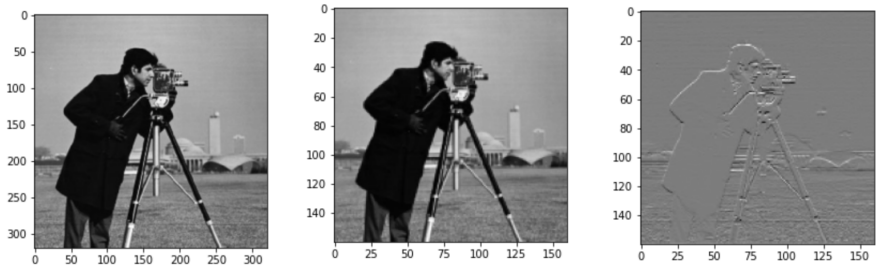

Figure 12: **Haar wavelet transform. Left : original , Centre:LL wavelet image , Right : LH wavelet image**

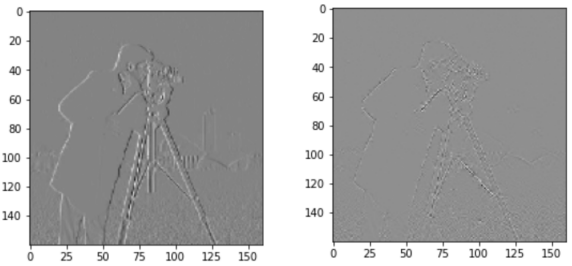

Figure 13: **HAAR wavelet transform.Left:HL wavelet image , Right : HH wavelet image**

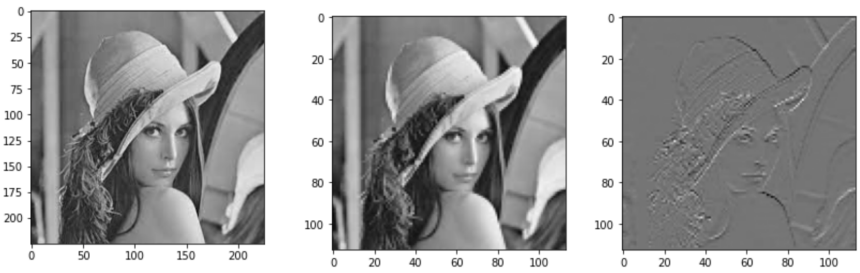

Figure 14: **Daubechies wavelet transform,Left : original , Centre:LL wavelet image , Right : LH wavelet image**

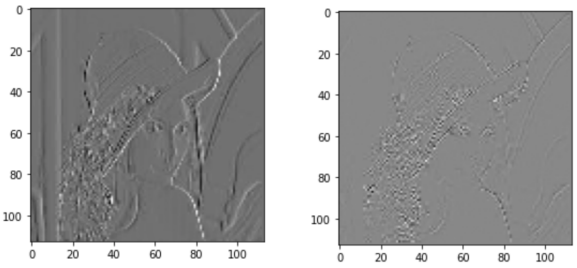

Figure 15: **Daubechies wavelet transform.Left:HL wavelet image , Right : HH wavelet image**

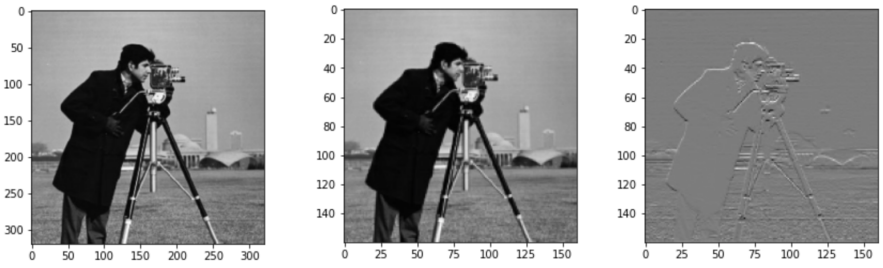

Figure 16: **Daubechies wavelet transform.Left : original , Centre:LL wavelet image , Right : LH wavelet image**

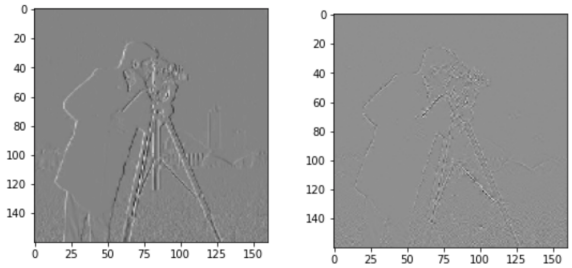

Figure 17: **Daubechies wavelet transform.Left:HL wavelet image , Right : HH wavelet image**

2.1.3 Observation

- Figure 10 and 11 shows the low and high frequency components on lena image using HAAR .Figure 12 and 13 shows the low and high frequency components on cameraman image using HAAR .
- Figure 14 and 15 shows the low and high frequency components on lena image using DB .Figure 16 and 17 shows the low and high frequency components on cameraman image using DB .

2.2 Compression Scheme

2.2.1 Python Code

```
import numpy as np
import matplotlib.pyplot as plt

import pywt
import pywt.data
import cv2

k = 5

original = cv2.imread('cameraman.jpg',0)
# original = cv2.cvtColor(original, cv2.COLOR_BGR2GRAY)


coeffs2 = pywt.dwt2(original, 'db1') #db1
LL, (LH, HL, HH) = coeffs2



new_list = []

new_list.extend(list(LL.flatten()))
new_list.extend(list(LH.flatten()))
new_list.extend(list(HL.flatten()))
new_list.extend(list(HH.flatten()))

new_list.sort(reverse=True)



topk = int((k*262144)/100)
index_value = new_list[topk]

LL[LL<index_value]=0
LH[LH<index_value]=0
HL[HL<index_value]=0
HH[HH<index_value]=0

new_coeffs =LL, (LH, HL, HH)
imgh = pywt.idwt2(new_coeffs, 'db1')

plt.imshow(imgh,cmap='gray')
```

2.2.2 Results

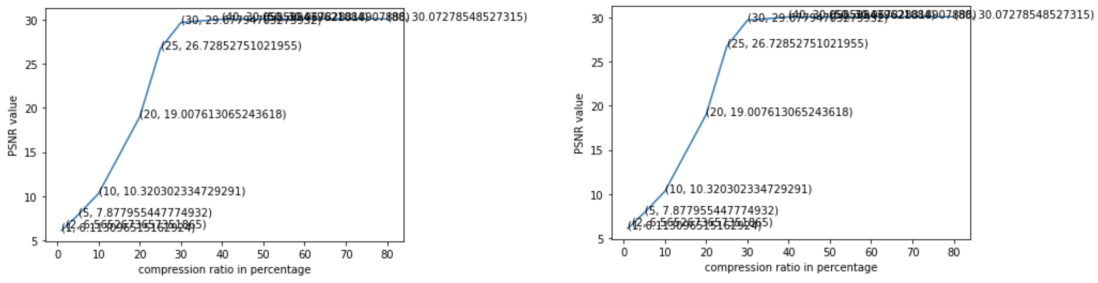

Figure 18: **Plot with respect to quality of image PSNR**

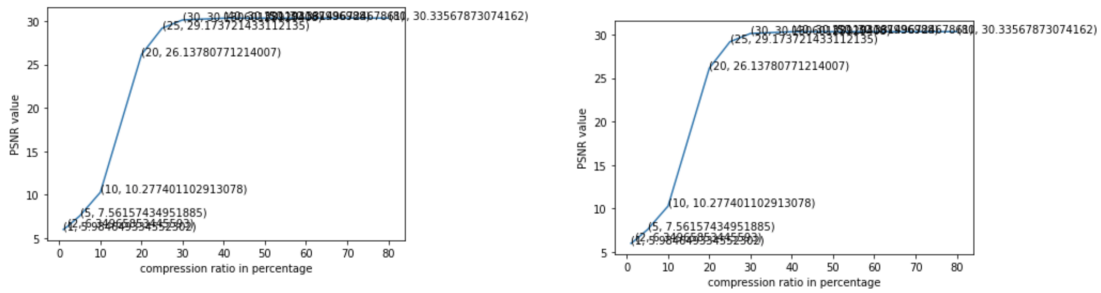

Figure 19: **Plot with respect to quality of image**

2.2.3 Observation

- In the compression scheme , we have retained the top k % wavelet coeefficient based on their value and make other as zero. In Figure 18 and 19 , we see that the PSNR value increases slowly with the increase in K . In Figure 30 and 31 , we see that the SNR value increases slowly with the increase in K .

2.3 Reconstruct and Quality of image

2.3.1 Python Code

```
import numpy as np
import matplotlib.pyplot as plt

import pywt
import pywt.data
import cv2

from math import log10, sqrt
import cv2
import numpy as np


#SNR value of image
def signaltonoise_dB(a, axis=0, ddof=0):
    a = np.asanyarray(a)
    m = a.mean(axis)
    sd = a.std(axis=axis, ddof=ddof)
    return 20*np.log10(abs(np.where(sd == 0, 0, m/sd)).mean())


#PSNR value (Quality of image)
def PSNR(original, compressed):
        mse = np.mean((original - compressed) ** 2)
        if(mse == 0): # MSE is zero means no noise is present in the signal .
                      # Therefore PSNR have no importance.
                return 100
        max_pixel = 255.0
        psnr = 20 * log10(max_pixel / sqrt(mse))
        return psnr



klist = [1,2,5,10,20,30,40,50,80]

xlabel = []
ylabel= klist

for k in klist:
```

```
    original = cv2.imread('cameraman.jpg',0)
    # original = cv2.cvtColor(original, cv2.COLOR_BGR2GRAY)


    coeffs2 = pywt.dwt2(original, 'db1') #db1
    LL, (LH, HL, HH) = coeffs2




    new_list = []

    new_list.extend(list(LL.flatten()))
    new_list.extend(list(LH.flatten()))
    new_list.extend(list(HL.flatten()))
    new_list.extend(list(HH.flatten()))

    new_list.sort(reverse=True)


    size = original.shape[0]*original.shape[1]
    topk = int((k*size)/100)
    index_value = new_list[topk]

    LL[LL<index_value]=0
    LH[LH<index_value]=0
    HL[HL<index_value]=0
    HH[HH<index_value]=0

    new_coeffs =LL, (LH, HL, HH)
    imgh = pywt.idwt2(new_coeffs, 'db1')
    print(k)
    plt.imshow(imgh,cmap='gray')
    plt.show(block=False)

    value = PSNR(original, imgh)

    print(k,value)
    xlabel.append(value)




plt.plot(ylabel,xlabel,)
plt.ylabel('PSNR value')
plt.xlabel('compression ratio in percentage')
for i_x, i_y in zip(ylabel, xlabel):
    plt.text(i_x, i_y, '({}, {})'.format(i_x, i_y))

plt.show()
```

2.3.2 Results

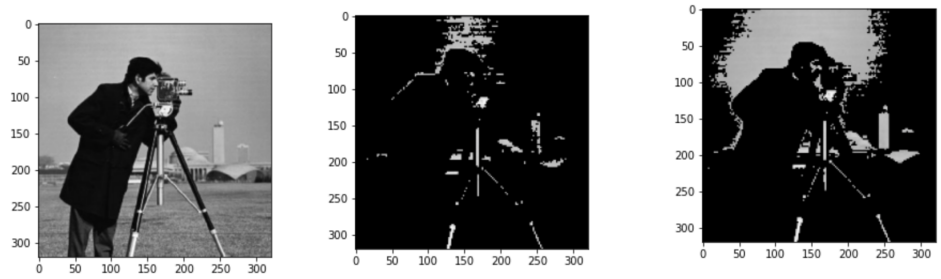

Figure 20: **Reconstruction of cameraman image with db wavelet transform for different value of k =1,5(left to right))**

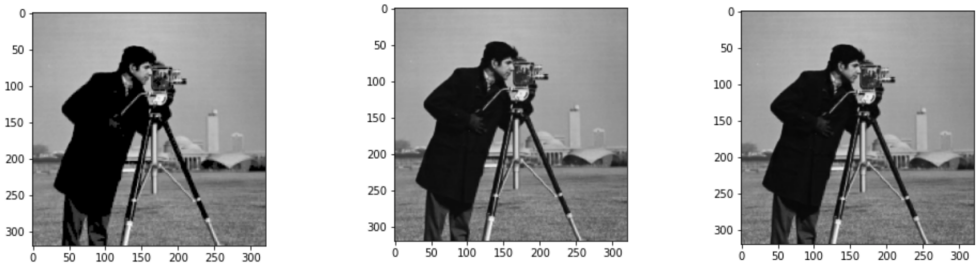

Figure 21: **Reconstruction of cameraman image with db wavelet transform for different value of k =20,30,50(left to right)**

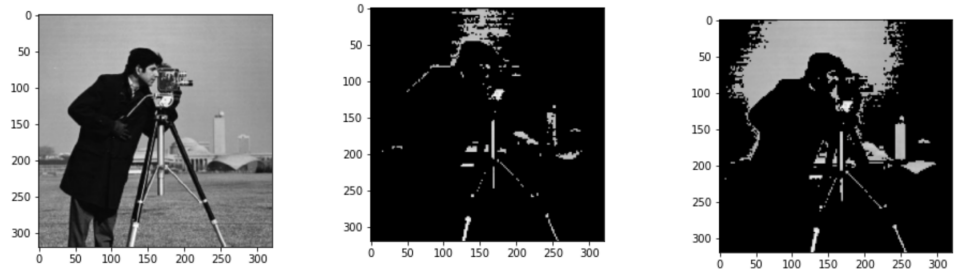

Figure 22: **Reconstruction of camwraman image with haar wavelet transform for different value of k =1,5(left to right)**

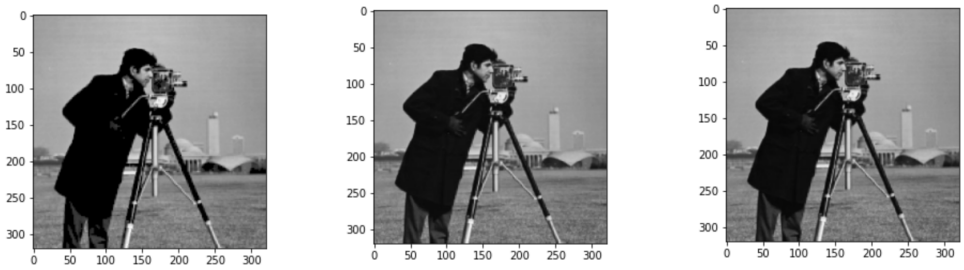

Figure 23: **Reconstruction of cameraman image with haar wavelet transform for different value of k =20,30,50(left to right)**

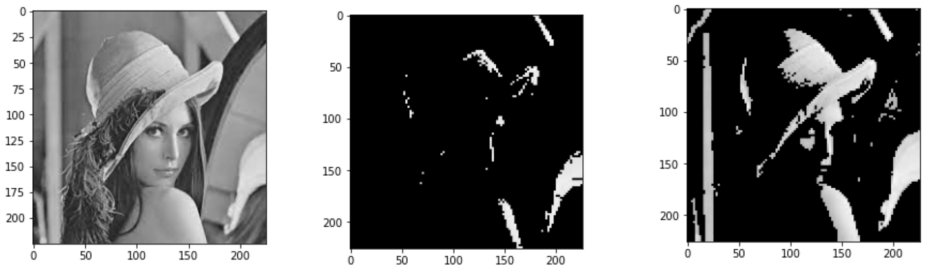

Figure 24: **Reconstruction of lena image with haar wavelet transform for different value of k =1,5(left to right)**

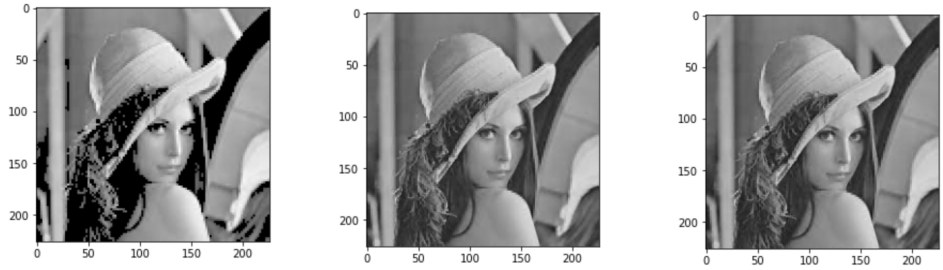

Figure 25: **Reconstruction of lenna image with haar wavelet transform for different value of k =20,30,50(left to right)**

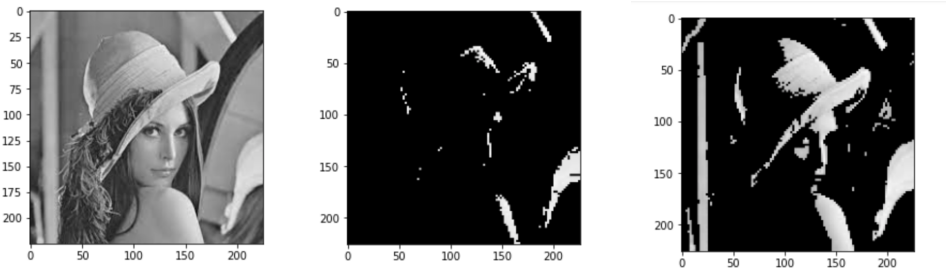

Figure 26: **Reconstruction of lenna image with db wavelet transform for different value of k =1,5(left to right)**

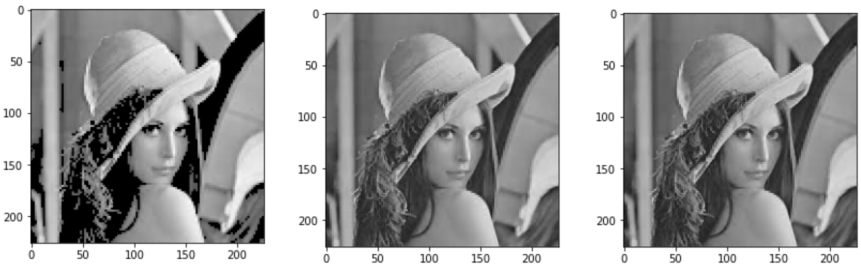

Figure 27: **Reconstruction of lenna image with db wavelet transform for different value of k =20,30,50(left to right)**

K IN PERCENTAGE	PSNR-db1-lena
1	6.1131
2	6.56527
5	7.87796
10	10.3203
20	19.0076
25	26.7285
30	29.6779
40	30.0506
50	30.0678
80	30.0728

K IN PERCENTAGE	PSNR-haar-lena
1	6.1131
2	6.56527
5	7.87796
10	10.3203
20	19.0076
25	26.7285
30	29.6779
40	30.0506
50	30.0678
80	30.0728

Figure 28: **Comparison of db and haar wavelet on quality of image(PSNR) on lenna image**

K IN PERCENTAGE	PSNR-db-cam
1	5.98465
2	6.34966
5	7.56157
10	10.2774
20	26.1378
25	29.1737
30	30.1306
40	30.3212
50	30.3315
80	30.3357

K IN PERCENTAGE	PSNR-haar-lena
1	6.1131
2	6.56527
5	7.87796
10	10.3203
20	19.0076
25	26.7285
30	29.6779
40	30.0506
50	30.0678
80	30.0728

Figure 29: **Comparison of db and haar wavelet on quality of image(PSNR) on cameraman image**

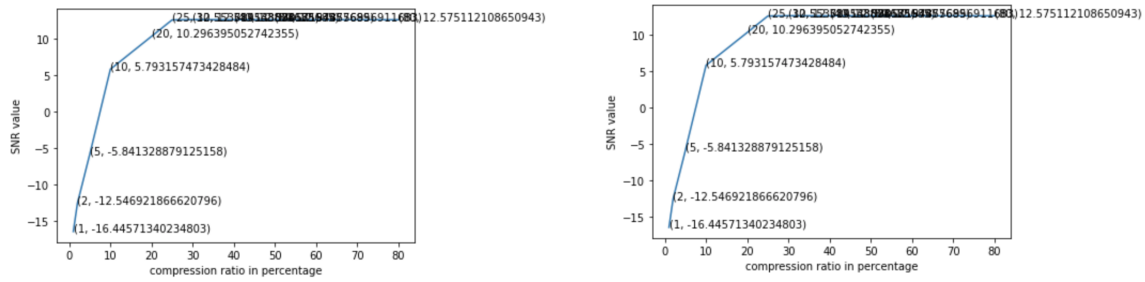

Figure 30: **Plot of SNR vs k on reconstructed lenna image. left: plot of SNR vs k for haar reconstructed image . Right : plot of SNR vs k for DB reconstructed image**

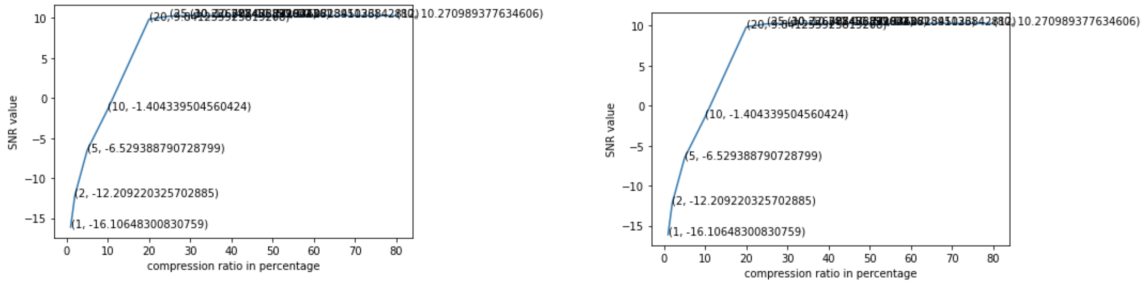

Figure 31: **Plot of SNR vs k on reconstructed cameraman image.left: plot of SNR vs k for haar reconstructed image . Right : plot of SNR vs k for DB reconstructed image**

2.3.3 Observation

- Figure 21,22,23,24,25,26,27 shows the reconstructed image for various value of k for DB and haar wavelet tranform on lenna and cameraman images.
- Comparison in terms of quality of image, We have used the metric PSNR .Higher the PSNR , higher the quality of image. The Table 28 and 29 , there is not much difference in haar and db reconstructed image at different value of k . But we do see that, at the knee point i.e. 30 percent retention of k wavelet , we have little higher psnr value in case of DB reconstructed cameraman image .
- Figure 30 and 31 shows the plot of SNR value in DB scale with respect to k . We. an see that for for lower value of K we are getting lower SNR and when we reconstruct the image with 30 % of top wavelets,

we are able to reconstruct most of the image . After increasing the gain , we are not able to gain much in SNR and the curve gets flatten.