Digital Signal Processing Laboratory Report
Experiment 2: Sampling

Rajeswari Mahapatra
15EC10044
Thursday: Group 40

1 Aim

(a) Study A-law and μ-law companding, and compare uniform and non-uniform quantization of signals

(b) Study the effects of dithered quantization on signals

(c) Study Max Floyd Optimisation of a grayscale image, and compare uniform and non-uniform quantization of images

(d) Study the effects of dithering in Image Processing using error carry

2 Part (a): Study of A-law and μ-law Companding

2.1 Theory:

The human auditory system is believed to be a logarithmic process in which high amplitude sounds do not require the same resolution as low amplitude sounds. The human ear is more sensitive to quantization noise in small signals than large signals. A-law and m-law coding apply a logarithmic quantization function to adjust the data resolution in proportion to the level of the input signal. Smaller signals are represented with greater precision – more data bits – than larger signals. The result is fewer bits per sample to maintain an audible signal-to-noise ratio (SNR).

Rather than taking the logarithm of the linear input data directly, which can be computationally difficult, A-law/μ-law PCM matches the logarithmic curve with a piece-wise linear approximation. Eight straight-line segments along the curve produce a close approximation to the logarithm function. Each segment is known as a chord. Within each chord, the piece-wise linear approximation is divided into equally size quantization intervals called steps. The step size between adjacent codewords is doubled in each succeeding chord. Also encoded is the sign bit for the original integer. The result is an 8-bit logarithmic code composed of a 1-bit sign bit, a 3-bit chord, and a 4-bit step.

A-Law Compander:
A-law is the CCITT recommended companding standard used across Europe. Limiting the linear sample values to 12 magnitude bits, the A-law compression is defined by Equation 1, where A is the compression parameter (A=87.6 in Europe), and x is the normalized integer to be compressed.

$$y = \begin{cases} \frac{1+A*|x|}{1+ln(A)} & 0 \leq |x| \leq \frac{1}{A} \\ \frac{(sign(x))(1+A*|x|)}{1+ln(A)} & \frac{1}{A} \leq |x| \leq 1 \end{cases}$$

μ-Law Compander:
United States and Japan use μ-law companding. Limiting the linear sample values to 13 magnitude bits, the μ-law compression is defined by Equation 2, where μ is the compression parameter (μ =255 in the U.S. and Japan) and x is the normalized integer to be compressed.

$$y = \begin{cases} \frac{(sign(x))(1+\mu*|x|)}{1+ln(\mu)} & 0 \leq |x| \leq 1 \end{cases}$$

2.2 Code Snippets:

- A-law companding

  Given signal:

$$x(t) = 1.5 * cos(2\pi \times 5 \times 10^3 t) + sin(2\pi \times 4 \times 10^3 t); \quad (1)$$

```
clear all;
clc;
A=87.6;
Fs=100e3;
t=[0:1/Fs:255*1/Fs]; %63,127,255 for different number of samples
L=length(t);
x=1.5*cos(2*pi*t*5e3)+sin(2*pi*t*4e3);

%uniform quantization
V=max(abs(x));
bit_sig=1;
quant1 = (floor(x*(2^(bit_sig-1))/V))*(V/(2^(bit_sig-1)));

bit_sig=2;
quant2 = (floor(x*(2^(bit_sig-1))/V))*(V/(2^(bit_sig-1)));




for i=1:1:length(x)
    if abs(x(i))<1/A
```

```
        y(i)=sign(x(i))*A*abs(x(i))/(1+log(A));
    else
        y(i)=sign(x(i))*(1+log(A*abs(x(i))))/(1+log(A));
    end
end
MSE=0;
for i=1:1:length(x)
    if abs(y(i))<1/(1+log(A))
        z(i)=sign(y(i))*abs(y(i))*(1+log(A))/A;
    else
        z(i)=sign(y(i))*exp(abs(y(i))*(1+log(A))-1)/A;
    end
    MSE=MSE+(z(i)-x(i))^2;
end

figure(1);
plot(t,x);
title('Input Signal');
xlabel('time(secs)');
ylabel('Amplitute');

figure(2);
plot(t,y);
title('Compressed Signal');
xlabel('time(secs)');
ylabel('Amplitute');

figure(3);
plot(t,z);
title('Extracted Signal');
xlabel('time(secs)');
ylabel('Amplitute');

figure(4);
plot(sign(x).*abs(x), y);
title('Quantization of the signal');
xlabel('Signal Amplitude');
ylabel('Quantized Amplitude');

figure(5);
plot(t, quant1);
title('Uniform 1 bit quantization of the signal');
xlabel('time(s)');
ylabel('Quantized Amplitute');

figure(6);
plot(t, quant2);
title('Uniform 2 bits quantization of the signal');
xlabel('time(s)');
ylabel('Quantized Amplitute');




disp(MSE);
```

- μ-law companding:

  Given signal:

$$x(t) = 3 * cos(2\pi \times 5 \times 10^3 t) + 2 * sin(2\pi \times 4 \times 10^3 t); \quad (2)$$

```
clear all;
u=255;
Fs=100e3;
t=[0:1/Fs:255*1/Fs]; %63,127,255 for different number of samples
L=length(t);
x=3*cos(2*pi*t*5e3)+2*sin(2*pi*t*4e3);

%uniform quantization
V=max(abs(x));
bit_sig=1;
quant1 = (floor(x*(2^(bit_sig-1))/V))*(V/(2^(bit_sig-1)));

bit_sig=2;
quant2 = (floor(x*(2^(bit_sig-1))/V))*(V/(2^(bit_sig-1)));

%k=sqrt(3^2 + 2^2);
y=sign(x).*log(1+u.*abs(x))/log(1+u);

z=sign(y).*((1+u).^abs(y)-1)/u;

MSE=sum((z-x).^2);

figure(1);
plot(t,x);
title('Input Signal');
xlabel('time(secs)');
ylabel('Amplitute');

figure(2);
plot(t,y);
title('Compressed Signal');
xlabel('time(secs)');
ylabel('Amplitute');

figure(3);
plot(t,z);
title('Extracted Signal');
xlabel('time(secs)');
ylabel('Amplitute');

figure(4);
plot(sign(x).*abs(x), y);
title('Quantization of the signal');
xlabel('Signal amplitude');
ylabel('Quantized Amplitute');

figure(5);
plot(t, quant1);
title('Uniform 1 bit quantization of the signal');
xlabel('time(s)');
ylabel('Quantized Amplitute');

figure(6);
plot(t, quant2);
title('Uniform 2 bits quantization of the signal');
xlabel('time(s)');
ylabel('Quantized Amplitute');


disp(MSE);
```

2.3 Figures and Plots:

2.3.1 A-law Compander:

A=87.6
MSE observed=1.1285e-28

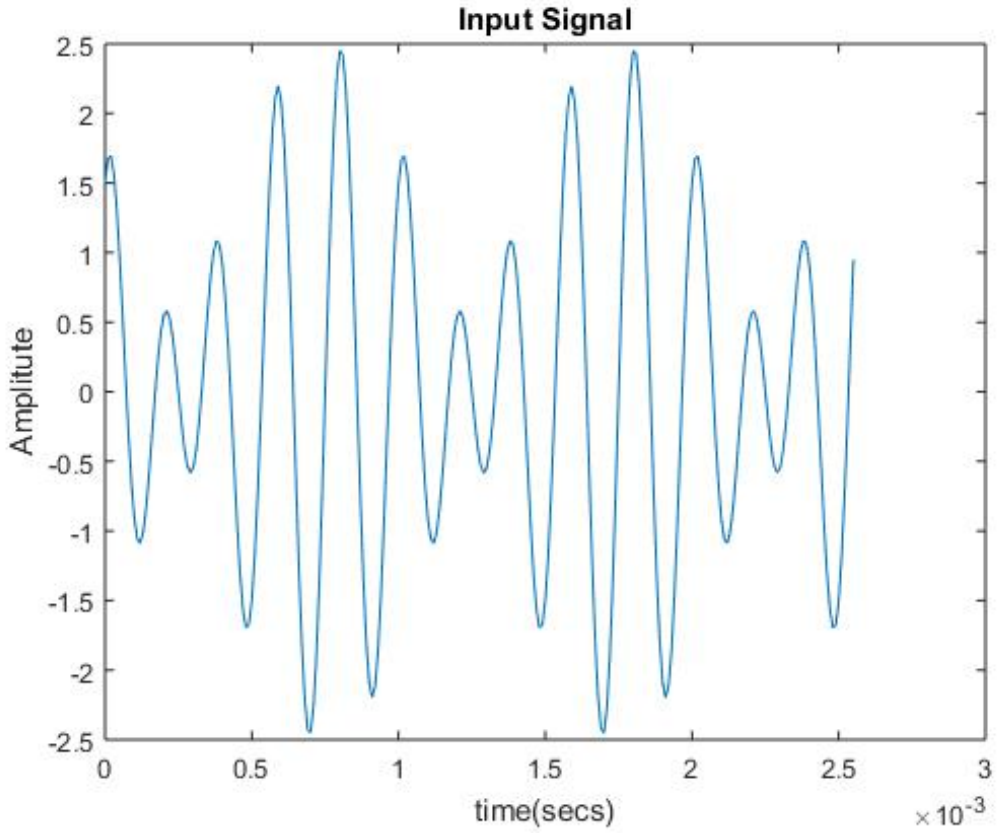

Figure 1: Input Signal

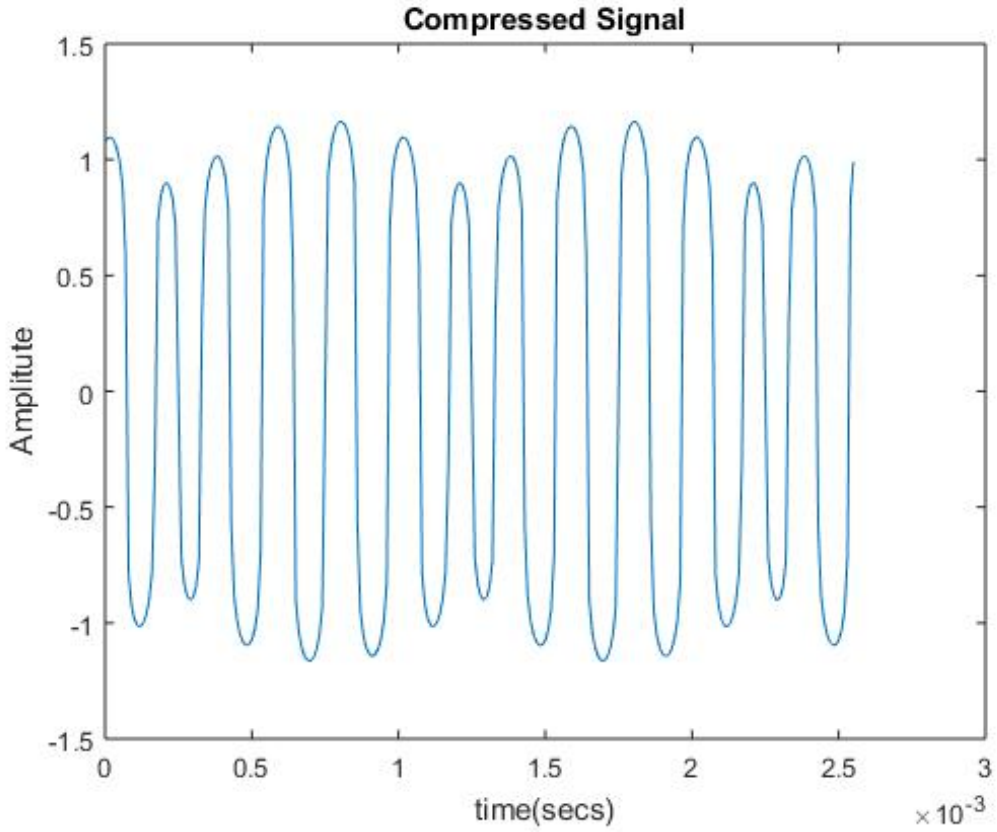

Figure 2: Compressed Signal

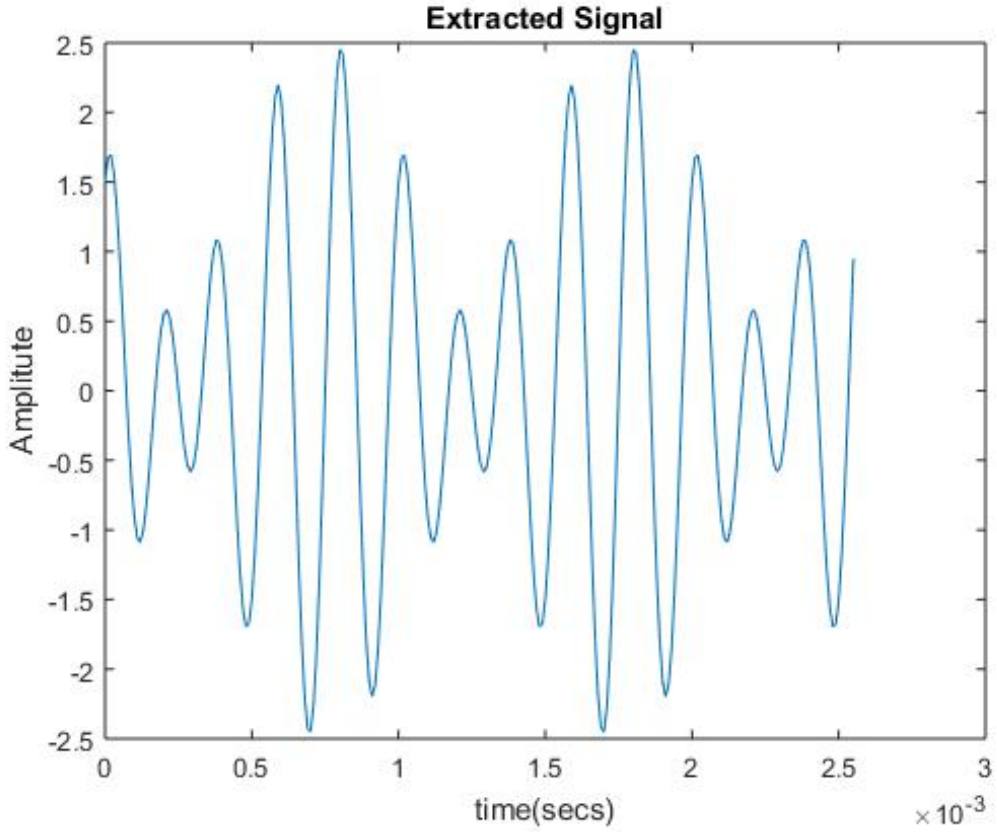

Figure 3: Extracted Signal

The following plot between input amplitude and quantized amplitude shows how, signals with lower amplitude are quantized with better accuracy than those of higher amplitudes. The quantized

amplitudes saturate at higher input amplitudes. And for, lower amplitude till a specified range, the quantization is linear, as it can be seen from the plot.

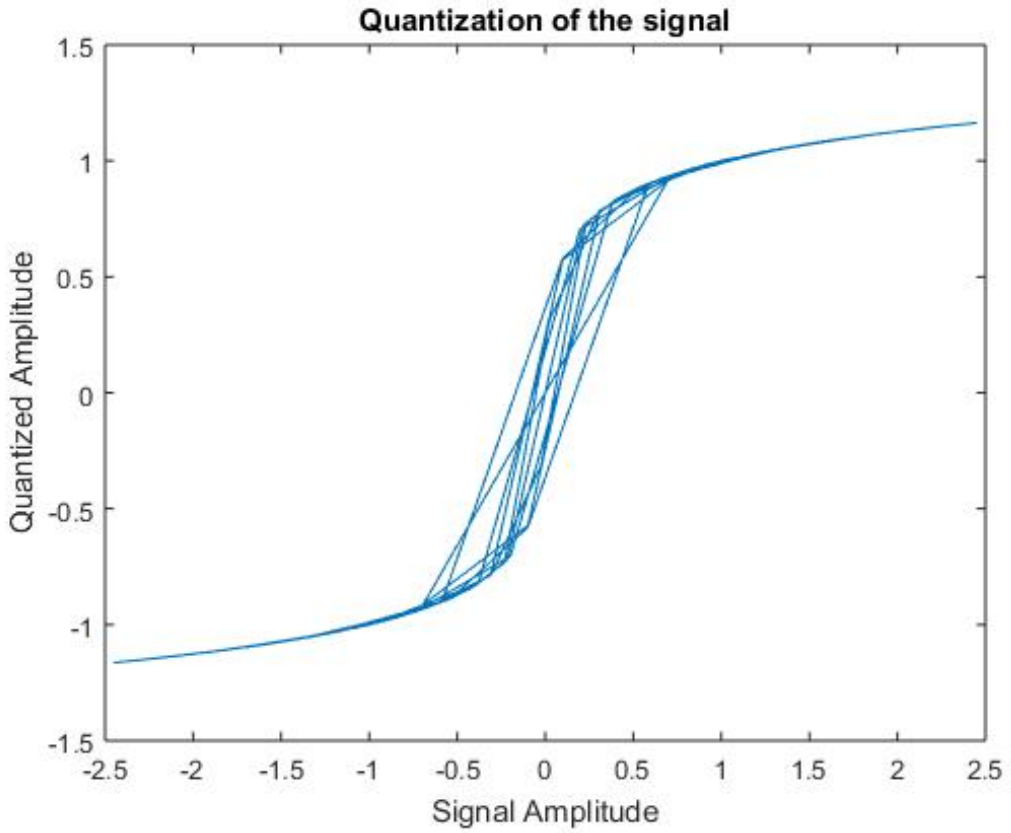

Figure 4: Quantized amplitude vs. Signal Amplitude

When compared to uniform quantization:

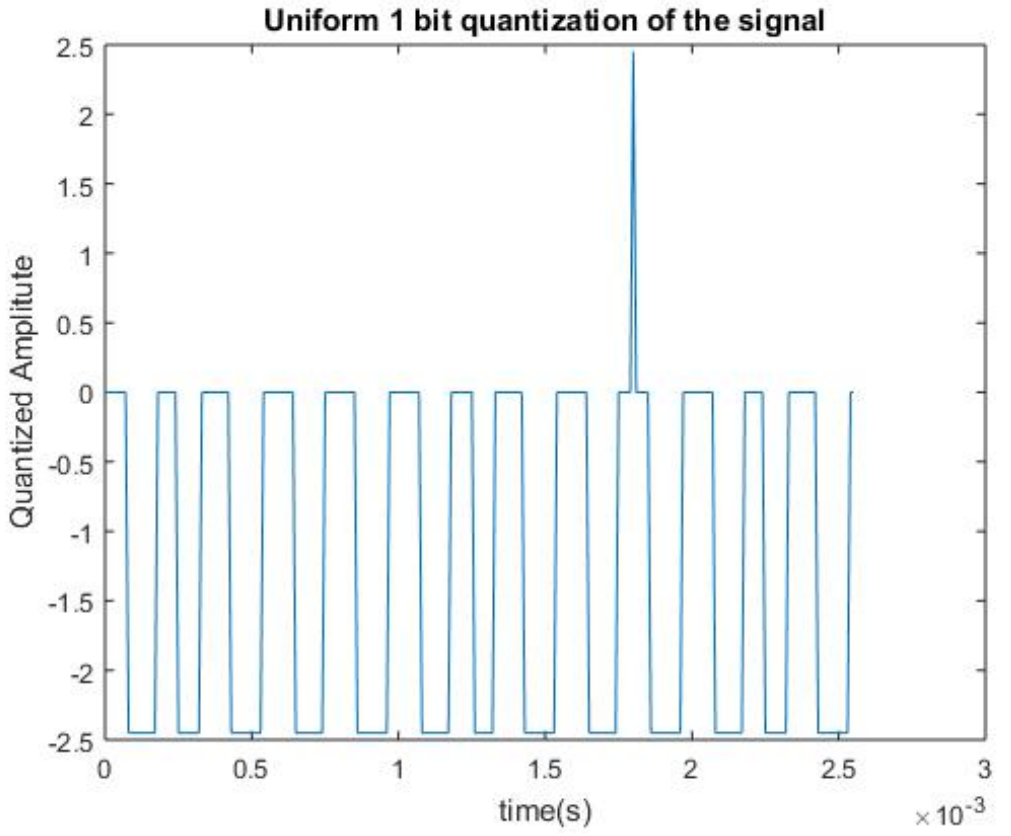

Figure 5: Uniform 1-bit quantized signal

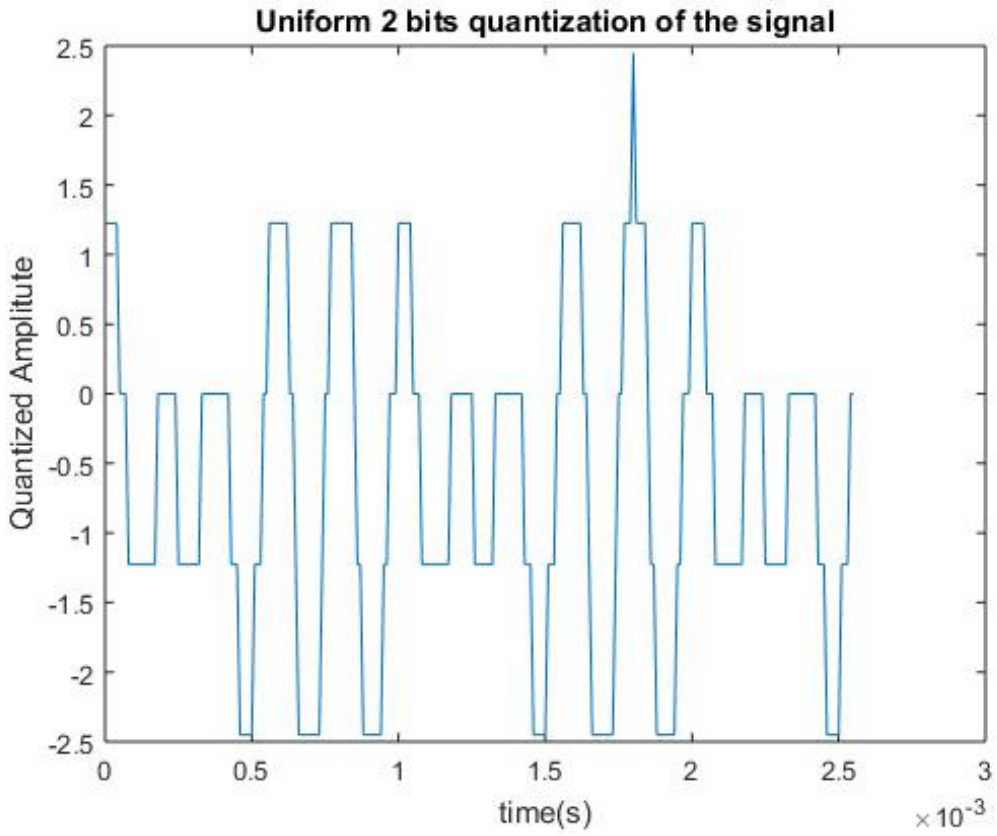

Figure 6: Uniform 2-bit quantized signal

2.3.2 μ-law compander:

μ=255
MSE observed: 3.7014e-28

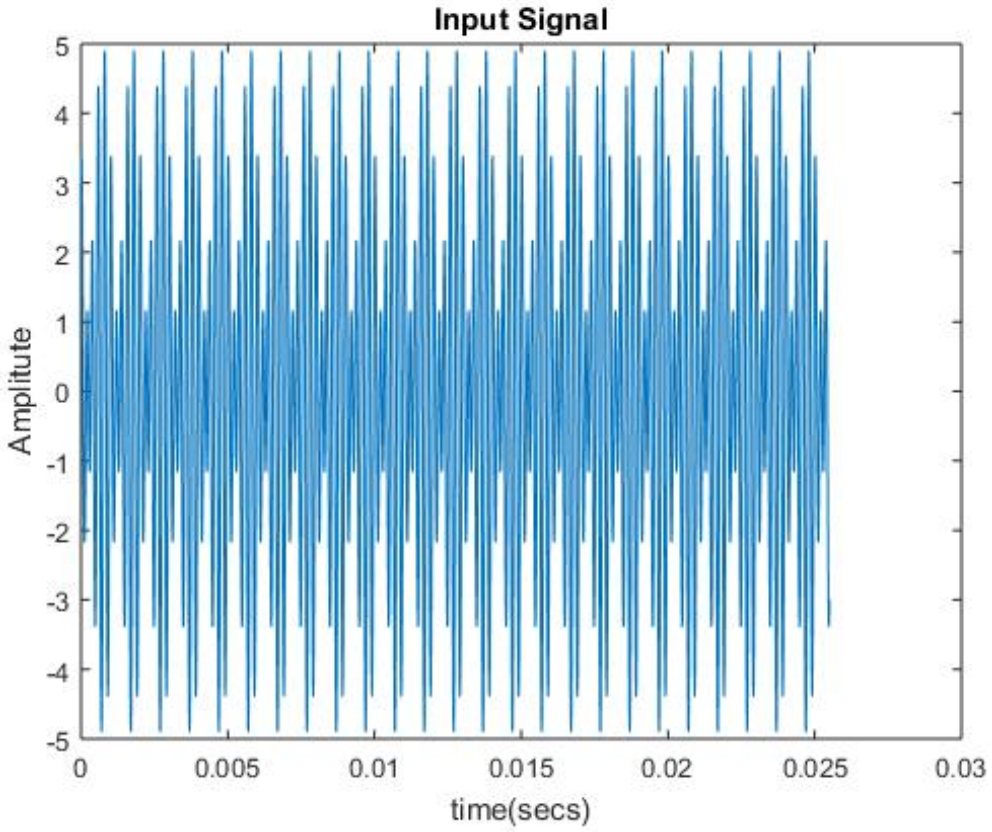

Figure 7: Input Signal

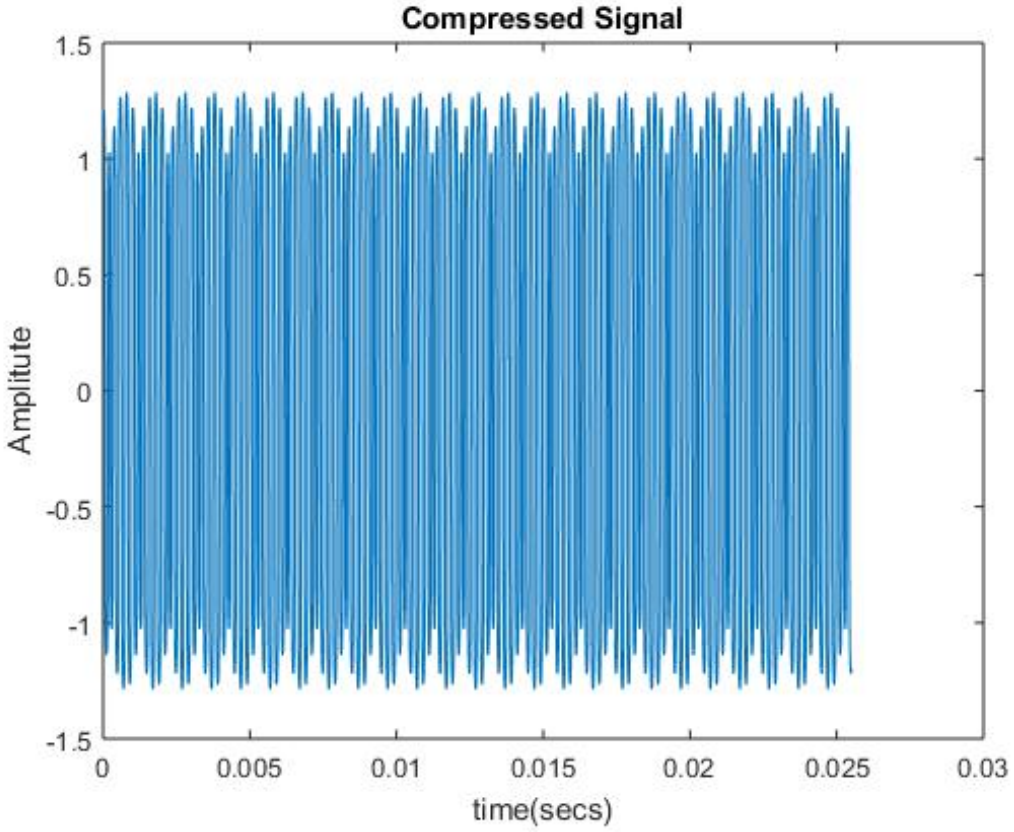

Figure 8: Compressed Signal

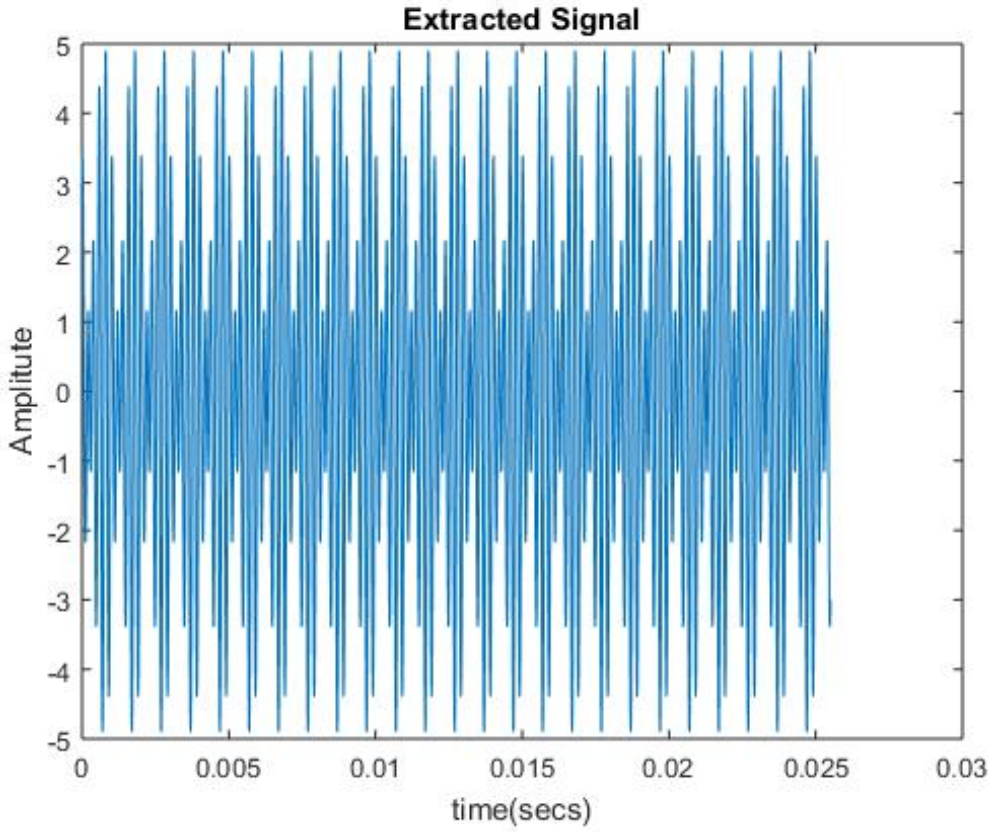

Figure 9: Extracted Signal

The following plot between input amplitude and quantized amplitude shows how, signals with lower amplitude are quantized with better accuracy than those of higher amplitudes. The quantized amplitudes saturate at higher input amplitudes.

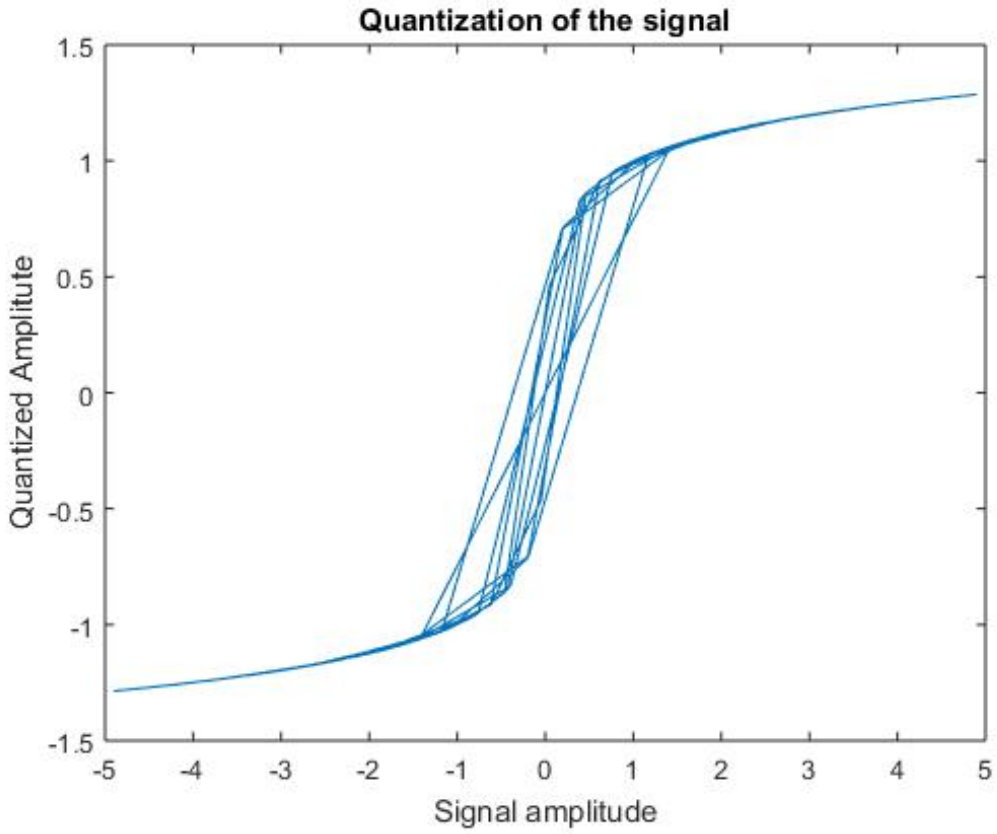

Figure 10: Quantized amplitude vs. Signal Amplitude

When compared to uniform quantization:

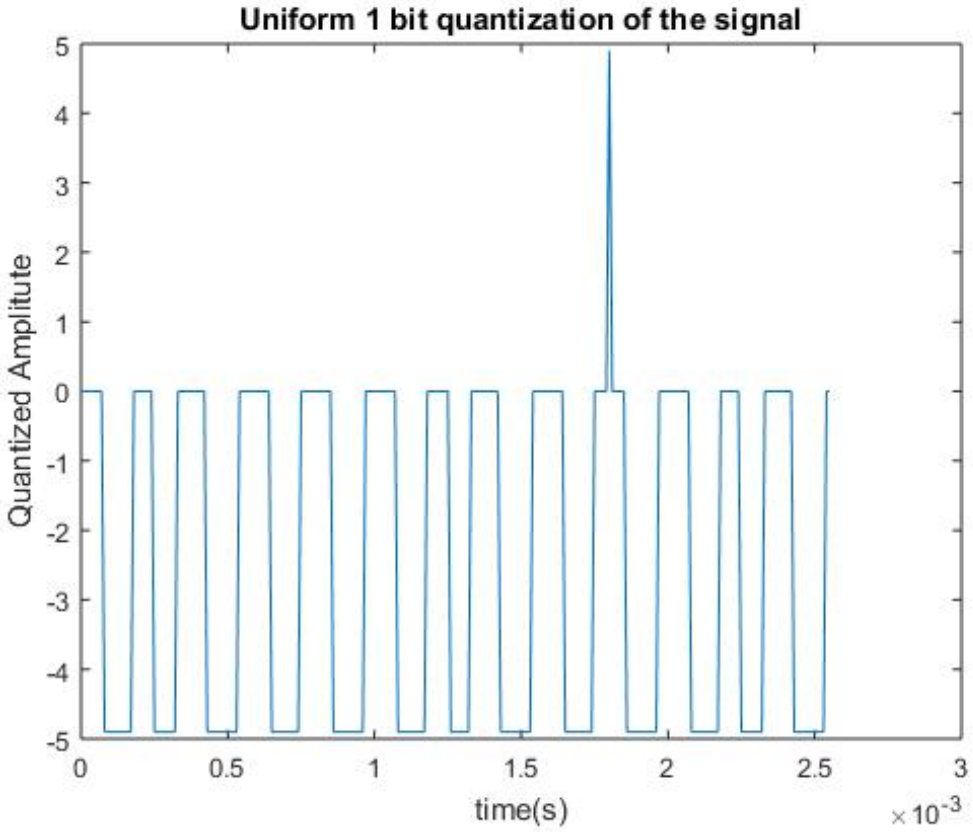

Figure 11: Uniform 1-bit quantized signal

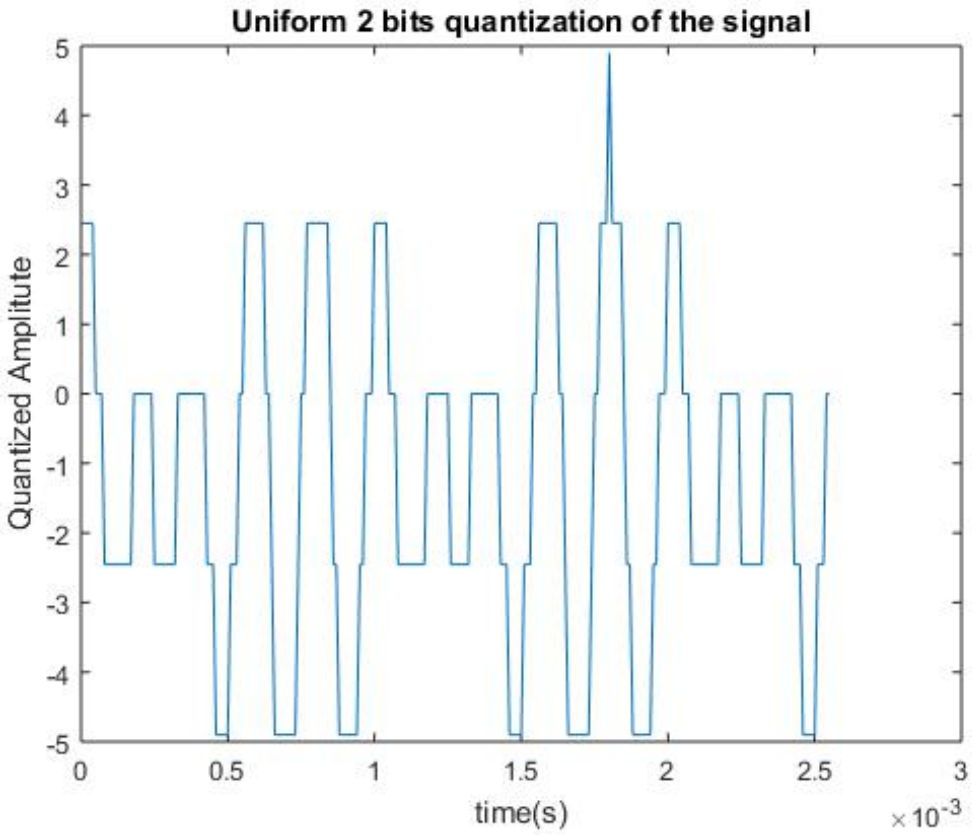

Figure 12: Uniform 2-bit quantized signal

2.4 Discussions:

- A-law and μ-law are two algorithms for image quantization which provide higher resolution for lower signal amplitudes, considering their higher probability of occurrence.
- μ-law has more dynamic range than A-law quantization. A-law does linear quantization to some extent in the lower amplitudes.
- Thus, μ-law distorts signals more than A-law owing to the existence of linear region in A-law quantization. And, μ-law encoders operate on linear 13-bit magnitude data, whereas 12-bit magnitude data is used in A-law.
- The outputs have also been compared to those of uniform quantization. In uniform quantization, we divided the entire spectrum into equally spaced intervals. Though, uniform quantization is quite straight forward, it is not optimal.
- We see overshoot in 1-bit and 2-bit quantization of the image when the input amplitude exceeds the maximum or the minimum quantized amplitude.

3 Part(b): Study the effects of dithering in quantization on signals

3.1 Theory:

Dither is an intentionally applied form of noise used to randomize quantization error, preventing large-scale patterns such as color banding in images. Dither is routinely used in processing of both digital audio and video data, and is often one of the last stages of mastering audio to a CD.

Dither is a form of noise, "erroneous" signal or data which is intentionally added to sample data for the purpose of minimizing quantization error. It is utilized in many different fields where digital processing is used, such as digital audio and images. The quantization and re-quantization of digital data yields error. If that error is repeating and correlated to the signal, the error that results is repeating. In some fields, especially where the receptor is sensitive to such artifacts, cyclical errors yield undesirable artifacts. In these fields dither is helpful to result in less determinable distortions.

In, our experiment, we add white Gaussian noise to the 8-bit input signal, so as to retain information while quantization.

3.2 Code Snippets:

Given signal:

$$x(t) = 10 * cos(2\pi \times 10^3 t) + 3 * sin(2\pi \times 5 \times 10^3 t); \tag{3}$$

```
clear all;
clc;

Fs=100e3;
t=(0:1/Fs:127*1/Fs);
L=length(t);
x=10*cos(2*pi*t*1e3)+3*sin(2*pi*t*5e3);
xm=x/max(abs(x));
B=12;
B_Final=8;
if xm*2^(B-1)-floor(xm*2^(B-1))<0.5
    y_x=floor(xm*2^(B-1))/2^(B-1);
else
    y_x=ceil(xm*2^(B-1))/2^(B-1);
end

y=awgn(y_x,20*(B_Final-1)*0.301);

if y*2^(B_Final-1)-floor(y*2^(B_Final-1))<0.5
    z=floor(y*2^(B_Final-1))/2^(B_Final-1);
else
    z=ceil(y*2^(B_Final-1))/2^(B_Final-1);
end
audio=audioplayer(z,Fs);
play(audio);

figure(1);
plot(t,xm);
title('Normalised Input Signal');
xlabel('time(secs)');
ylabel('Amplitute');

figure(2);
plot(t,y_x);
title('Quantised Input Signal (12 bit)');
xlabel('time(secs)');
ylabel('Amplitute');

figure(3);
```

```
plot(t,y);
title('Input signal with addded white noise');
xlabel('time(secs)');
ylabel('Amplitute');

figure(4);
plot(t,z);
title('Quantized Signal with added white noise');
xlabel('time(secs)');
ylabel('Amplitute');
```

3.3 Figures and Plots:

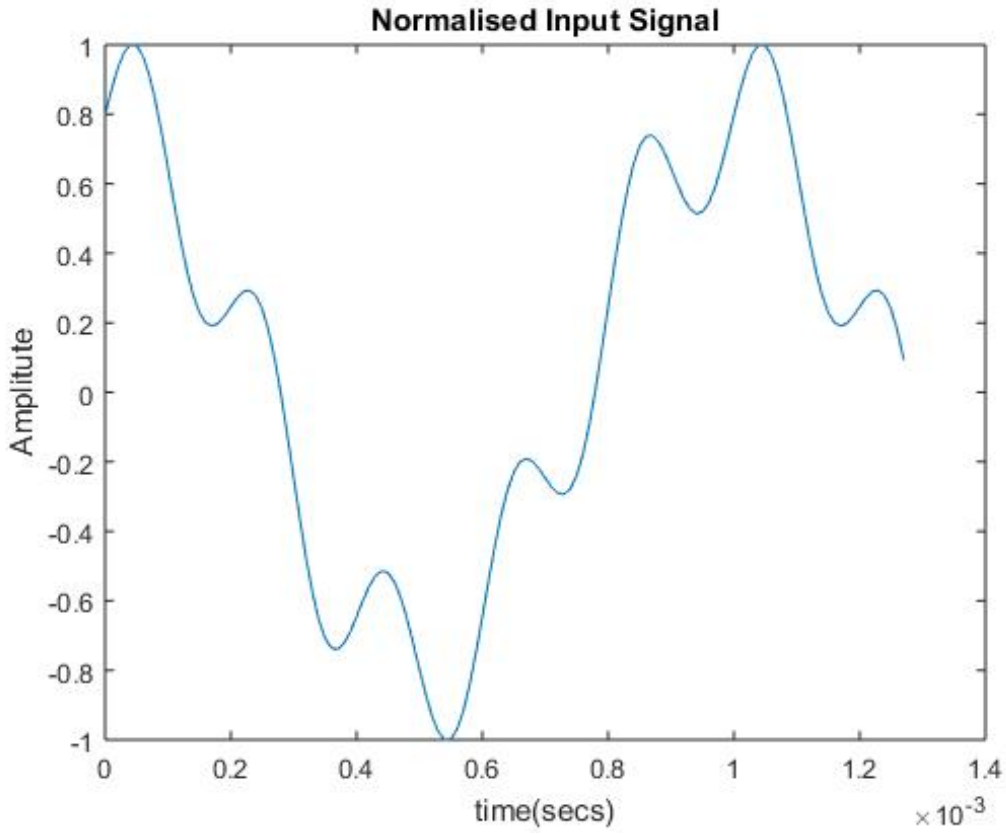

Figure 13: Input Signal

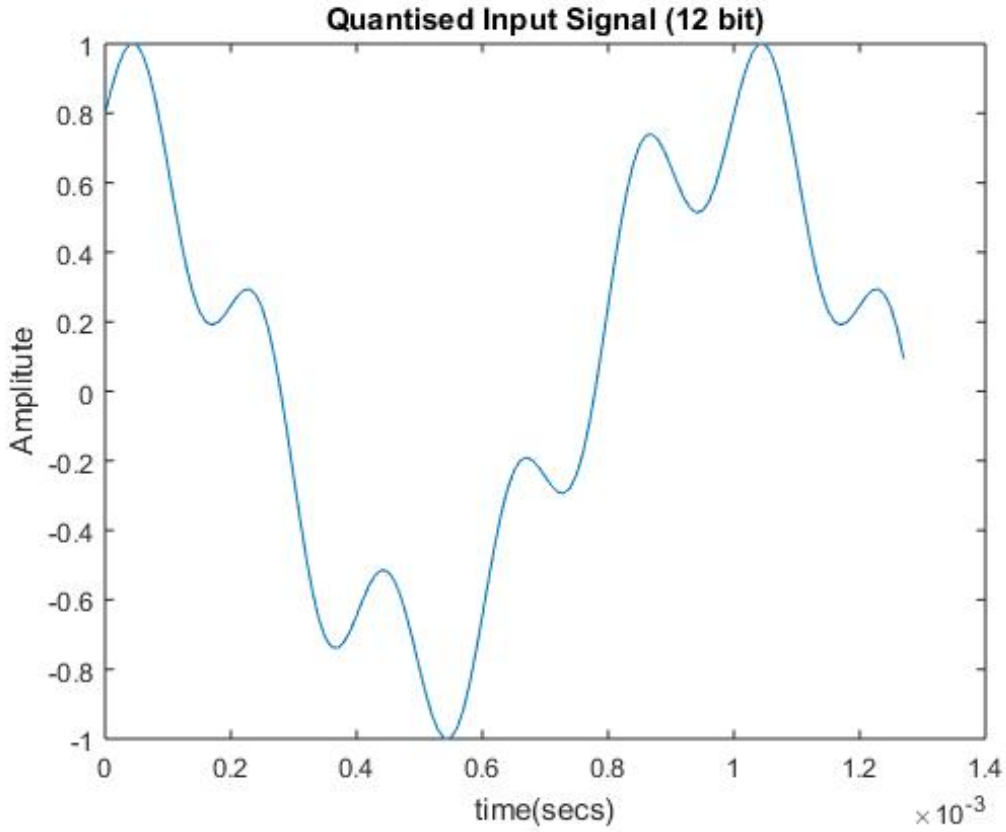

Figure 14: 12 bit quantized input

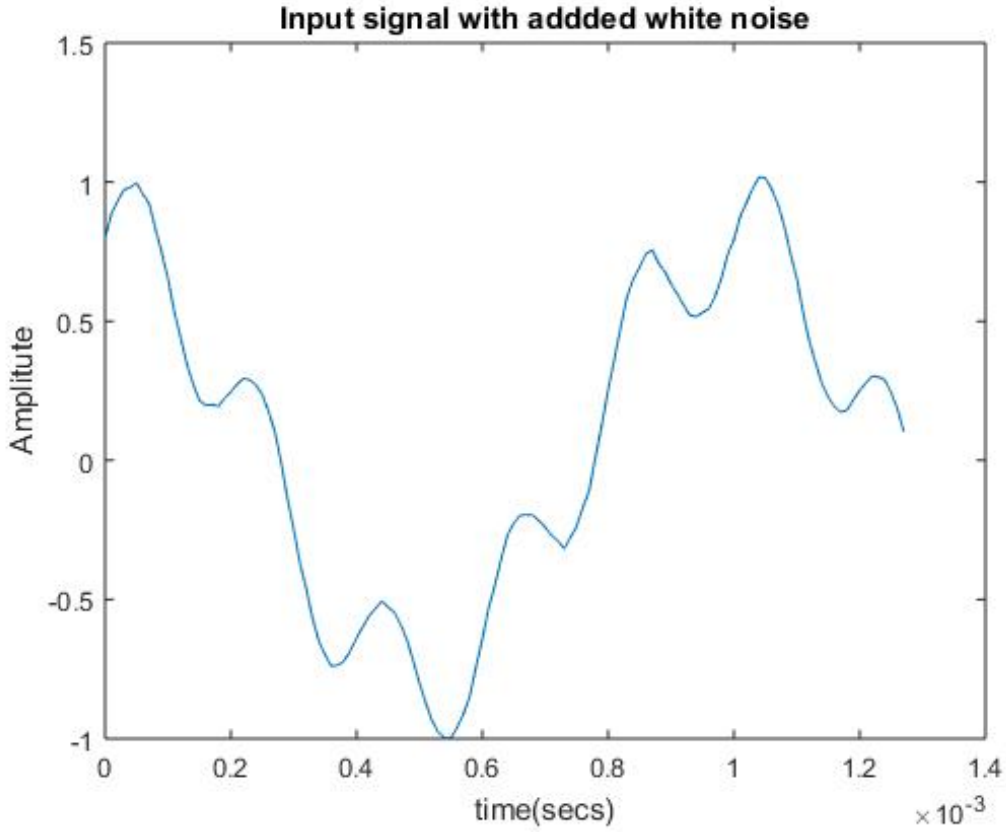

Figure 15: 8 bit input signal with AWGN

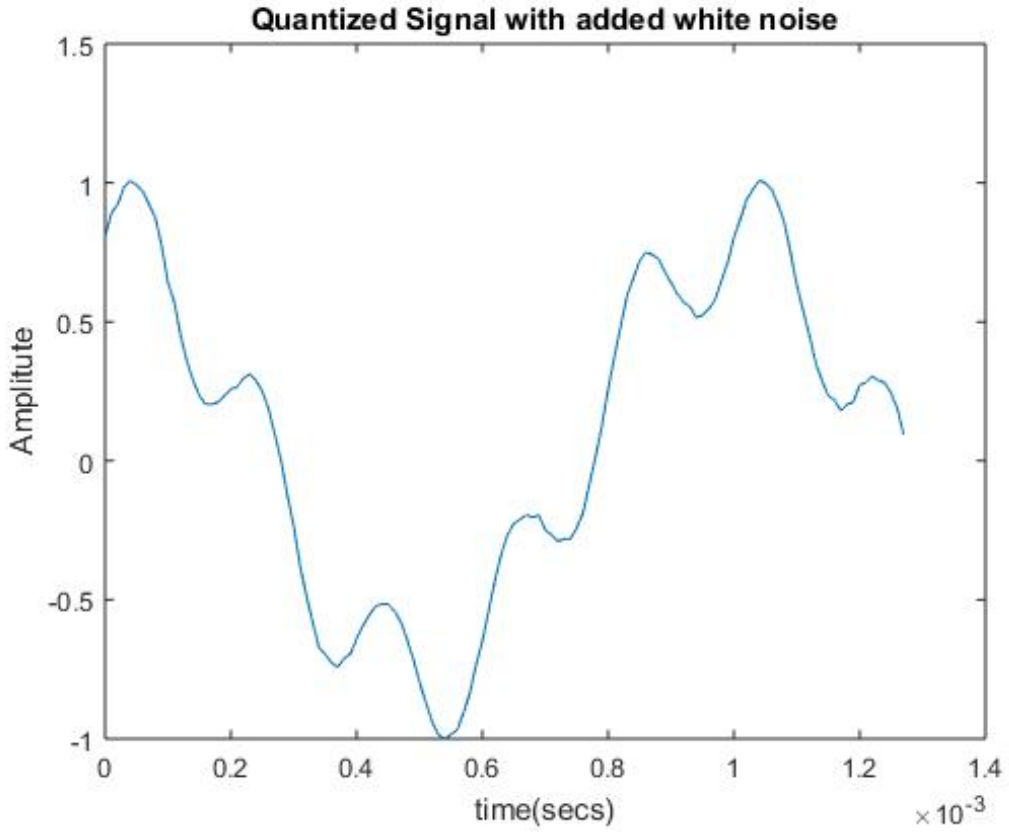

Figure 16: Quantized signal with the noise

3.4 Discussions:

- In dithering of signals or images, we intentionally, add noise to the input, so as to retain information when quantization is done.
- Generally, quantization leads to loss of information, but randomly added noise can reduce information loss to a great extent, and a more decipherable signal/image can be reconstructed from a dithered signal/image.
- The bits are truncated to 8 in order to add noise to the signal, the dithered audio is played.
- Dither should be added to any low-amplitude or highly periodic signal before any quantization or re-quantization process, in order to de-correlate the quantization noise from the input signal and to prevent non-linear behavior (distortion); the lesser the bit depth, the greater the dither must be. The result of the process still yields distortion, but the distortion is of a random nature so the resulting noise is, effectively, de-correlated from the intended signal. Any bit-reduction process should add dither to the waveform before the reduction is performed.

4 Part (c): Study Lloyd-Max Optimised Quantisation of a grayscale image, plot histograms and compare uniform and non-uniform quantization of images

4.1 Theory:

Lloyd-Max algorithm:

1. Choose initial quantization levels;
2. Assign points to a quantization level and reconstruct image;
3. Compute the new quantization levels as the mean of the value of all points assigned to each quantization level.
4. Go back to 2 until reduction of MSE is minimal.

4.2 Code Snippets:

- Non-uniform quantization of signal

```
clear all;
x = rgb2gray(imread('lena.ppm'));
s=size(x);
bits=4;
p=2^bits;
y=0;
intensity(1:256)=0; %keeps count of intensity level
for i=1:1:s(1)
    for j=1:1:s(2)
        intensity(x(i,j)+1)=intensity(x(i,j)+1)+1;
    end
end

figure(1);
histogram(x);
title('Histogram of the input image');
j=1;
z(1:p)=0;
bin(1:p+1)=255;
error=0;
% intensity(25)=0;
for i=1:1:255
    if abs(error+z(j)-s(1)*s(2)/p)<abs(error+z(j)+intensity(i)-s(1)*s(2)/p)
        bin(j)=i;
        j=j+1;
        error=z(j)-s(1)*s(2)/p;
    end
    z(j)=z(j)+intensity(i);
end

for i=1:1:s(1)
    for j=1:1:s(2)
        for k=1:1:p
            if ((x(i,j)>=bin(k))&&(x(i,j)<bin(k+1)))
                y(i,j)=bin(k);
            end
            if x(i,j)==bin(p+1)
                y(i,j)=bin(p+1);
            end
       end
    end
end
y=uint8(y);
```

```
figure(2);
histogram(y);
title('Histogram of the unequally quantized image output');
imwrite(y,'DFTBA_Unequal_Quantization.jpg');
```

- Llyod-Max quantization of a signal along with comparision with 1 bit uniform quantization

```
clear all;
clc;
image =(rgb2gray(imread('lena.ppm')));
x=double(image);
s=size(x);
B=4;

for i=1:1:s(1)
    for j=1:1:s(2)
        if image(i,j)<128
           U(i,j)=63;
        else if image(i,j)>127 && image(i,j)<256
           U(i,j)=192;
            end
        end
    end

end
a(1:1:2^B)=2^(8-B):2^(8-B):2^8;
b(1)=0;
b(2^B+1)=255;
for i=1:1:length(a)-1
    b(i+1)=(a(i)+a(i+1))/2;
end

 for k=1:1:5
     for r=1:1:2^B
        sum_a=0;
        count=0;
        for i=1:1:s(1)
            for j=1:1:s(2)
                if x(i,j)>=b(r) && x(i,j)<b(r+1)
                    sum_a=sum_a+x(i,j);
                    count=count+1;
                end
            end
        end
        a(r)=floor(sum_a/count);
    end
    b(1)=0;
    b(2^B+1)=255;
    for i=1:1:length(a)-1
        b(i+1)=(a(i)+a(i+1))/2;
    end
 end

a=floor(a);
b=floor(b);
MSE=0;

for i=1:1:s(1)
    for j=1:1:s(2)
        for r=1:1:2^B
            if x(i,j)>=b(r) && x(i,j)<b(r+1)
```

```
                    y(i,j)=a(r);
                    MSE=MSE+(y(i,j)-x(i,j))^2;
                end
            end
            if x(i,j)==b(2^B+1)
                y(i,j)=a(2^B);
                MSE=MSE+(y(i,j)-x(i,j))^2;
            end
        end
    end
y=uint8(y);
imwrite(y,'DFTBA_Unequal_Quantization_Max_Lloyd.jpg');

figure(1);
histogram(x);
title('Histogram of the input image');



figure(2);
histogram(y);
title('Histogram of the quantized image');



figure(3);
histogram(U);
title('Histogram of 1 bit uniformly quantized image');

disp(MSE);
```

4.3 Figures and Plots:

4.3.1 Non-uniform quantization of the image:

Figure 17: Input Image

Figure 18: Non-uniformly quantized output image

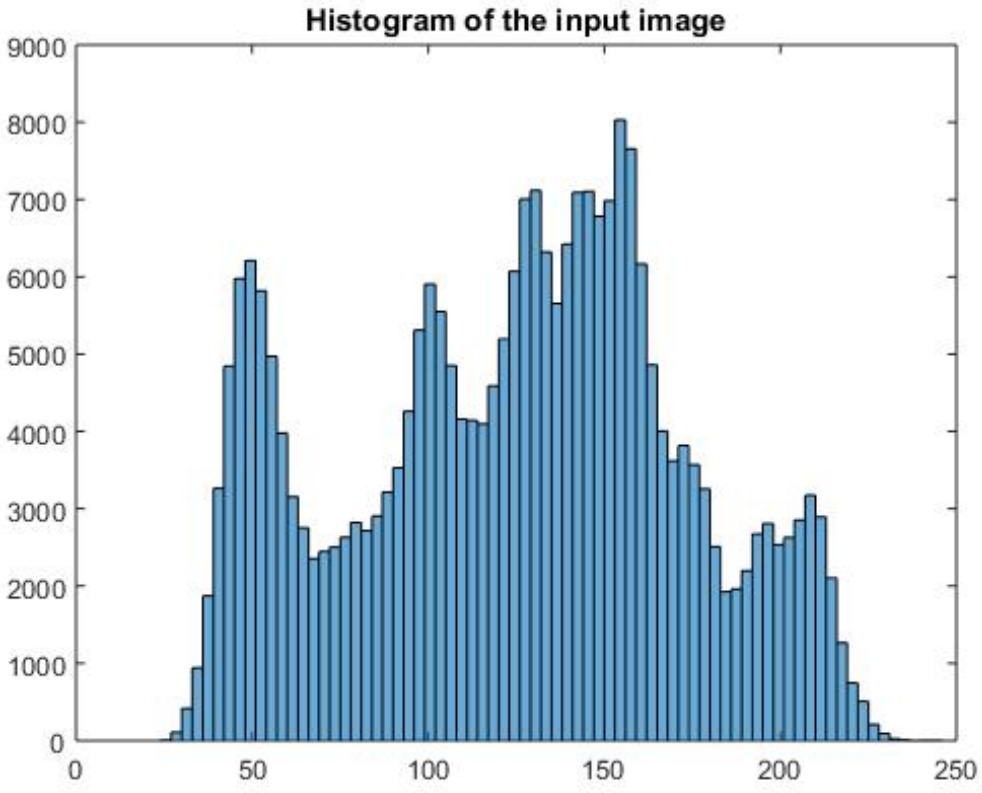

Figure 19: Histogram of the input image

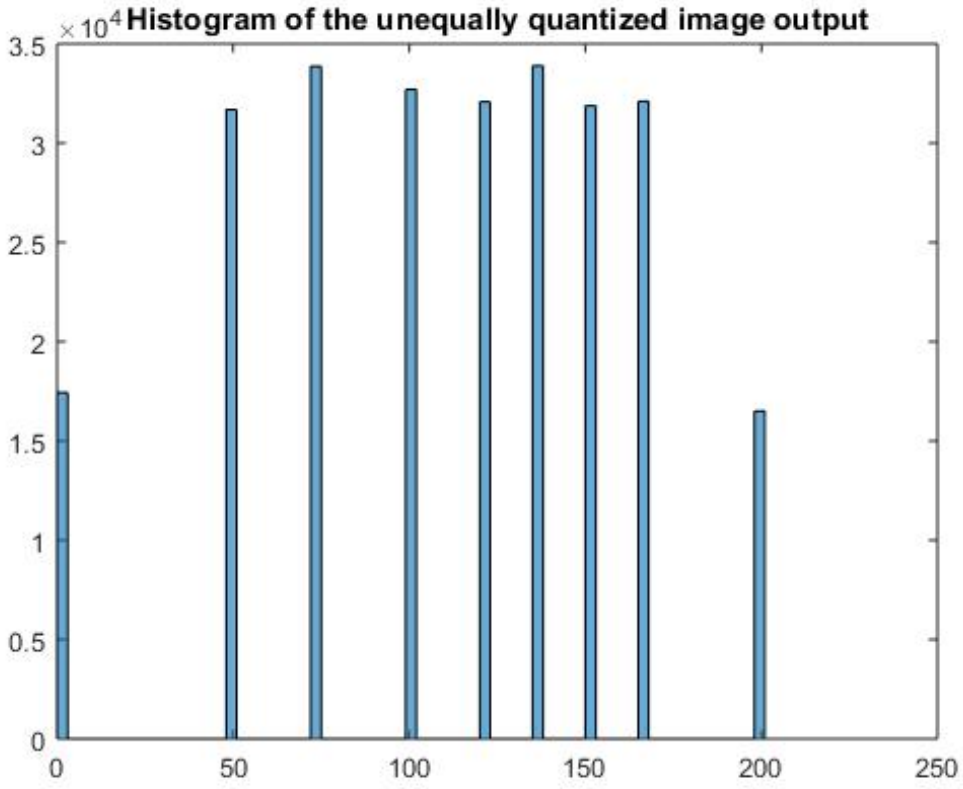

Figure 20: Histogram of the non-uniformly quantized image

4.3.2 Llyod-Max Quatization of an image:

Figure 21: Input Image

Figure 22: Llyod-Max quantized output image

Figure 23: 1-bit quantized image(for comparision)

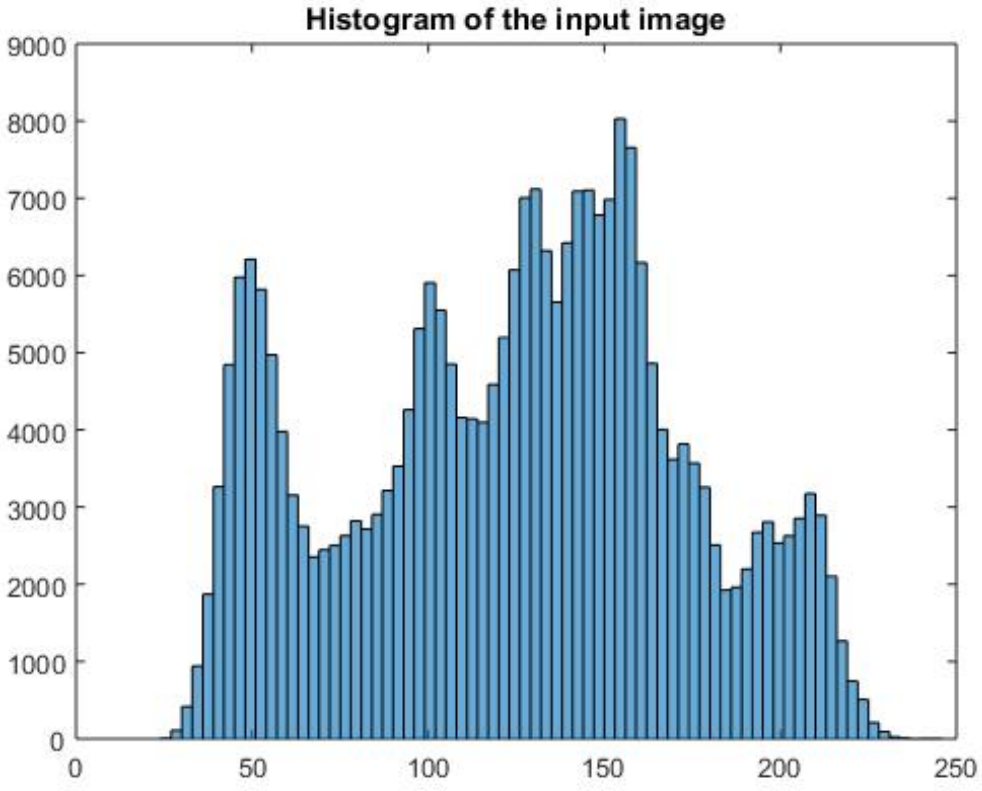

Figure 24: Histogram of the input image

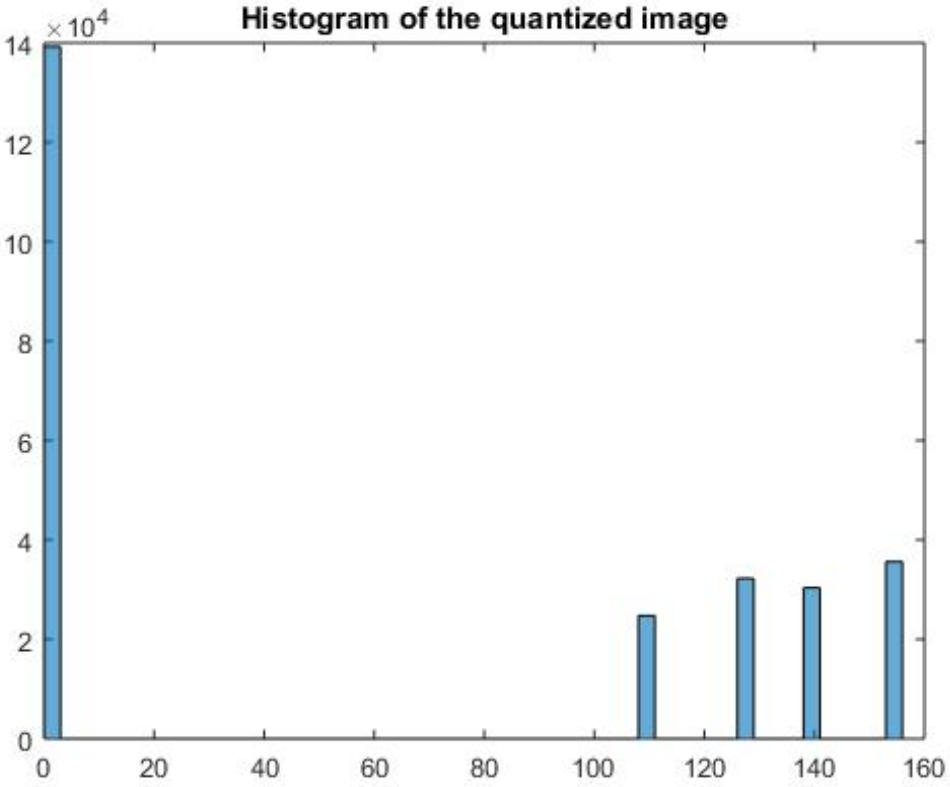

Figure 25: Histogram of the Max-Lloyd quantized image

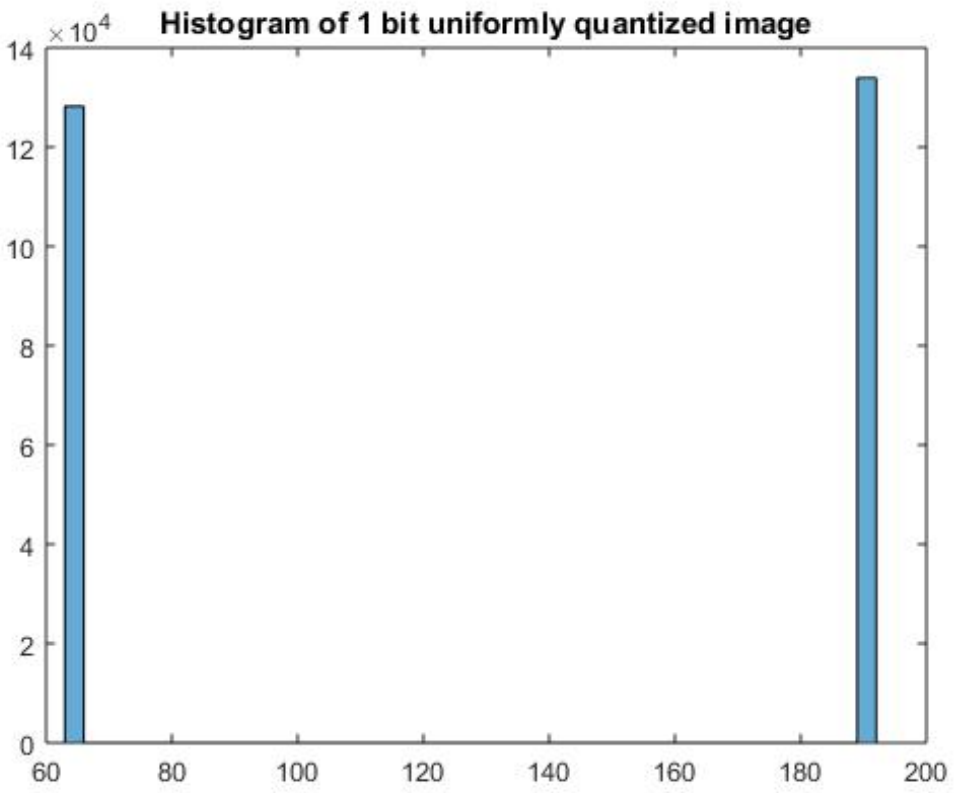

Figure 26: Histogram of the 1-bit quantized image(for comparision)

4.4 Discussions:

- Llyod-Max algorithm is an optimized quantization algorithm, and is used in this experiment to quantize a gray scale image.
- Optimum decision levels lie halfway between the optimum reconstruction levels, which, in turn, lie at the centre of the PDF in between the decision levels.
- It is desired to find optimum the decision levels tk and the reconstruction levels rk for an L-level quantizer such that the mean square error (MSE) (or quantization distortion) i's minimized.

5 Part (d): Study the effects of dithering in Image Processing

5.1 Theory:

Dithering is used in computer graphics to create the illusion of "color depth" in images with a limited color palette - a technique also known as color quantization. In a dithered image, colors that are not available in the palette are approximated by a diffusion of colored pixels from within the available palette. The human eye perceives the diffusion as a mixture of the colors within it (see color vision). Dithered images, particularly those with relatively few colors, can often be distinguished by a characteristic graininess or speckled appearance. In our experiment on images, similar to signals, we can add noise to image pixels before quantizing them, so as to retain more information.

5.2 Code Snippets:

```
clear all;
clc;
img=rgb2gray(imread('lena.ppm'));
s=size(img);
for i=1:1:s(1)
    error=0;
    for j=1:1:s(2)
        if img(i,j)+error>127
            y(i,j)=255;
        else
            y(i,j)=0;
        end
        error=(img(i,j)+error)-y(i,j);
    end
end
imwrite(y,'DFTBA_Dithered_Quantization.jpg');
```

5.3 Figures and Plots:

Figure 27: Input Image

Figure 28: Dithered image(Noise added to the original image)

5.4 Discussions:

- In dithering of signals or images, we intentionally, add noise to the input, so as to retain information when quantization is done. Generally, quantization leads to loss of information, but randomly added noise can reduce information loss to a great extent, and a more decipherable signal/image can be reconstructed from a dithered signal/image.
- Dithering is error diffusion, meaning it pushes (adds) the residual quantization error of a pixel onto its neighboring pixels, to be dealt with later.
- The diffusion coefficients have the property that if the original pixel values are exactly halfway in between the nearest available colors, the dithered result is a checkerboard pattern. For example, 50 per cent gray data could be dithered as a black-and-white checkerboard pattern.