Communication Theory Lab Report

Name: Aditya Sarkar
Roll No.: B19003
Branch: Electrical Engineering
Date: April 8, 2022
Communication Theory Lab
(EE304P, Spring-2022)

Lab Assignment # 6

Indian Institute of Technology Mandi, HP, India
School of Computing and Electrical Engineering (SCEE)

Quantization Techniques in Digital Communication

Abstract

In this experiment, we have tried to understand the concept of quantization in digital communication. Here we have only focused on uniform quantisation. We analysed the waveforms generated after applying quantization, by plotting their graphs in MATLAB. In all the techniques, we observed the quantized waveforms and have tried to explained them in this report.

1 Theory

In communication theory, quantization is defined as a way of mapping continuous values into a finite set of values depending on the number of bits being used to transmit data. Suppose we have 4 bits, then the number of levels of quantization will be $2^4 = 16$ levels. Quantization techniques are widely used for digital communication because of its improved transmission and information storage capacity. We have two types of quantization, but in this report, we will be focusing only on the uniform quantization.

There are two types of quantization techniques

1. Uniform Quantization
2. Non Uniform Quantization

Uniform Quantization is a way of quantization in which one tries to keep uniformly spaced quantization levels. Quantization always results in a loss of information. The difference between the quantized signal and the input signal is called quantization error. On increasing the number of levels will reduce the loss of information i.e. the quantization error. These levels are also called reconstruction level and the space between them is termed as step size.

1.1 Block Diagram

The block diagram is for quantization. In step 1, we convert an analog signal to discrete signal using sampling techniques. Analog signal has a continuous amplitude and time period while discrete signal has a continuous amplitude but discrete time period. In step 2, we convert an discrete signal to digital signal using quantization techniques. Digital signal has a discrete amplitude and a discrete time period.

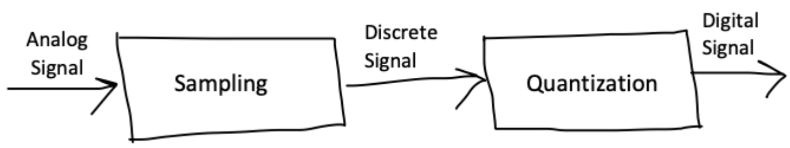

Figure 1: Quantization block diagram

1.2 Expected Outcome

The figures below are not drawn to scale.

1.2.1 Answer 1

This question was taken directly from what was taught in the lectures.

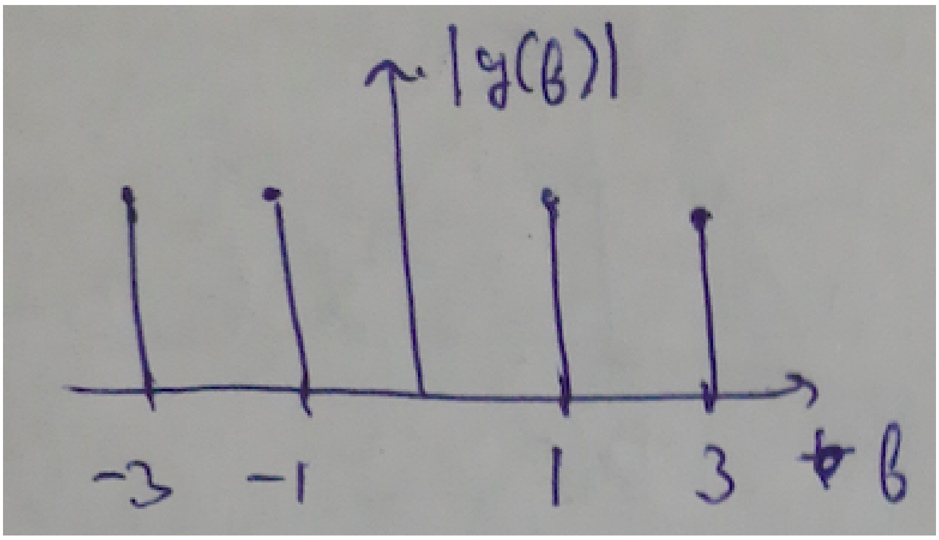

Figure 2: Figure for part A

1.2.2 Answer 2

This is directly taken from the lecture.

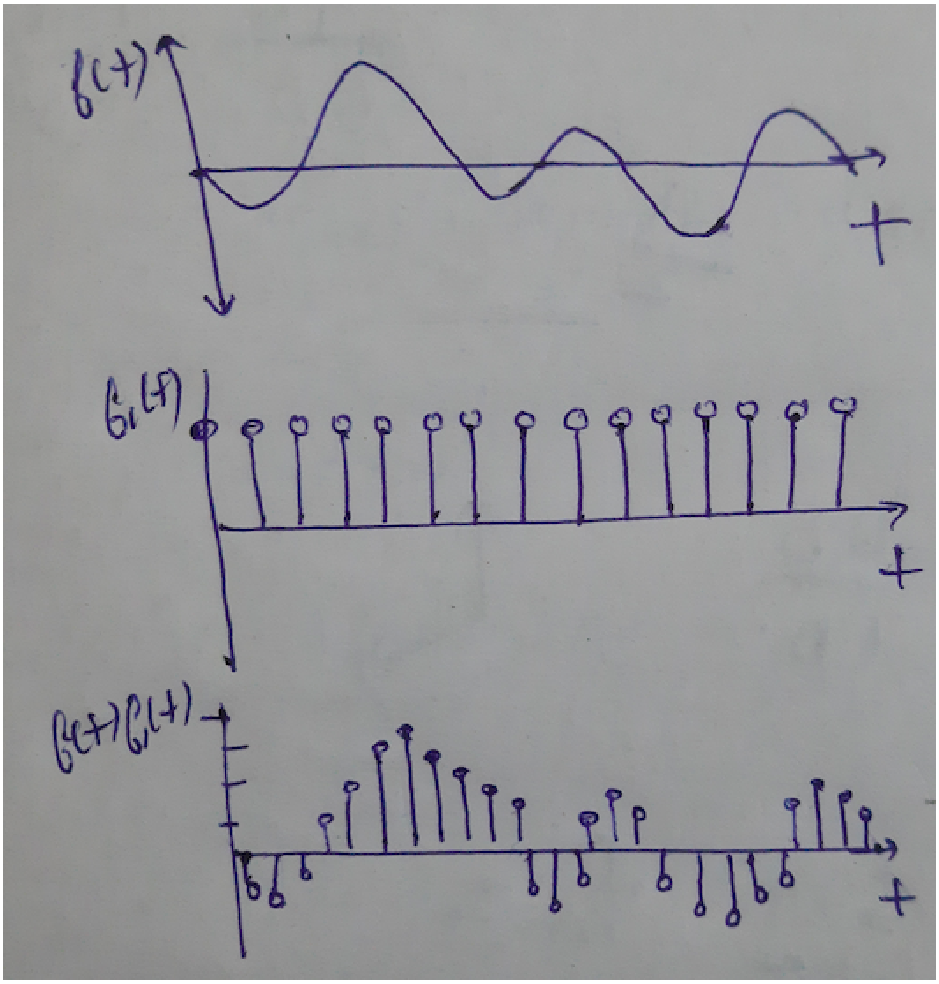

Figure 3: Figure for part B

1.2.3 Answer 3

This question was taking inference from what was taught in the lectures.

1.3 Application

Applications of quantization:

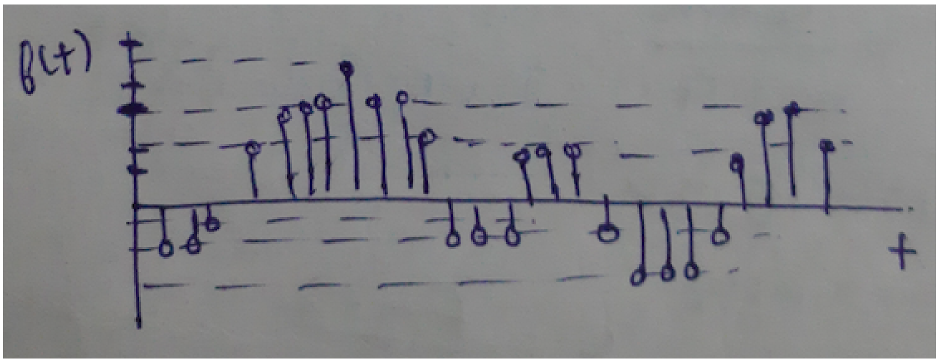

Figure 4: Figure for part C

1. `Control systems:` It is often used in low power micro-controllers. It helps produce an output signal given an input signal with a reasonable precision under low power constraints.

2. `Deep learning :` To reduce power consumption and memory and to accelerate the inferences, one performs quantization. 8 bit integers are used to make inferences, while having a decent accuracy and size of networks.

3. `Image compression:` Quantization is used in JPEG image compression.

2 Results and Inferences

2.1 Answer 1

We have,

$$g(t) = (sin(2*pi*1*t) – sin(2*pi*3*t)) \tag{1}$$

We have FT of Sin(w_0*t) as,

$$g(t) = \frac{\pi}{j} * [\delta(w - w_0) - \delta(w + w_0)] \tag{2}$$

Inferences are :

1. We can see that for in frequency domain, we have four peaks at the middle. This is because of :

(a) For one sine function, we get two impulses. In the question the equation that was a sum of two sine functions. Hence we will have 2x2 =4 peaks.

(b) we have taken fftshift. FFTshift is a method in MATLAB which shifts the zero-frequency component to the center of the spectrum. We can see that in the plot as well.

2. The amplitude of impulses is around 500 with those at -1Hz, 1Hz slightly lower and those at -3Hz, 3Hz slightly bigger.

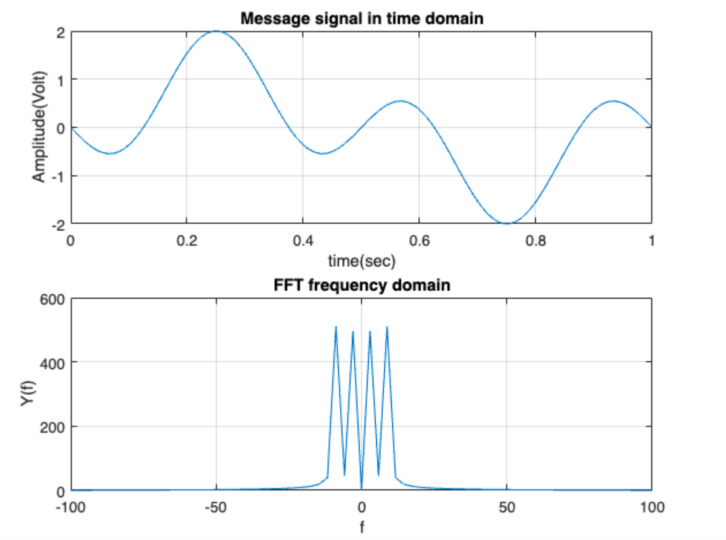

Figure 5: Figure for part A

2.2 Answer 2

We have 50 Hz as the frequency of impulse train. It means that it has a time period of 0.02 secs.

Inferences are :

1. We can see that the impulses in the impulse train are happening after 0.02 secs.
2. The message signal has got sampled after 0.02 secs for 5 signals are present in a time period of 0.1 secs.
3. We have followed the Nyquist theorem for sampling, hence the sampling is proper.
4. We can also consider it as Pulse Amplitude Modulation (PAM) as we can observe that the amplitude of pulse carrier is following the instantaneous amplitude of the message signal.

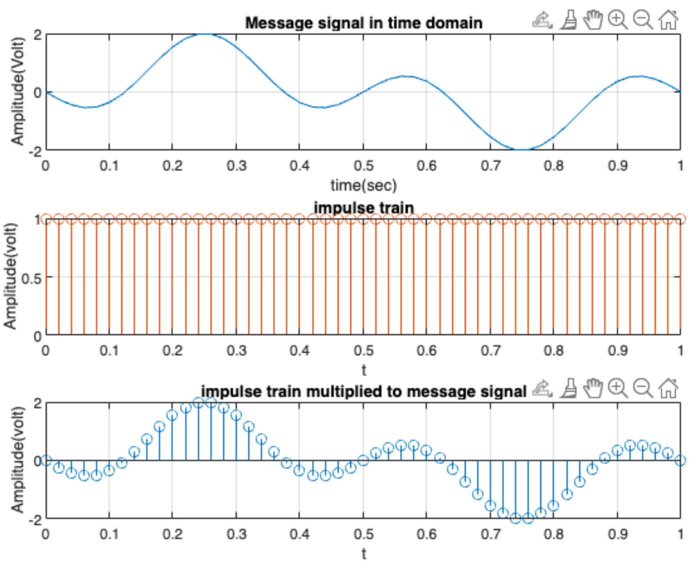

Figure 6: Figure for part B

2.3 Answer 3

We have quantized the discrete signal here into 16 levels. The peak amplitudes have got divided into 16 levels. In this the step size or quant will be 2*2/16 = 0.25V. This is uniform quantization as we can see that the step size is fixed or uniform.

Inferences are :

1. We can observe uniform quantization from the plot as the step size is constant and uniform.
2. Non-uniform quantisation can be a better option here because we can see smaller amplitude pulses in the waveform. Uniform quantization is more susceptible to adding more noise to the signal. Non-uniform distribution will better adapt by having smaller step sizes at low amplitudes and bigger step sizes at higher amplitudes.
3. Number of levels are 16. Thus $log_2(n) = 4$ bits are required for information transmission.

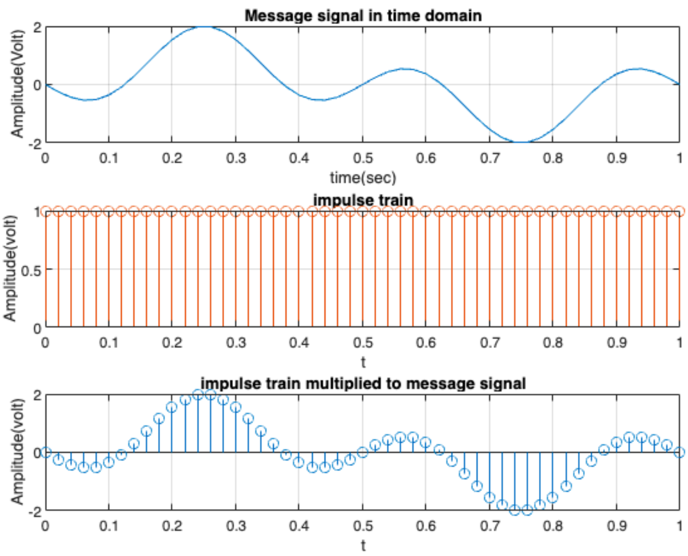

Figure 7: Figure 1 for part C

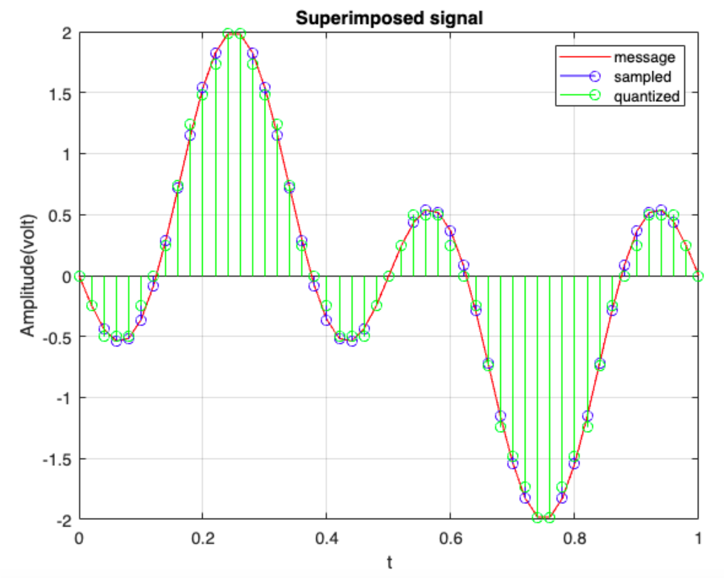

Figure 8: Figure 2 for part C

Appendix

A Matlab Commands

Table 1: Matlab commands used in this lab.

Matlab Command	**Function**
`plot(`x, y`)`	plots values of the simulation series y along the y-axis, with values of the simulation series x along the x-axis.
`figure()`	creates a new figure in MATLAB.
`title(x)`	adds a title x to the plot
`xlabel(x)`	adds a horizontal label x (along x axis) to the plot
`ylabel(x)`	adds a vertical label x (along y axis) to the plot
`grid on`	adds a grid to the plot.
`clc`	clears everything from the matlab command line window.
`linspace(x1,x2,p)`	generates p equally distant points between x1 and x2.
`subplot(abc)`	generates a subplot of size a x b, and the current image is of index c
`ones(length)`	creates an array of 1s.
`size`	gives length of an array

B Matlab Code

Matlab codes for each part.

B.1 Q1(a)

```
% question1(a)
clc;
close all;
```

```
% defining time period
Am =1;
fm1 = 1;
fm2 = 3;
t = 0:1/999:1;
N= length(t);
N = 2^ceil(log2(N));
f=(-N/2:N/2-1) / ((N/3)*1/999);

% message signal
ym = Am*(sin(2*pi*fm1*t)-sin(2*pi*fm2*t));
figure();
subplot(2,1,1);
plot(t,ym);
title("Message signal in time domain");
xlabel("time(sec)");
ylabel("Amplitude(Volt)");
grid on;

%FT of message Signal
y_F=fftshift(fft(ym,N));
subplot(2,1,2);
plot(f,abs(y_F));
title("FFT frequency domain");
xlabel("f");
ylabel("Y(f)");
ylim([0 600]);
xlim([-100 100]);
grid on;
```

B.2 Q1(b)

```
% question1(b)
clc;
close all;

% message signal
Am =1;
fm1 = 1;
```

```
fm2 = 3;
t=0:0.02:1;
ym = Am*(sin(2*pi*fm1*t)-sin(2*pi*fm2*t));
figure(1);
subplot(3,1,1);
plot(t,ym);
title("Message signal in time domain");
xlabel("time(sec)");
ylabel("Amplitude(Volt)");
grid on;

% impulse train
t = 0:0.02:1;
y = ones(length(t));
subplot(3,1,2);
stem(t,y);
title("impulse train");
xlabel("t");
ylabel("Amplitude(volt)");
grid on;

% sampling
gs= ym.*y;
subplot(3,1,3);
stem(t,ym);
title("impulse train multiplied to message signal");
xlabel("t");
ylabel("Amplitude(volt)");
grid on;
```

B.3 Q1(c)

```
% question1(c)
clc;
close all;

% message signal
Am =1;
fm1 = 1;
```

```
fm2 = 3;
t = 0:0.02:1;

ym = Am*(sin(2*pi*fm1*t)-sin(2*pi*fm2*t));
figure(1);
subplot(3,1,1);
plot(t,ym);
title("Message signal in time domain");
xlabel("time(sec)");
ylabel("Amplitude(Volt)");
grid on;

%impulse train
t=0:0.02:1;
y = ones(length(t));
subplot(3,1,2)
stem(t,y);
title("impulse train");
xlabel("t");
ylabel("Amplitude(volt)");
grid on;

%sampled signal
gs= y.*ym;
subplot(3,1,3);
stem(t,ym);
title("impulse train multiplied to message signal");
xlabel("t");
ylabel("Amplitude(volt)");
grid on;

% superimposed signal
l = 16;
peak = max(abs(ym));
delta = 2*peak/16;

gkts = ym/delta;
gkts = round(gkts);
gkts = gkts*delta;

figure(2);
```

```
plot(t,ym,'r');
hold on;
stem(t,ym,'b');
hold on;
stem(t,gkts,'g');
legend('message','sampled','quantized');
title("Superimposed signal");
xlabel("t");
ylabel("Amplitude(volt)");
grid on;
```

References

[1] IIT Mandi lectures on EE304 offered by Dr Adarsh https://cloud.iitmandi.ac.in/d/4bb3a5f304334160ab67/

[2] Tutorialspoint lectures on quantisation https://www.tutorialspoint.com/digital_communication/digital_communication_quantization.htm

[3] Wikipedia notes on quantisation for signal processing https://en.wikipedia.org/wiki/Quantization_(signal_processing)

[4] Digital quantisation on Youtube https://youtu.be/cUdnGkymAWU