Communication Theory Lab Report

Name: Aditya Sarkar
Roll No.: B19003
Branch: Electrical Engineering
Date: April 24, 2022
Communication Theory Lab
(EE304P, Spring-2022)

Lab Assignment # 8

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

Signal Reconstruction at the Receiver End

Abstract

In this experiment, we have tried to understand the concept of reconstructing the message signal at the receiver from the transmitted waveform over a noiseless channel after sampling, quantization, and waveform coding. Here we have assumed that the channel is noiseless. We analysed the waveforms generated after applying reconstruction, by plotting their graphs in MATLAB.

1 Theory

In communication theory, reconstruction of signal means that we have to determine the original continuous signal from the sampled and quantized which has been transferred over the noiseless channel. Consider m(t) as the message signal, $M(\omega)$ be its CTFT, X_s be its sampled and quantized signal and $X_s(\omega)$ be its CTFT. Then we can extract $X_c(\omega)$ using the following equation -

$$X_c(\omega) = H(\omega).X_s(\omega)$$

Here $H(\omega)$ is a low pass filter. On frequency domain, $H(\omega)$ is a rectangle function so in time domain, it will become a sinc function. Also the above equation in time domain will become convolution, so we can rewrite it as -

$$x_c(t) = h(t) \circledast x_s(t)$$

$$x_c(t) = \Sigma x_s(kT_s).h(t - kT_s)$$

where h(t) is the sinc function.

$$h(t) = Sinc(2\pi B(t))$$

Replace the above equation to the RHS of the convolution equation and you will get the proper equation of reconstruction.

$$x_c(t) = \Sigma x_s(kT_s).Sinc(2\pi B(t - kT_s))$$

1.1 Block Diagram

The block diagram is for quantization and reconstruction. In step 1, we convert an analog signal to discrete signal using a sampling technique. 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.

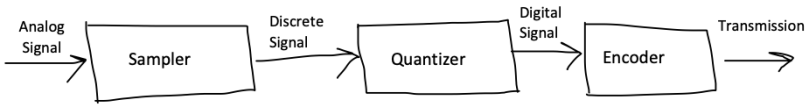

Figure 1: Quantization block diagram

The second block is for converting the quantized signal to the message signal. We will convolve the sinc function over the message signal to form the reconstructed signal.

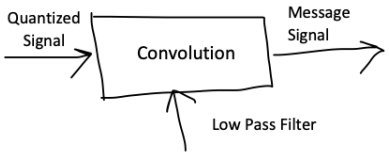

Figure 2: Reconstruction block diagram

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.

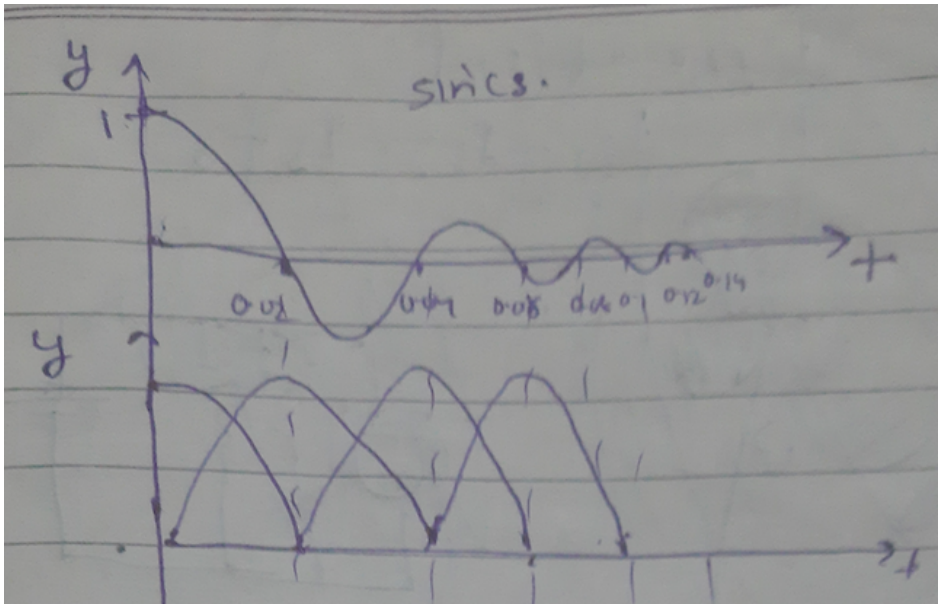

Figure 3: Figure for part A

1.2.2 Answer 2

This is directly taken from the lecture.

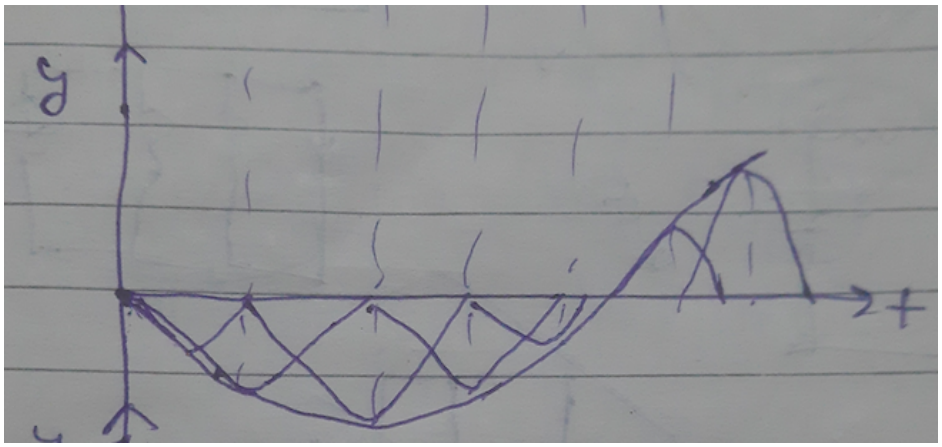

Figure 4: Figure for part B

1.2.3 Answer 3

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

Figure 5: Figure for part C

1.2.4 Answer 4

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

Figure 6: Figure for part D

1.3 Application

The broad concept of reconstruction is used in various fields such as :

1. `3D reconstruction in Electron Microscopy:` It is currently being used in reconstructing a three dimensional density function from a set of two dimensional electron microscopy (EM) images.

2. `Sensor Detection at PNNL :` It is used in sensor technologies which is aimed at detecting threats to national and personal security in Pacific Northwest National Laboratory (PNNL) in Washington State.

3. `Image and file decompression:` Reconstruction is used to decompress JPEG compressed images or compressed files/folders.

2 Results and Inferences

2.1 Answer 1

We have the following equation for the time shifted sinc pulses.

$$g(t) = \Sigma Sinc(2\pi B(t - kT_s)$$

Inferences are :

1. In first graph, we can observe a sinc pulse for time duration t = 0 to t = 0.2 sec. Just like a normal sinc graph, it is highest at origin and is decreasing in both positive and negative directions. As we move farther from the origin, the sinc is trying to converge itself to the horizontal axis.

2. In second graph, we can observe a time shifted version of sinc pulses. The sincs are shifted by a unit time which is 0.2/10 or 0.02 secs in our case. We have selected different colors to represent each plots clearly.

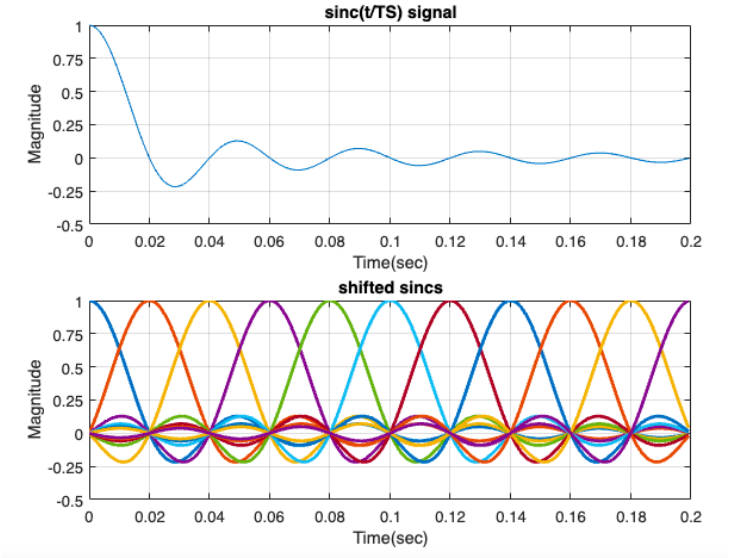

Figure 7: Figure 1 for part A

2.2 Answer 2

In this question, we are given that the sampling frequency fs = 50Hz, so 50 samples are present within the duration of 1 second. In other words, 10 samples are present in the duration of 0.2 seconds.

Inferences are :

1. We can observe that at every sampling time instant, a sample is multiplied to the sinc curve which is time shifted, so at that particular time instant, the sinc pulse is at its high. $g(t_k)$ = gs(kTs) * sinc((t-kTs)/Ts), for k $\in 0, 1, ..., 10$.

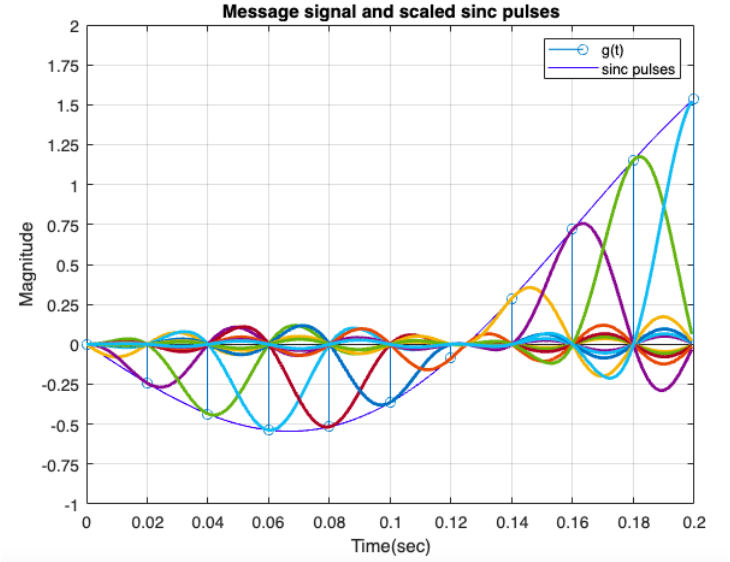

Figure 8: Figure for part B

2.3 Answer 3

Inferences are :

1. We can see that the reconstructed signal is following the original message signal for earlier part of the time that is from 0 to 0.18 secs. However from 0.18 – 0.2 sec, around that time period actually, we can see some small deflections from the original message signal. These deflections are because in the code, we are trying to estimate upto 0.2 secs. On increasing estimation time period to 1 sec, it will be able to fit the original curve completely. Another way to properly fit the estimated curve is using more samples of time shifted sinc functions. Another reason for the improper fitting at the end of time interval is that the message signal in this region is changing rapidly.

2. Another thing that we can observe is that the signal is not quantized here. If we try to quantize the signal, we will end getting quantization error. As a result, it will not fit the curve so properly. For that, we will need to increase the number of sinc functions.

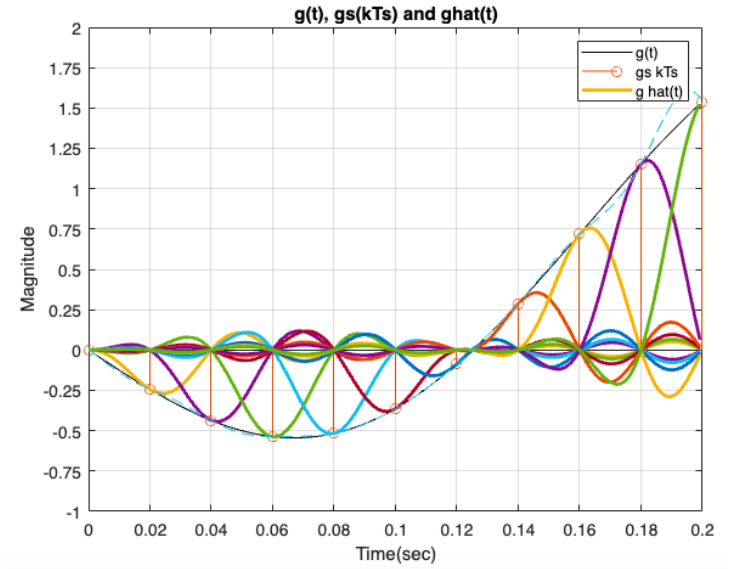

Figure 9: Figure for part C

2.4 Answer 4

In part (c), we haven't performed any quantization. In this part, we are first quantizing the sampled signal and then passing it through the filter. Quantization of sampled signal is necessary as we can't have a bit for a continuous value that the sampled signal takes. It is like infinite quantization levels, which is impractical in real life. We have to send the message in as much less bits as possible, keeping in mind the quantization error. Here we have quantized the signal into 16 levels or 4 bits.

Inferences :

1. We can see that the reconstructed signal is following the message signal for most of the time period except for the later time that is around 0.18 to 0.2 sec. This is because of the same reasons stated in the previous answer.

2. The levels of quantization is finite here. So it is a more closer to the real world communication process case as compared to the previous one.

3. In the whole experiment, we have assumed that we are having a noiseless channel. Suppose we have a channel with additive white gaussian noise. The case mentioned in part 1c will be completely impractical to implement, as the noise can take any value (due to randomness). Creating a decision boundary for predicting the correct bit sequence will also be tough. Having infinite quantization layers won't make any sense. Reducing the quantization levels to 16 will ease the work of making decision boundaries and improve the prediction of the bit sequence transmitted. Thus we can say that it will be more robust to noise as compared that of 1c.

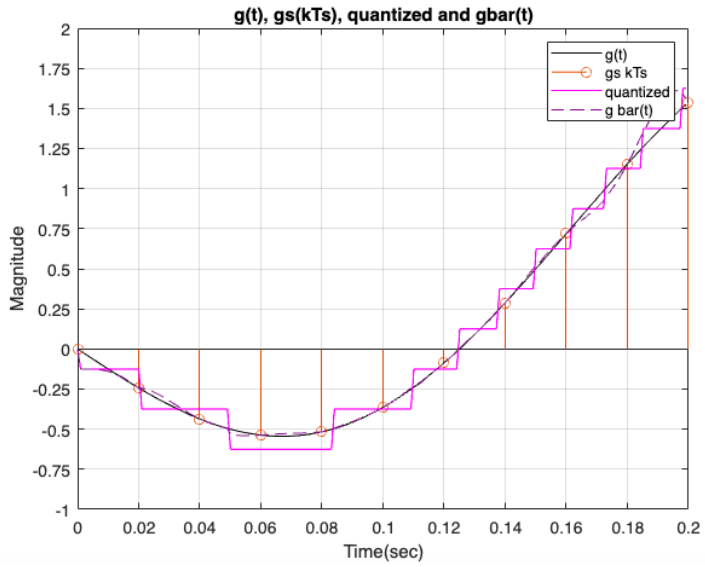

Figure 10: Figure for part D

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
`xticks`	marks the ticks in x axis
`yticks`	marks the ticks in y axis

B Matlab Code

Matlab codes for each part.

B.1 Q1(a)

```
% question1(a)
```

```
clc;
close all;
t = 0:1/999:0.2;

%frequency of the impulse in Hz
f=50;
fs=f;
y=zeros(size(t));
y(1:fs/f:end)=1;

%time shifted Sinc function

y1 = sinc(t*f);
figure(2);

subplot(2,1,1);
plot(t,y1);
grid on;
title('sinc(t/TS) signal');
xlabel('Time(sec)')
ylabel('Magnitude')
yticks([-2 -1.75 -1.5 -1.25 -1 -0.75 -0.5 -0.25
0 0.25 0.5 0.75 1 1.25 1.5 1.75 2])
xticks([0 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2]);
grid on;

subplot(2,1,2);
T = 1/50;

% Shifted sincs
% iterate from 0 to 10
for k = 0:10
    y2 = sinc((t/T) - k);
    plot(t,y2,'linewidth',2);
    hold on;
end

title('shifted sincs');
xlabel('Time(sec)')
ylabel('Magnitude')
yticks([-2 -1.75 -1.5 -1.25 -1 -0.75 -0.5 -0.25
```

```
0 0.25 0.5 0.75 1 1.25 1.5 1.75 2])
xticks([0 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2]);
grid on;
```

B.2 Q1(b)

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

%Message signal
Am =1;
fm1 = 1;
fm2 = 3;
t = 0:1/999:0.2;
ym = Am*(sin(2*pi*fm1*t)-sin(2*pi*fm2*t));

%impulse train
fs=50;
T = 1/fs;
y=zeros(size(t));
y(1:1:end)=1;

%sampled signal
figure(1);
g_kTs= ym .* y;
t1 = 0:0.02:0.2;
temp=zeros(size(t1));
i=1;
for k=t1
    temp(i) = Am*(sin(2*pi*fm1*k)-sin(2*pi*fm2*k));
    i=i+1;
end
stem(t1,temp);
hold on;

%Sinc function
plot(t,ym,'b');
hold on;
```

```
for k = 0:10
    y2 = sinc(((t - k*T) / T));
    y3 = y2 .* g_kTs;
    plot(t,y3,'linewidth', 2);
    hold on;
end

title('Message signal and scaled sinc pulses');
xlabel('Time(sec)')
ylabel('Magnitude')
yticks([-2 -1.75 -1.5 -1.25 -1 -0.75 -0.5 -0.25
0 0.25 0.5 0.75 1 1.25 1.5 1.75 2])
xticks([0 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2]);
legend('g(t)','sinc pulses')
grid on;
```

B.3 Q1(c)

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

%Message signal
Am =1;
fm1 = 1;
fm2 = 3;
t = 0:1/999:0.2;
ym = Am*(sin(2*pi*fm1*t)-sin(2*pi*fm2*t));


%impulse train
fs=50;
T = 1/fs;
y=zeros(size(T));
y(1:1:end)=1;

%sampled signal
g_kTs= y.*ym;
```

```
%summation of products
y1 = sinc(t*fs);
y2 = g_kTs.* y1;
plot(t,ym,'k');
hold on;

%plot(t,g_kTs);
t1 = 0:0.02:0.2;
temp=zeros(size(t1));
i=1;
for k=t1
    disp(i);
    temp(i) = Am*(sin(2*pi*fm1*k)-sin(2*pi*fm2*k));
    i=i+1;
end
stem(t1,temp);
hold on;

for k = 1:10
    y2 = y2 + g_kTs .* sinc((t/T) - k);
    y3 = g_kTs .* sinc((t/T) - k);
    plot(t,y3,'linewidth', 2);
    hold on;
end

plot(t,y2,"--");
hold off
title('g(t), gs(kTs) and ghat(t)');
xlabel('Time(sec)')
ylabel('Magnitude')
yticks([-2 -1.75 -1.5 -1.25 -1 -0.75 -0.5 -0.25
0 0.25 0.5 0.75 1 1.25 1.5 1.75 2]);
xticks([0 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2]);
legend('g(t)','gs kTs','g hat(t)');
grid on;
```

B.4 Q1(d)

```
% question1(d)
```

```
clc;
close all;

%Message signal
Am =1;
fm1 = 1;
fm2 = 3;
t = 0:1/999:0.2;
ym = Am*(sin(2*pi*fm1*t)-sin(2*pi*fm2*t));


%impulse train
fs=50;
T = 1/fs;
y=zeros(size(t));
y(1:1:end)=1;

%sampled signal
g_kTs= ym.*y;

%Quantisation
nbits = 4;
qLevels = 2^nbits;

signalMin = -2;
signalMax = 2;
scalingFactor = (signalMax-signalMin)/qLevels;
gs_kTs = g_kTs/scalingFactor;

gs_kTs1 = floor(gs_kTs);
gs_kTs2 = ceil(gs_kTs);
gs_kTs = (gs_kTs1 + gs_kTs2)/2;
gs_kTs = gs_kTs * scalingFactor;

%Summing of Sinc functions
y1 = sinc(t*fs);
y2 = gs_kTs .* y1;
plot(t,ym,'k');
hold on;
%plot(t,g_kTs);
t1 = 0:0.02:0.2;
```

```
temp=zeros(size(t1));
i=1;
for k=t1
    disp(i);
    temp(i) = Am*(sin(2*pi*fm1*k)-sin(2*pi*fm2*k));
    i=i+1;
end
stem(t1,temp);
%hold on;
hold on;
plot(t,gs_kTs,'m');
hold on;

for k = 1:10
    y2 = y2 + g_kTs .* sinc((t/T) - k);
end

plot(t,y2,'--');
title('g(t), gs(kTs), quantized and gbar(t)');
xlabel('Time(sec)');
ylabel('Magnitude');
yticks([-2 -1.75 -1.5 -1.25 -1 -0.75 -0.5 -0.25
0 0.25 0.5 0.75 1 1.25 1.5 1.75 2]);
xticks([0 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2]);
legend('g(t)','gs kTs','quantized','g bar(t)');
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 signal reconstruction for signal processing `https://en.wikipedia.org/wiki/Signal_reconstruction`

[4] Reconstruction of signals on Youtube `https://youtu.be/cUdnGkymAWU`

[5] Signal Reconstruction lectures `https://ptolemy.berkeley.edu/projects/chess/eecs124/reading/LeeAndVaraiya11.pdf`