Soraya charkas - parisa toumari - radin khayyam
99101387 - 99101857 - 99101579

PROJECT REPORT

Question 1

Write a MATLAB code to simulate the communication system. Create separate functions for each block and then, connect them in a main mfile. Name the functions adc, linecoder, channel, linedecoder, and dac. Each function might have an arbitrary number of input arguments. For example, adc function might accept and f_s as its input arguments.

(a) ADC Function

This function receives analog signal , time(signal's length in seconds) , nu(number of quantization bits) , and f_S (sampling frequency)as input. at first, we have to find our analog signal's frequency(we assume that our analog signal is a signal with a high sampling frequency) by dividing our signal's length by time.after that, we down sampled our signal with a specific rate(which is the analog frequency to f_s ratio).after that, we adjust L as our down-sampled signal's length.then we quantized our signal with our written function.the output is our digital signal.

```
function [t,digital_signal]=adc(analog_signal,time,nu,fs)
    f_analog=length(analog_signal)/time;
    sampled_signal=downsample(analog_signal,ceil(f_analog/fs));
    L=length(sampled_signal);
    quantized_signal=zeros(1,L);
    N=2^nu;
    min_signal=min(sampled_signal);
    max_signal=max(sampled_signal);
    delta=(max_signal-min_signal)/N;
    for i=1:1:L
        for k=0:1:N-1
            if((sampled_signal(i)>=min_signal+k*delta)&&(sampled_signal(i)<=min_signal+(
                k+1)*delta))
               quantized_signal(i)=min_signal+k*delta+delta/2;
            end
        end
    end
    digital_signal=quantized_signal;
    t=linspace(0,time,length(digital_signal));
end
```

now we plot the output for a sample sinusoidal input,and we will repeat the experiment for different values for ν. figures are plotted at the end of the report.

(b) Line Coder Function

we define the time it takes for our signal to be communicated L.D(D equals to length of each pulse and L is the number of bits being communicated).using this definition, we define tD with a length of L.D, we generated ak as written (note that it should be polar).we define the line coded signal as a 0 matrix with a length of 2 times L.the pulse we generated depends on beta and r.after defining all of the parameters,we use a loop with a length of ak , in this loop we define the s vector which consists of pulse vector in the middle and zero vectors

Question 1 continued on next page...

at the beginning and at the end.by shifting the pulse vector,we are also shifting the s vector and in the end, we have our final line coded signal by adding the previous value to the new one which is $s(k+1)a_k$

```
function [t,line_coded_signal] = line_coder_nyquist(encoded_signal, r,A,beta,line_code_f
    )

    L=length(encoded_signal);
    D=1/r;
    tD=-L*D/2:1/line_code_f:L*D/2-1/line_code_f;
    ak=(encoded_signal-0.5)*A;
    line_coded_signal=zeros(1,round(2*L*D*line_code_f));
    p =cos(2*pi*beta*tD) ./ (1-(4*beta*tD).^2) .* sinc(r*tD);
    for k=0:1:L-1
        s=[zeros(1,round(k*D*line_code_f)) p zeros(1,round((L-k)*D*line_code_f))];
        line_coded_signal=line_coded_signal+ak(k+1)*s;
    end
    line_coded_signal=line_coded_signal(1/2*L*D*line_code_f:3/2*L*D*line_code_f-1);
    t=0:1/line_code_f:L*D-1/line_code_f;
end
```

(c) Band-limited Channel Function

In this function,first we create a white Gaussian noise using Matlab's pre-defined functions.then we add the noise to our signal and we pass the new signal through a low-pass filter .note that B must be smaller than $\frac{f_s}{2}$ and bigger than signal's bandwidth.

```
function [recieved_signal]=channel(line_coded_signal,N0,B,fs)
    var=N0*B;
    noise=wgn(size(line_coded_signal,1),size(line_coded_signal,2),var,'linear');
    noise_added_signal=noise+line_coded_signal;
    filtered_signal=lowpass(noise_added_signal,B,fs);
    recieved_signal=filtered_signal;
end
```

the output signal will be as below.

(d) Line Decoder Function

After receiving the signal from the previous function , we use it as an input for our line decoder function and decoded it the following instruction:we sample our signal with a sampling rate of D,then we set the value of each sample 0 or 1. if it is more than zero, then we assume that we were trying to communicate 1 and if its less than zero, we assume that we were trying to communicate 0. with this method, we can regenerate the 0-1 string we had before.

```
function [line_decoded_signal]=line_decoder(recieved_signal,line_code_f,r)

    sampled = recieved_signal(1:line_code_f/r:end);
    L=length(sampled);
    line_decoded_signal = zeros(1,L);
    line_decoded_signal(sampled>0)=1; %creating binary string

end
```

Question 1 continued on next page...

(e) DAC Function

In this function,we have to regenerate an analog signal from the string we generated in the line decoder function.to achieve this goal, we separate our string into ν bit sub strings and we dedicate the decimal value to each sub string.at this point, we have a digital sampled signal with a frequency of 8800.as we know we are trying to have an analog signal with a frequency of 44000so we have to do the opposite of what we have done at the first place,so we have to up sample our signal to generate an analog signal with a frequency of 44000.the only difference between the analog signal that we had at the first place and the signal we regenerated now is that due to the quantization we have lost some data.we can reduce the effect of quantization by increasing ν

```
function[analog_converted_signal]=dac(line_decoded_signal,nu,min_signal,max_signal,fs,
    f_analog)

    N=2^nu;
    delta=(max_signal-min_signal)/N;
    dec_data=reshape(line_decoded_signal,nu,length(line_decoded_signal)/nu)';
    decoded_sampled=bi2de(dec_data,'left-msb');%converting the binary string to
        decimal levels
    reconstructed_signal=(delta*decoded_sampled);%reconstructing the signalS
    reconstructed_signal=reconstructed_signal+(min_signal+ delta/2);
    analog_converted_signal=interp(reconstructed_signal,f_analog/fs);

end
```

Question 2

Feed your simulation setup with a recorded audio file and play the received voice and hear it for different noise level N_0 and channel bandwidth B. Assume that the ADC/DAC parameters are suitably tuned such that aliasing and SQNR have acceptable values and the channel bandwidth is enough such that no ISI appears. How do you feel when you hear the received voice? Note that you can record your voice from your laptop microphone and feed it to the communication system. You can also hear the received voice via your laptop speaker. MATLAB has useful internal commands for working with microphones and speakers!

after testing for different noise levels and bandwidths, we understood that the influence of the bandwidth is much more less than noise's. first we fix B at 500 and change the N_0's order from 10^{-5} and 10^{-4} and the difference is soooo noticeable!the actual voice is hardly recognizable. now we fix N_0 at 10^{-5} and change the bandwidth from 50 and 5. the results aren't much different from eachother and the audio is still clear.
the plots are in the end of the report.

Question 3

Use the simulation setup to calculate the BER of the system and plot it in terms of noise σ^2 and amplitude A. Assume that the ADC/DAC parameters are suitably tuned such that aliasing and SQNR have acceptable values and the channel bandwidth is enough such that no ISI appears. Discus the obtained plots.

Question 3 continued on next page...

According to the Nyquist theory, we know that the sampling rate must be 2 times bigger than the bandwidth. and the bandwidth of the human voice is around 20khz but we can not sample at 40khz.since the signal's bandwidth witch contains almost every info is in around 4-5khz,if we sample at 8khz , we wont be loosing much info.
The more the sampling rate increases, the more immune the result would be from noises. on the other hand, the more ν increases, the more precise the output signal would be; but we cant increse it as much as we want; cuz the bandlimmited filter that we apply in the channel is depended on r (r=νf_s) and it shouldnt get as high as much as we want cuz we have a fixed bandwidth so we should either increase f_s or ν.

Question 4

Repeat the previous two parts when the channel bandwidth limitation imposes ISI. Investigate the impact of rolloff factor β on the imposed ISI.

we can see that when we increase baud rate or roll-off factor,the ISI decreases.for example, we have set $B = 0.01$ and in this case we have imposed ISI.by decreasing B to 0.005 the quality of output sound is lowered and ISI is increased. but when we increase it the noise becomes less and ISI becomes less.

Question 5

Investigate the impact of the sampling rate fs and number of quantization bits ν on the performance of the communication system.

the more we increase f_s or ν , the more precise and better the output will be. and we should know that the quantization operation that has effected the input signal, is not reversible; so we can use a high level of quantization to reduce the effect in the output.on the other hand,the frequency range of our voice is about 8khz and most of the information of the input signal is in the bandwidth of 8khz ; so if we reduce f_s to around 8khz, we wont lose much info and also f_s can't be more that half of the Fs or alising would happen.
the plots are in the end of the report.

BONUS QUESTIONS

Question 6

Return your answers by filling the LaTeXtemplate of the assignment.

recorded sounds are attached in the project's zip file.

Question 6 continued on next page...

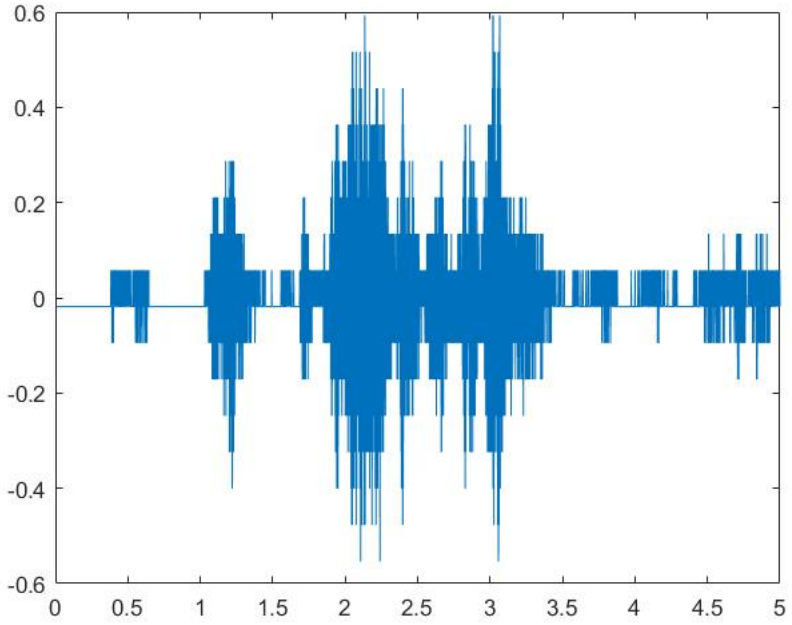

Figure 1: ADC function output for $\nu = 4$

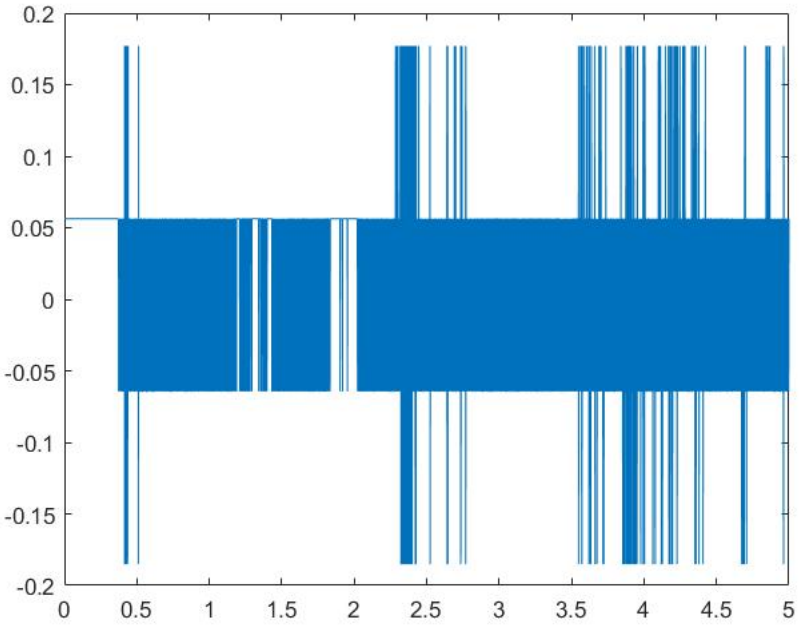

Figure 2: ADC function output for $\nu = 2$

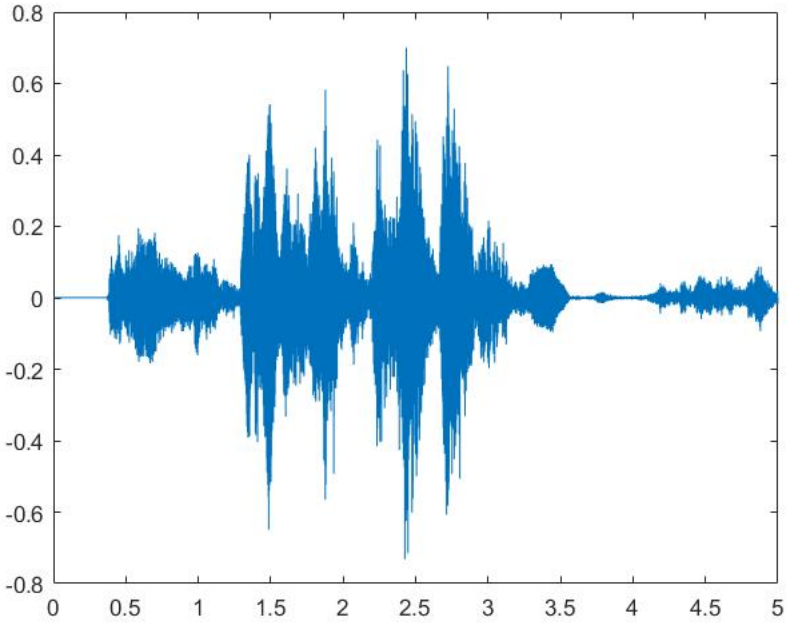

Figure 3: ADC function output for $\nu = 10$

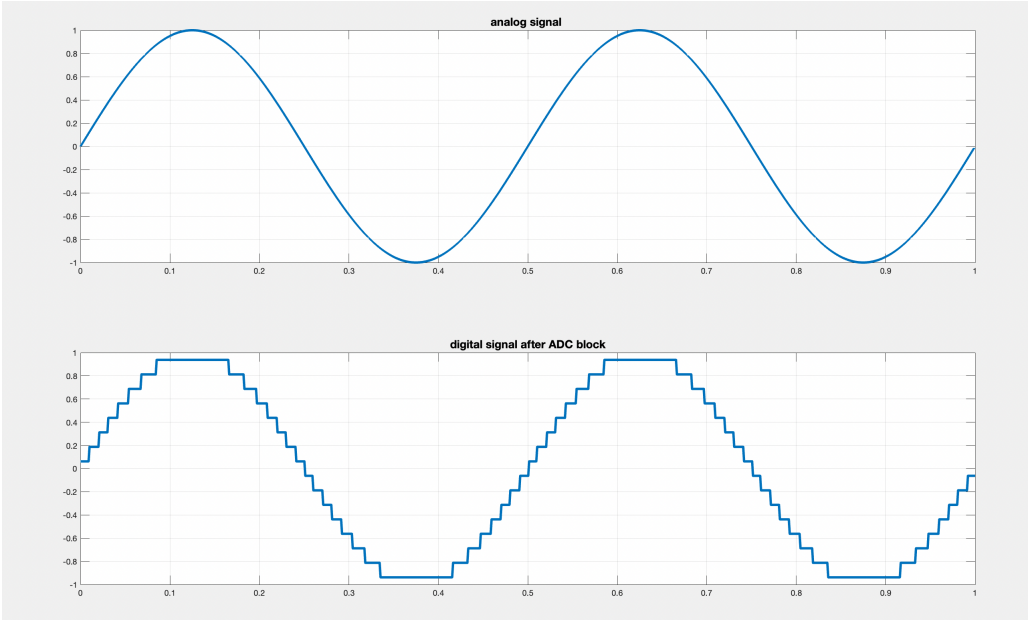

Figure 4: analog signal-digital signal after ADC function

Question 6 continued on next page...

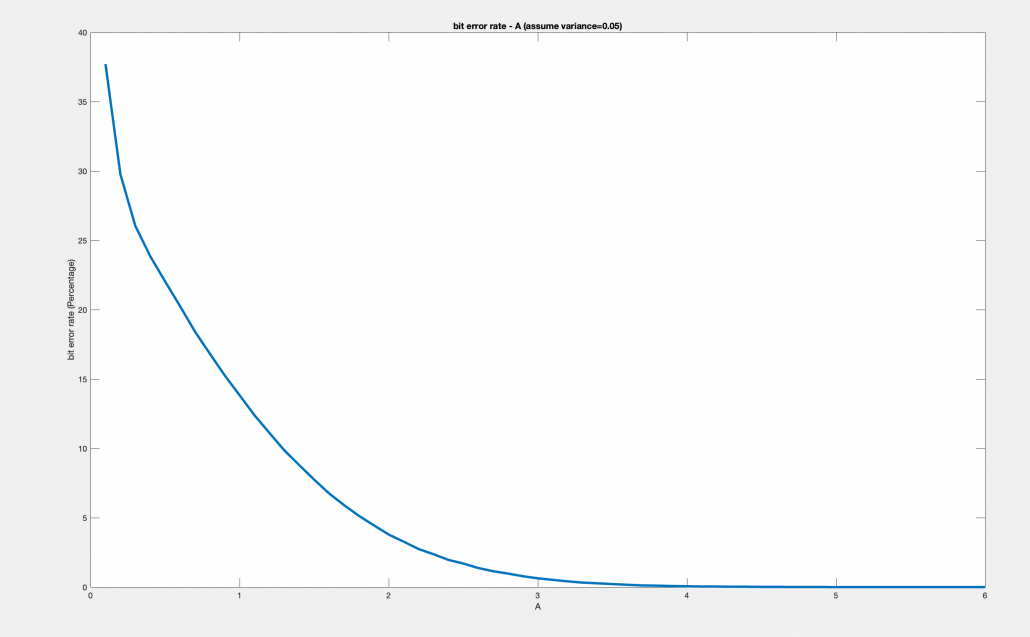

Figure 5: Bit Error Rate-Amplitude

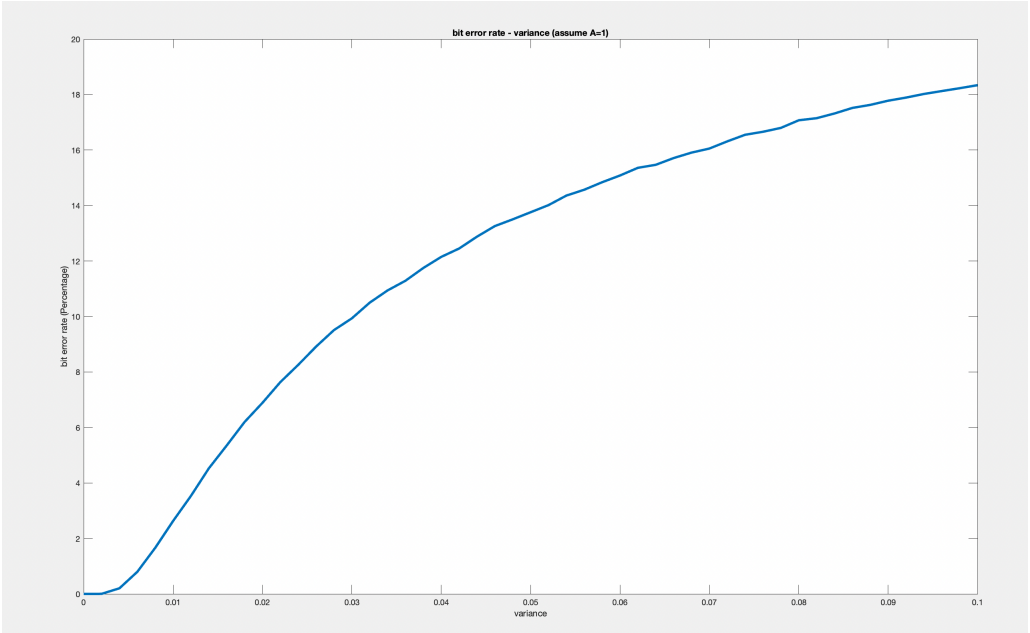

Figure 6: Bit Error Rate-Variance

Question 6 continued on next page...

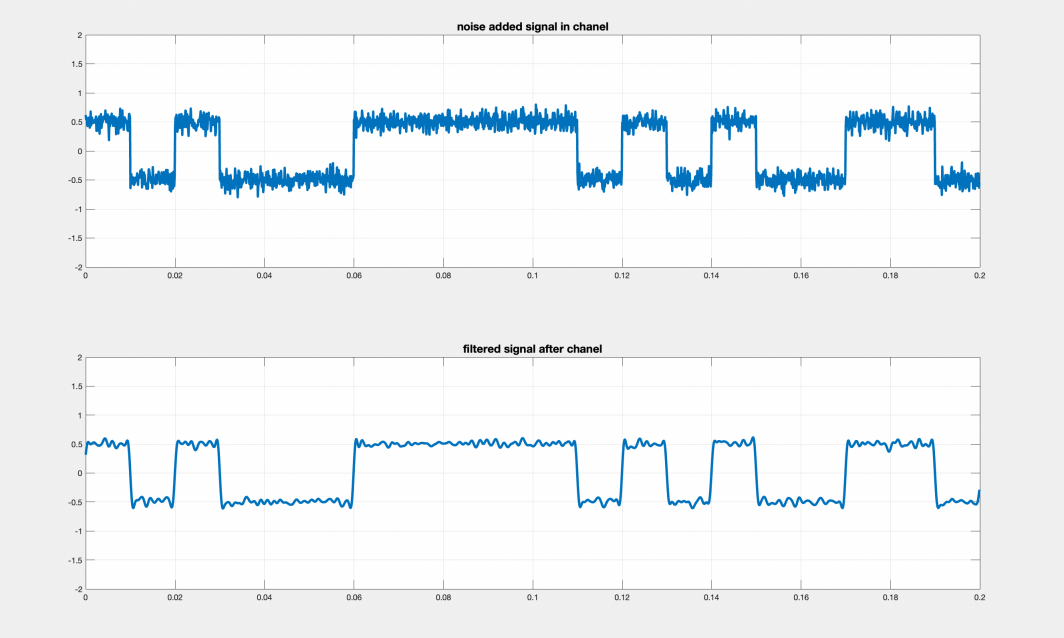

Figure 7: Channel

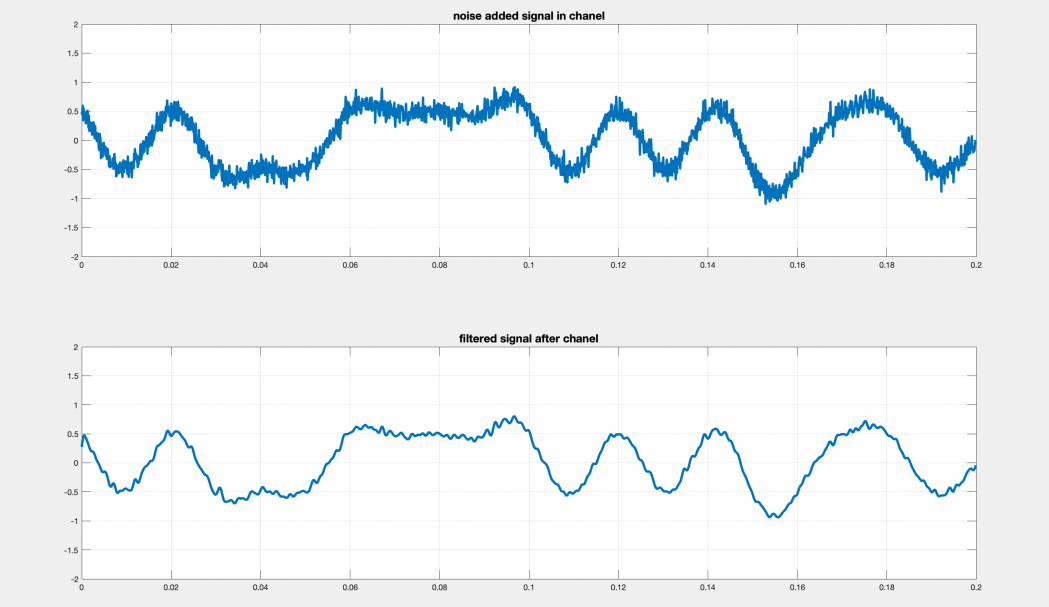

Figure 8: line coding-channel

Question 6 continued on next page...

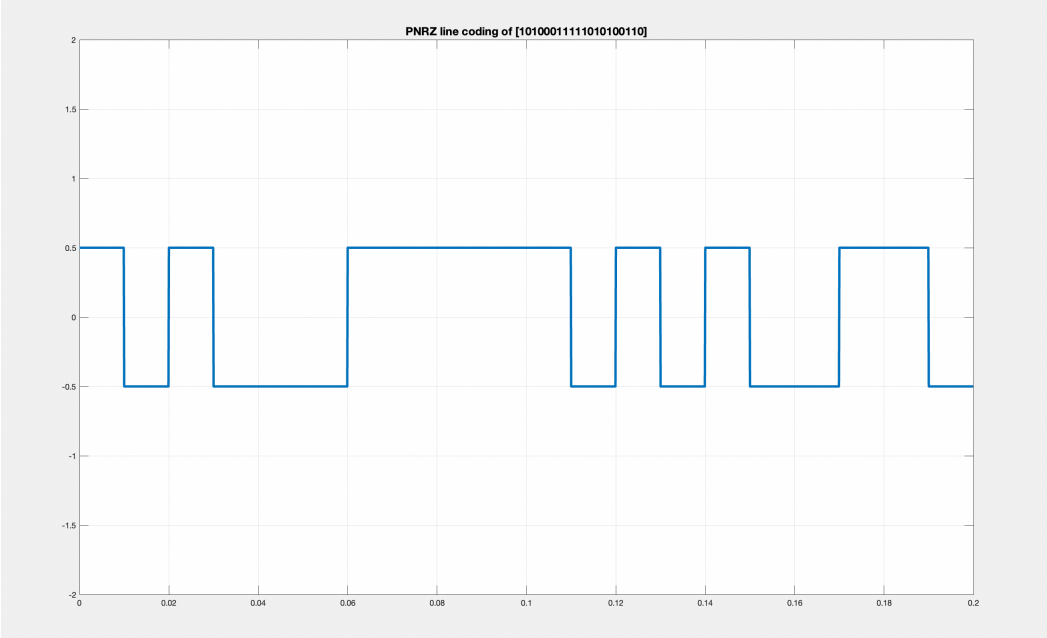

Figure 9: PNRZ line coding

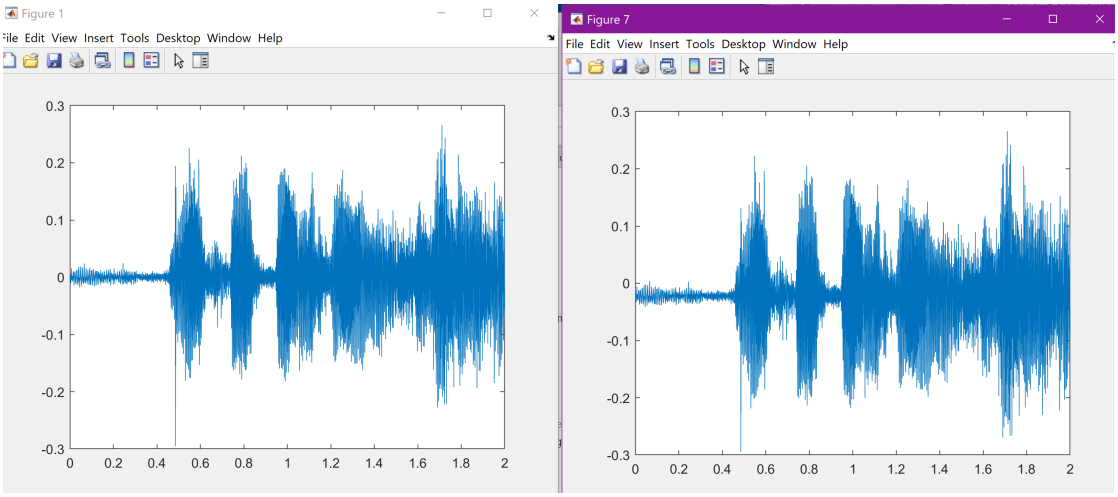

Figure 10: Bandwidth 5 and $N_0 = 10^{-5}$

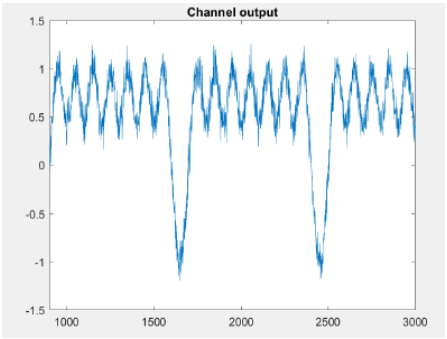

Figure 11: $B = 0.01$

Question 6 continued on next page...

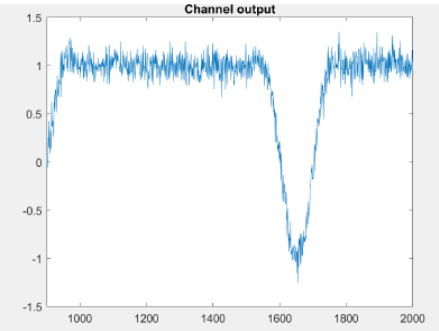

Figure 12: $B = 0.005$

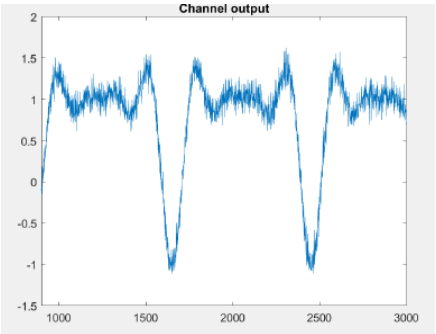

Figure 13: Imposed ISI- increasing r

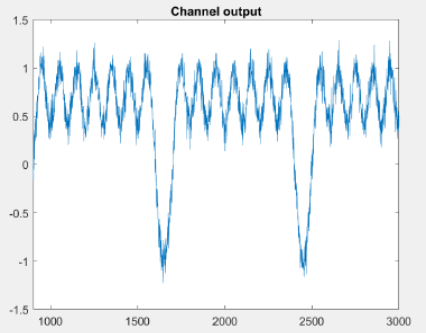

Figure 14: Imposed ISI-decreasing r

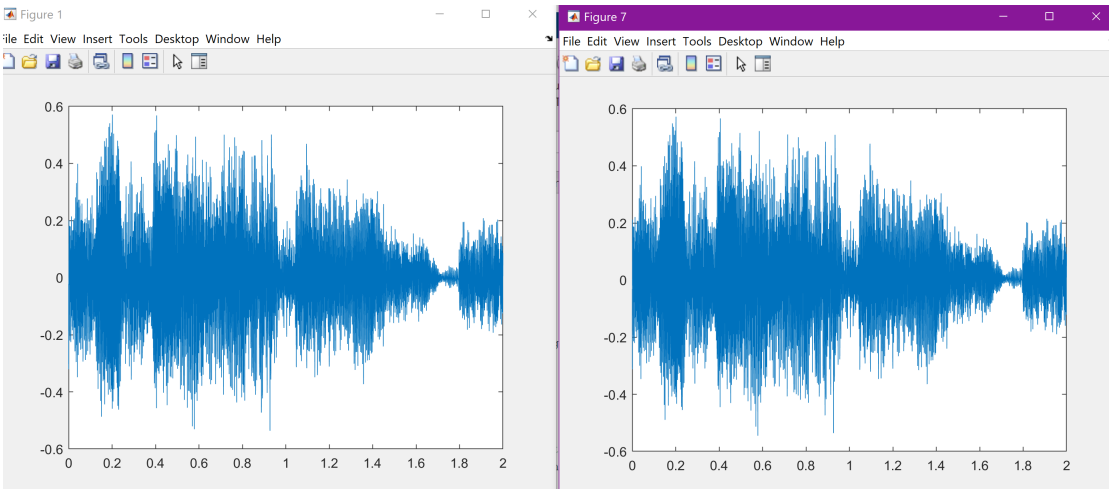

Figure 15: Bandwidth 50 and $N_0 = 10^{-5}$

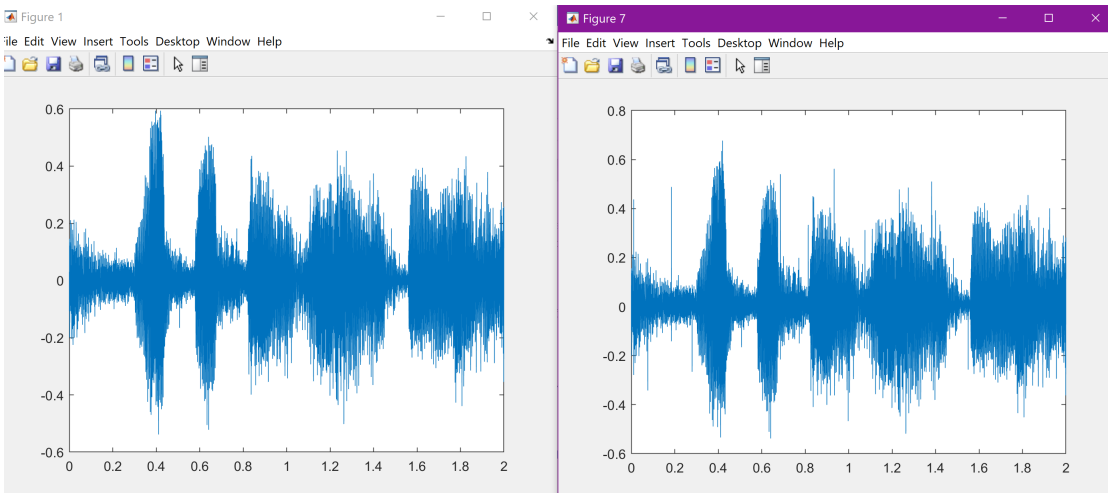

Figure 16: Bandwidth 500 and $N_0 = 10^{-5}$

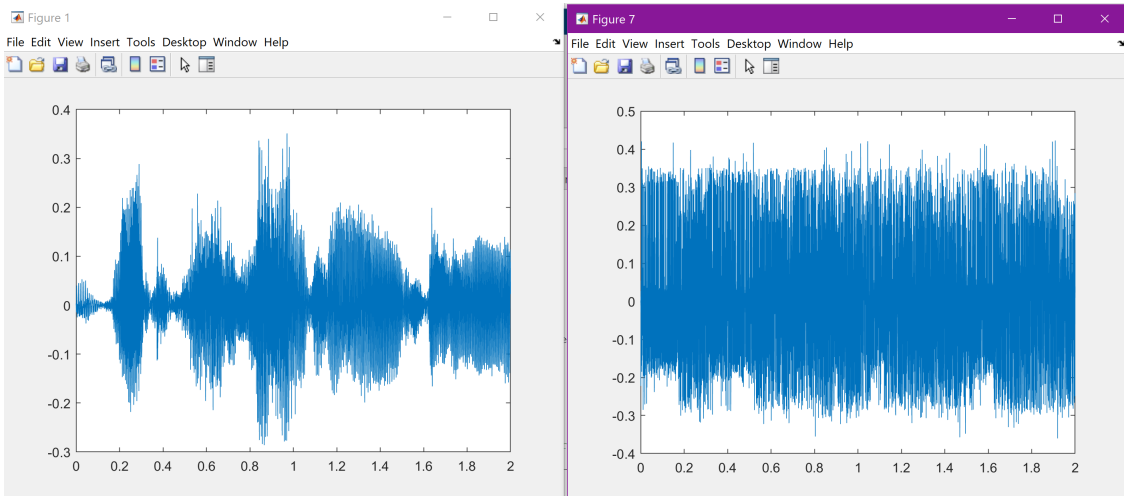

Figure 17: Bandwidth 500 and $N_0 = 10^{-4}$

Question 6 continued on next page...

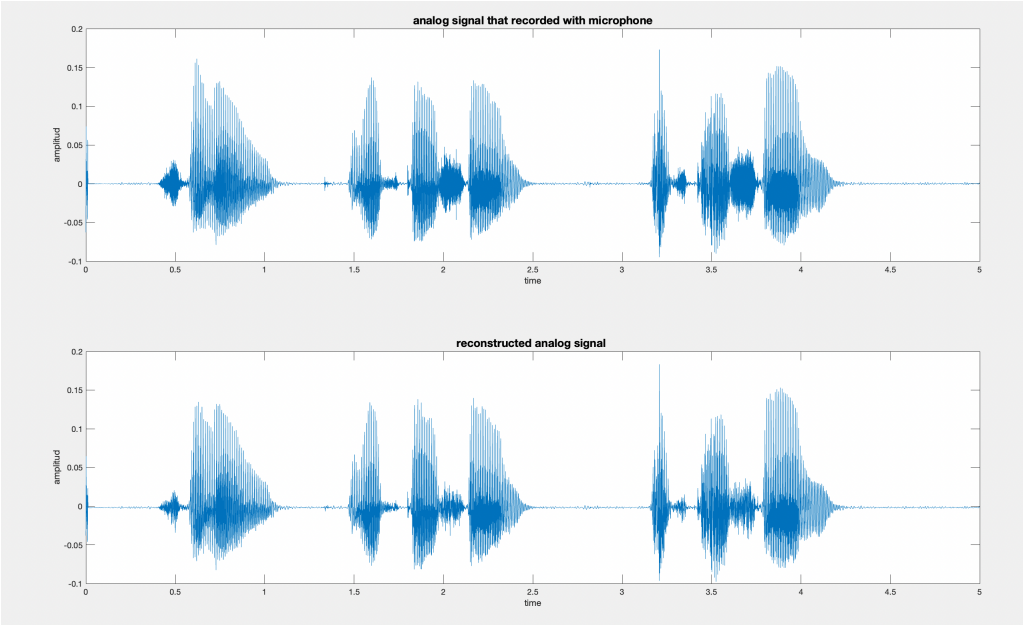

Figure 18: Input and output with $f_s = 0.05F_s$

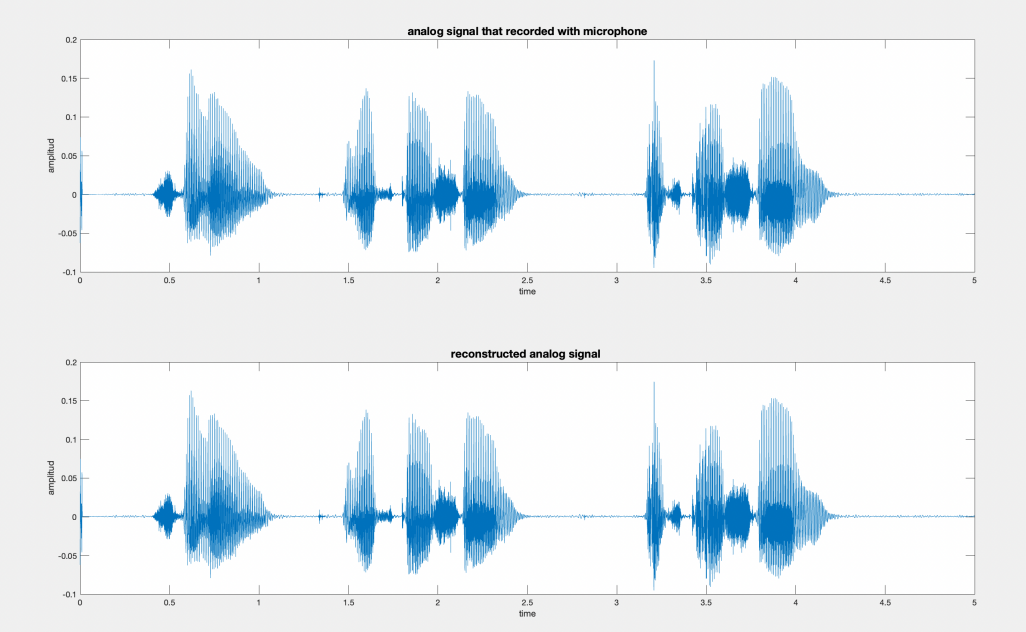

Figure 19: Input and output with $f_s = 0.5F_s$

Question 6 continued on next page...