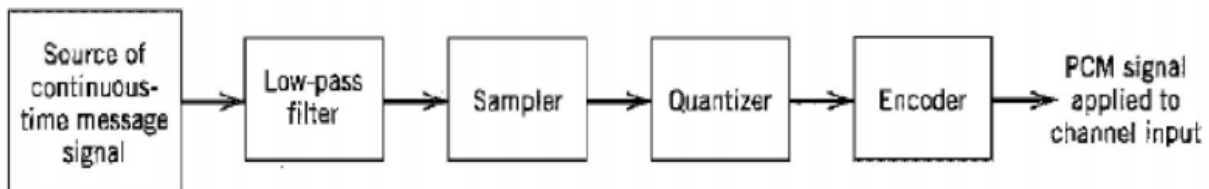


Lab 2: Pulse code modulation (PCM)

Objective:

- (1) Investigate the PCM system components.
- (2) Investigate the effect of changing the number of levels.
- (3) Investigate the oversampling, critical sampling and under sampling cases.
- (4) Calculate the quantization error of the transmitted signal.

Theoretical Background:



pulse code modulation is the traditional baseband analog to digital converter which consists of sampling, quantization and coding processes.

The sampling process is transformation of analog signal into discrete signal, the signal must be sampled with at least Nyquist rate $f_s = 2f_m$. At this case the sampling would be critical sampling which requires a sharp ideal reconstruction filter. if the sampling is done with $f_s \gg 2f_m$, the sampling would be oversampling and a smoother filter can be used. if $f_s < 2f_m$, the signal cannot be reconstructed due to aliasing.

Quantization process is approximation process of the output samples from the sampling process to discrete levels which equals 2^n where n is the number of bits that defines the sample. Increasing number of bits will result in enhancing the SQNR and hence better QoS. The encoding process is transforming the quantized bits into proper line code.

Calculation mean square quantization error

The quantization error is calculating by averaging the quantization error over all the transmitted samples

$$MSE_q = \frac{1}{N} \sum_{n=0}^{N-1} [x_q(n) - x(n)]^2$$

Generate any signal, quantize it. Draw the quantization error on the Y axis versus n bits on the horizontal axis

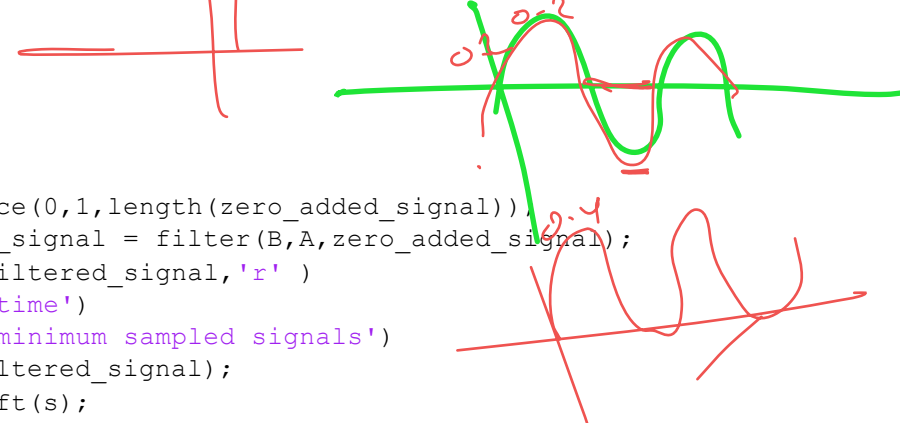
Procedure:

Quantization error:

- (1) Generate a sinusoidal wave of the following parameters:
 - a. Amplitude=1V.
 - b. Frequency= 2Hz.
 - c. Sampling frequency=4000Hz.
- (2) Quantize the sampled signal by m bits where $m = 2n + 1$ and n is the number of bits that will represents the integer value and the fraction part and the last bit is the sign bit
(Hint : use `fi` command)
`a = double(fi(v,s,m,n))`
- (3) Convert the quantized samples to binary (you may use `de2bi` or any other suitable method)
- (4) Calculate the mean square quantization error as equation for $n = 3, 4, 5, 10$

Sampling distortion:

```
% this part below must be copied to your m file and complete the %
required
clear
clc
% reconstruction from oversampling
t=0:0.001:1;% time signal
y=2*cos(2*pi*5*t);
[B,A] = BUTTER(3,1000/100000,'low'); % butter fly filter
zero_added_signal=zeros(1,length(y)*10);
for i=1:length(y)
    zero_added_signal(i*10)=y(i);
end
zero_added_signal(1:9)=[];
% Adding zeros enhances the signal display and don't change the
% spectrum, it changes sampling freq. only
t=linspace(0,1,length(zero_added_signal));
filtered_signal = filter(B,A,zero_added_signal);
plot(t,filtered_signal,'r')
XLABEL('time')
YLABEL('oversampled signals')
% construction from minimum sampling
figure
t=0:?:1; % replace ?? with the suitable number
y=2*cos(2*pi*5*t);
[B,A] = BUTTER(10,0.1,'low');
zero_added_signal=zeros(1,length(y)*10);
for i=1:length(y)
    zero_added_signal(i*10)=y(i);
end
zero_added_signal(1:9)=[];
```



```

t=linspace(0,1,length(zero_added_signal));
filtered_signal = filter(B,A,zero_added_signal);
plot(t,filtered_signal,'r' )
XLABEL('time')
YLABEL('minimum sampled signals')
s=fft(filtered_signal);
s=fftshift(s);
fs=100; % why 100?? Write your comments in the m file
freq=linspace(-fs/2,fs/2,length(s));
figure
plot(freq,abs(s))
XLABEL('freq')
YLABEL('magnitude of minimum sampled signals')
% construction from undersampling sampling
figure
t=0:0.2:1;
y=2*cos(2*pi*5*t);
[B,A] = BUTTER(10,0.2,'low' );
% complete this part as shown in the construction from minimum sampling
%and do the necessary changes , you have to do low pass filtering and %
displays the spectrum

```

For this code Complete the attached file to see the difference between the oversampling, critical sampling and under sampling, you need to understand all the used commands.

Report requirement:

- (1) Well commented M-file.
- (2) Softcopy report containing required figures , comment.
- (3) Repeat the program for changing fraction and integer parts , find the best resolution
- (4) Repeat the program using a quantizer of your choice using `quantize` command
- (5) Repeat the program with non-uniform quantizer (you can use `compand`)