Report of Signals and Systems Coursework (Sampling)



**Dosen Pengajar:**

Nama Dosen : Moh. Kamalul Wafi, S.T., MSc.DIC

**Disusun Oleh :**

Denny Alfani (02311940005012)

Fiqiyah ulul azmi (02311940005010)

Hanif adi rahman (02311940005011)

**DEPARTEMEN TEKNIK FISIKA**

**FAKULTAS TEKNOLOGI INDUSTRI**

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**

**SURABAYA**

**2019**

# Problem 1

Source code

%problem 1

clc; %clean screen

clear all; %clear removes all variables from the current workspace

close all; %deletes all figures whose handles are not hidden

[y,fs] = audioread('lagu.mp3');

audiowrite('lagu.flac',y,64000);

[y,fs] = audioread('lagu.mp3');

audiowrite('lagu.flac',y,128000);

[y,fs] = audioread('lagu.mp3');

audiowrite('lagu.flac',y,256000);

[y,fs] = audioread('lagu.mp3');

audiowrite('lagu.flac',y,320000);

note: berlaku pada format ogg dan wav.

Pada problem 1, kami memilih lagu dari band Evanescense yang berjudul Bring Me to Life yang kami rename dengan judul “lagu” agar memudahkan dalam pengerjaan. Lagu tersebut berformat mp3, yang kemudian di convert menjadi format: ogg, wav, dan flac, kemudian frekuensi sampling (fs) diubah menjadi: 64 kHz, 128 kHz, 256 kHz, dan 320 kHz.

Setelah diperdengarkan, kualitas audio pada format ogg, wav, flac, dan mp3 tidak terdengar ada perbedaan. ketika frekuensi sampling (fs) diganti menjadi 64 kHz, lagu yang mulanya berdurasi 3:59 menjadi 2:44, berlaku semua format (ogg, wav, dan flac). Pada kenaikan frekuensi sampling menjadi 128 kHz, durasi lagu menjadi lebih pendek yaitu 1:22. Pada kenaikan frekuensi sampling menjadi 256 kHz, durasi lagu menjadi setengah lebih pendek dari frekuensi sampling 128 kHz yaitu 0:41, namun, pada format file ogg tidak didapatkan hasil. Pada kenaikan frekuensi sampling menjadi 320 kHz, durasi lagu menjadi sangat pendek yaitu 0:32, dan format .ogg tidak didapatkan hasil. Karena pada case ini, menggunakan Analisa f=1/fs

Dari percobaan yang kami lakukan, semakin besar frekuensi sampling, maka semakin pendek durasi lagu yang didapatkan.

# Problem 2

Pada tugas ini diambil foto dari satu objek dengan kamera berbeda yang memiliki resolusi: 4MP, 6MP, 8MP, 12MP dan 16MP. Adapun hasil dari foto yang telah diambil adalah sebagai berikut:

A computer sitting on a table

Description automatically generated

Figure 1. using 4 MP

A close up of a computer

Description automatically generated

Figure 2. using 6 MP

A picture containing indoor, table

Description automatically generated

Figure 3. using 8MP

A picture containing indoor, table, desk, sitting

Description automatically generated

Figure 4. Using 12 MP

A picture containing table, indoor

Description automatically generated

Figure 5. using 16MP

Perbedaan paling utama adalah ukuran dari file gambar. Semakin besar resolusi kamera maka semakin besar ukuran file dari gambar yang dihasilkan. Selain itu, ketika dilakukan zoom pada gambar, kamera dengan resolusi yang semakin tinggi akan menampilkan citra yang semakin detail. Namun kualitas gambar ini tidak semata-mata ditentukan oleh tingkat resolusi dari kamera, tetapi juga dari kualitas lensa dan sensor kamera, serta diafragma lensa tersebut.

# Problem 3

Source code

%problem 3

clc; %clean screen

clear all; %clear removes all variables from the current workspace

close all; %deletes all figures whose handles are not hidden

fs=1000; %frequency sampling (banyaknya data yg diambil tiap detik)

t=0:1/fs:0.25; %time interval

f=200; %frequency

x=sin(2\*pi\*f\*t); %signal data set

sound(x,fs); %sends audio signal x to the speaker at sample rate Fs.

plot(t,x)

This command produces a sound at 100 Hz frequencies.

Pada problem 3, percobaan dilakukan dengan mengganti frekuensi, dari 100 sampai dengan 900, dengan setiap kenaikan sebesar 100. Pada f = 200, 300, 400, 600, 700, 800, dan 900 dihasilkan suara dengan frekuensi sesuai dengan nilai f. Namun, berbeda pada f=500 karena tidak ada suara yang dihasilkan.

Berbungi ‘tut’

Untuk nilai x = frekuensi alias, sin(2\*pi\*f\*t) = nilai signal x, untuk f adalah variable yang diganti.

Setelah didengarkan hasil dari suara yang keluar dan mengamati hasil grafik pada masing-masing frekuensi, didapatkan data sebagai berikut:

* Pada f=600 dihasilkan suara yang sama dengan f=400

A screenshot of a cell phone

Description automatically generated

Figure 6 Result of f=600

A screenshot of a cell phone

Description automatically generated

Figure 7 Result of f=400

* Pada f=700 dihasilkan suara yang sama dengan f=300

A screenshot of a social media post

Description automatically generated

Figure 8 Result of f=700

A screenshot of a social media post

Description automatically generated

Figure 9 Result of f=300

* Pada f=800 dihasilkan suara yang sama dengan f=200

A screenshot of a social media post

Description automatically generated

Figure 10 Result of f=800

A screenshot of a social media post

Description automatically generated

Figure 11 Result of f=200

* Pada f=900 dihasilkan suara yang sama dengan f=100.

A screenshot of a cell phone

Description automatically generated

Figure 12 Result of f=900

A screenshot of a cell phone

Description automatically generated

Figure 13 Result of f=100

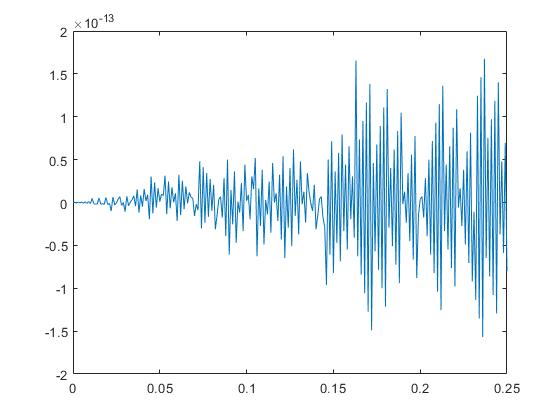


Figure 14. Result of f=500

Pada data di atas, dapat disimpulkan bahwa frekuensi yang memiliki kesamaan suara adalah penjumlahan dari kedua frekuensi yang menghasilkan frekuensi sampling. Selain itu juga dibuktikan dengan grafik **yang** sama pula.

Pada nilai f>Fn terjadi fenomena aliasing, dimana suara yang dihasilkan tidak lagi sesuai dengan nilai f, melainkan menghasilkan suara dengan frekuensi aliasnya. Dimana nilai dari frekuensi alias memenuhi persamaan sebagai berikut:

Untuk nilai x = frekuensi alias, sin(2\*pi\*f\*t) = nilai signal x, untuk f adalah variable yang diganti.

# Problem 4

%nomor 4

fs = 16000;

t = 0:1/fs:0.25;

f = [262,294,330,349,392,440,462,523]

%zeros matrix: matriks yg nilainya 0 semua

N = zeros(length(f)+1,length(t)); %lenght f syntax tidak wajib. berguna untuk mengetahui panjang suatu variabel yg tidak diketahui nilainya

%zeros formatnya baris dikurangi kolom

% ; akan disimpan di workspase, dan tidak diperlihatkan di command window

% kalau enggak, akan muncul di command window

for i = 1:length (f)

N (i,1:4001) = sin(2\*pi\*f(i)\*t);%kita akan mengisi matrix zeros pada baris ke i untuk semua kolom

%f, kita ingin mengakses f untuk tiap nilai

end

C = N (1,:);

D = N (2,:);

E = N (3,:);

F = N (4,:);

G = N (5,:);

A = N (6,:);

B = N (7,:);

C1 = N (8,:);

Nol = N (9,:);

Song1 = [C, E, C, E, F, G, G, Nol, B, C1, B, C1, B, G, Nol, Nol];

Song2 = [C, E, C, E, F, G, G, Nol, B, C1, B, C1, B, G, Nol];

Song3 = [C, Nol, E, Nol, G, Nol, F, F, G, F, E, C, F, E, C, Nol];

Song4 = [C, Nol, E, Nol, G, Nol, F, F, G, F, E, C, F, E, C];

Song = [Song1,Song2,Song3,Song4]

sound(Song,fs)

audiowrite('gundulgundulpacul.wav', Song, fs);

%sample rate terlalu kecil

%frekuensi sampling ada batasannya

Kemudian nilai Fs yang awalnya bernilai 16000 dirubah-rubah menjadi 10000, 2000, 1000, 900, 800, 700, 600, dan 500. Pada percobaan dengan nilai Fs antara 2000 – 16000 tidak terdengar adanya perpedaan pada audio yang dihasilkan. Mulai pada Fs = 1000 mulai terdpat perbedaan pada audio yang dihasilkan. Semakin rendah nilai Fs maka semakin tidak dikenali nada-nya, dan juga not tertinggi pada nadanya semakin rendah. Hal ini disebabkan oleh adanya aliasing pada sinyal ketika nilai frekuensi Nyquist lebih rendah dari not teringgi-nya. Dalam kasus ini, not tertinggi adalah c1 pada frekuensi 523Hz. Semakin kecil frekuensi sampling-nya, maka semakin rendah not tertinggi yang dapat direplikasi dengan benar.

# Problem 5

nb = 8;

ns = 120;

t = (0:19)/20; %untuk setiap elemen dr t dibagi dengan 20

f1 = 2\*pi\*t;

z = sawtooth(f1);

if z >= 1 %if ada dua keadaan

z = 1 - eps

end

f2 = (z+1)\*2^(nb-1);

z1 = floor(f2)%floor itu membulatkan ke atas

f3 = 2^(nb-1)

z2 = z1/f3;

f4 = ((2^nb)-1)/(2^nb);

zt = z2 - f4;

ze = z - zt;

figure(1)

stem(z,'b');

hold on;

stem(zt,'r');

hold on;

stem(ze,'g');

legend('Exact','Quantized','Error','Location','Southeast');

title(sprintf('Signal, Quantized signal and Error'));

hold off

figure(2)

spectrogram(z)

ketika di run pada matlab, maka hasil yang didapat adlaah sebagai berikut:

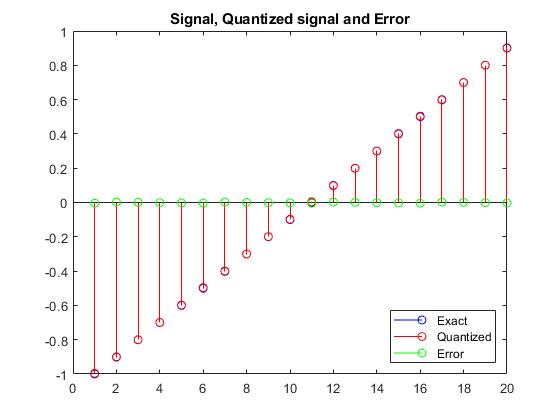


Figure 15. hasil sinyal ketika nb= 8 dan ns=120.

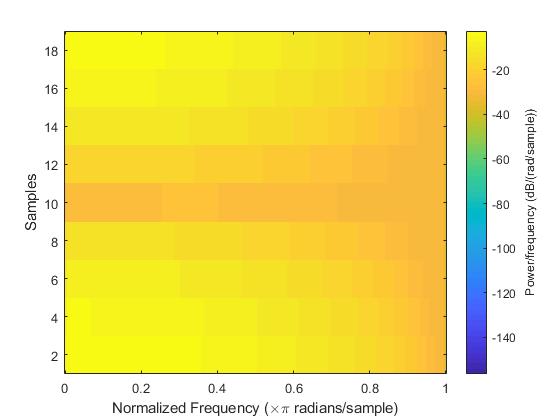


Figure 16. hasil spektrogram ketika nb=8 dan ns=120

Pada percobaan kali ini, dilakukan perubahan nilai nb dan ns. Kami mengganti dengan:

1. Nb=1 dan ns=230
2. Nb=2 dan ns=50
3. Nb=3 dan ns=500

Adapun hasil dari perubahan nb dan ns adalah sebagai berikut:

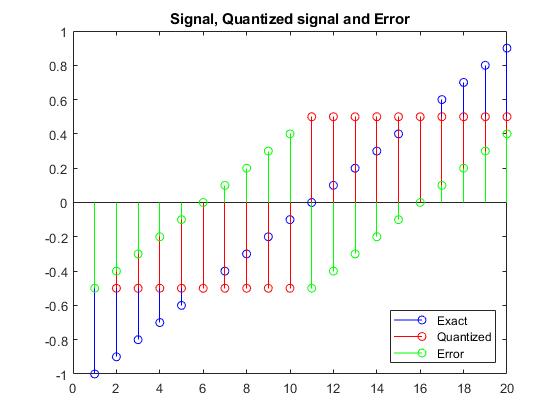


Figure 17. hasil sinyal ketika nb=1 dan ns=230

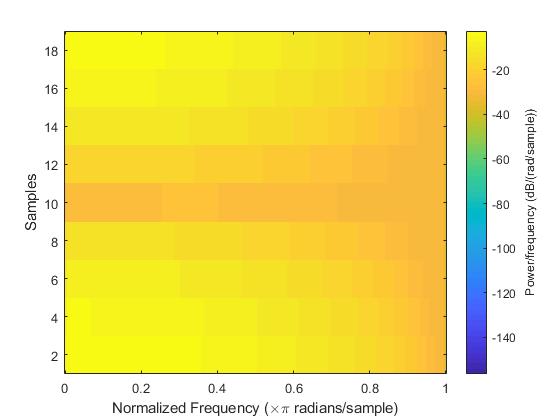


Figure 18. hasil spektrogram ketika nb=1 dan ns=230

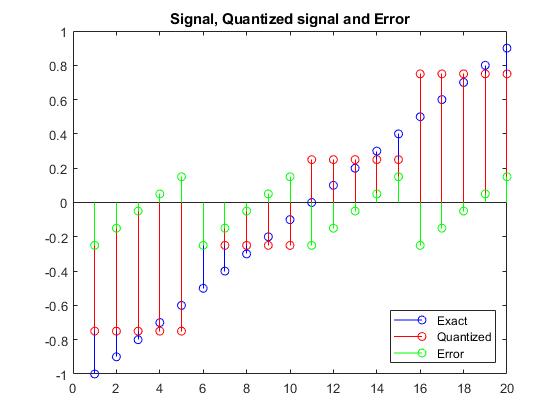


Figure 19. Hasil sinyal ketika nb=2 dan ns=50

Yang terjadi adalah ketika nilai n diperkecil maka error kuantisasi semakin besar, dan semakin diperbesar maka nilai error semakin kecil. Nilai m menentukan berapa banyak sampel-nya dan tidak berpengaruh pada besarnya error kuantisasi namun berpengaruh terhadap resolusi dari sinyal.

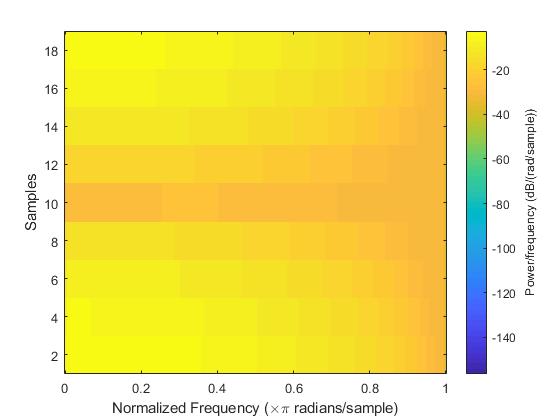


Figure 20. Hasil spektrogram ketika nb=2 dan ns=50

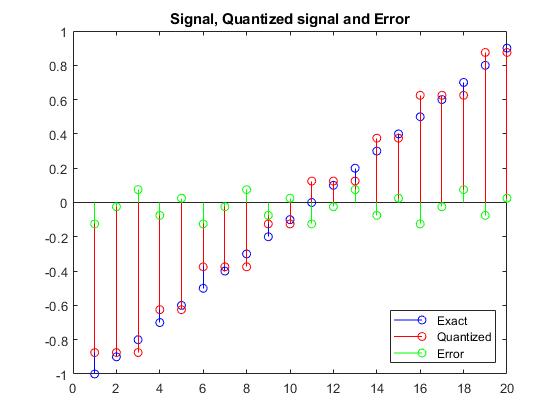


Figure 21. hasil sinyal ketika nb=3 dan ns=500

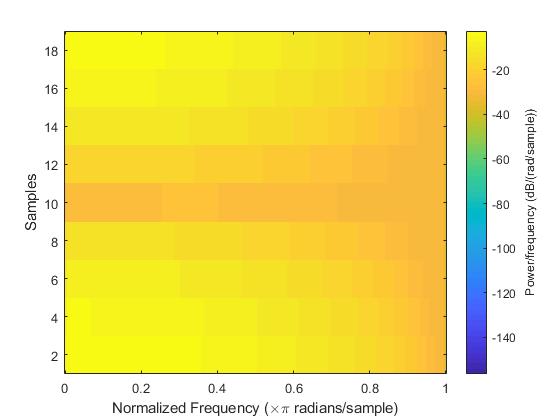


Figure 22. hasil spektrogram ketika nb=3 dan ns=500

Pada percobaan kali ini, audio diubah menjadi 3 variasi, yaitu z=sin(f1), z=cos(f1), dan z=tan(f1). Hasil sinyal yang dihasilkan adalah sebagai berikut:

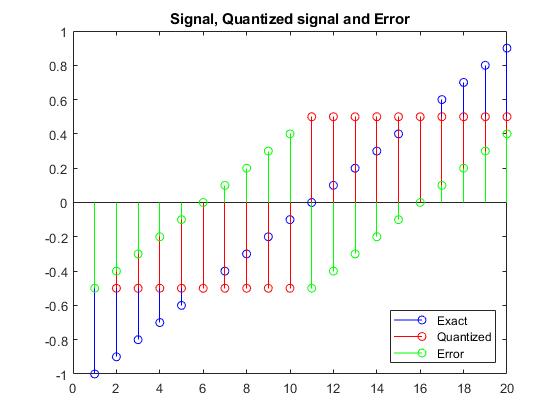


Figure 23. grafik pada saat z=sin(f1)

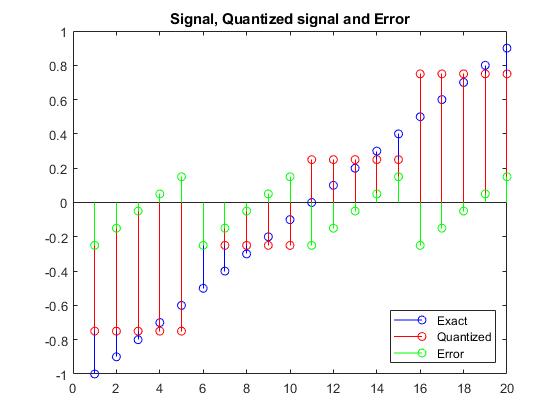


Figure 24. grafik pada saat z=cos(f1)

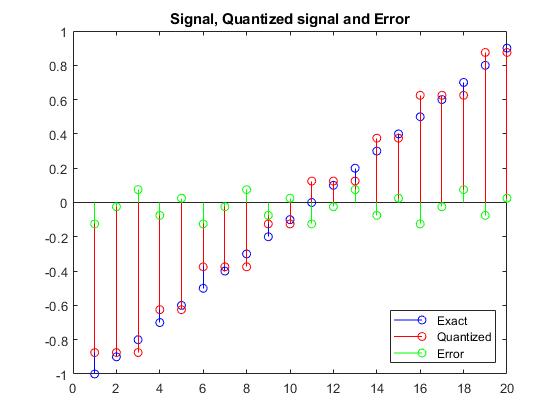


Figure 25. grafik pada saat z=tan(f1)

Pada figure 23, 24, dan 25 didapatkan perbedaan sinyal karena perbedaan tone yang dihasilkan, karena pada z diubah menjadi sin, cos, dan tan yang menghasilkan berbedaan tone.

Adapun spectrogram yang didapatkan adalah sebagai berikut:

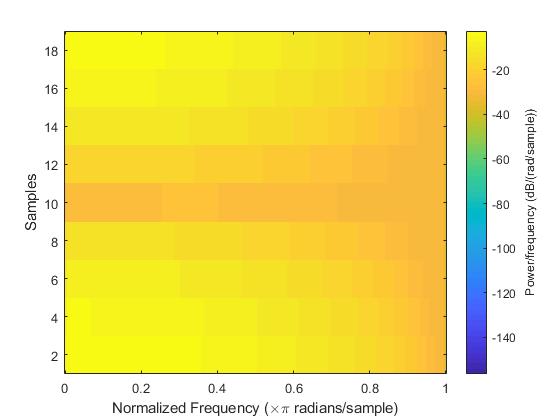


Figure 26. Spectrogram pada saat z=sin(f1)

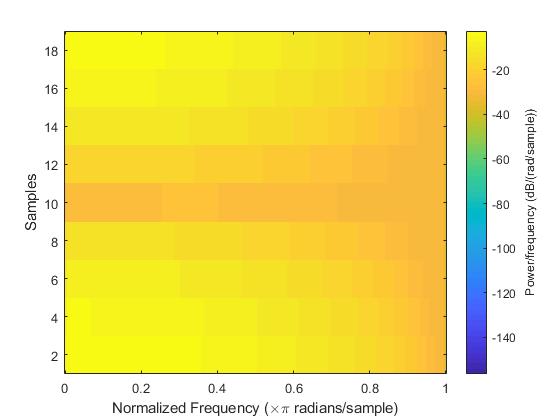


Figure 27. . Spectrogram pada saat z=cos(f1)

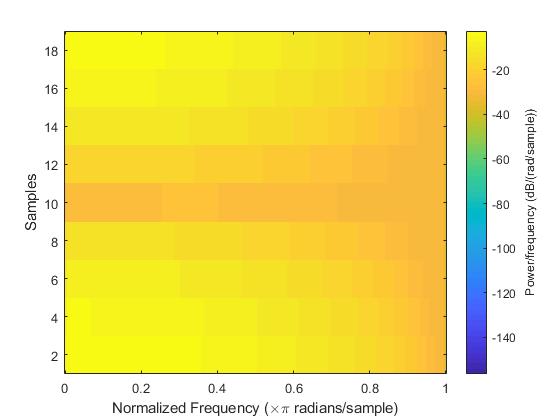


Figure 28. . Spectrogram pada saat z=tan(f1)

Dari percobaan tersebut, diketahui bahwa semakin kecil nilai n dan m, maka semakin buruk kualitas audio karena disebabkan oleh adanya distorsi. Oleh sebab itu, untuk mendapatkan audio digital yang seakurat mungkin, dibutuhkan nilai n dan m yang besar.

# Problem 6

Source code

%problem 6

clc; %clean screen

clear all; %clear removes all variables from the current workspace

close all; %deletes all figures whose handles are not hidden

Nsec = 3; % Number of seconds of the sound file to process

Bits = 12; % Number of bits for requantization

[x,R] = audioread('Pinball Machine.wav'); % Use any .wav file you like

info = audioinfo('Pinball Machine.wav');

Nbits = info.BitsPerSample;

x = x(1:Nsec\*R)\*2^(Nbits-1); % Just keep the first Nsec of the file and normalize to integer levels

soundsc(x,R) % Play back the original

pause(Nsec + 0.5)

xq = round(x/2^(Nbits-Bits)); % Scale wave file up to +- 2^nbits

soundsc(xq,R) % listen to the requantized sound

start = 1; % Select the part of the file to plot

stop = length(x); % If stop = length(x) the entire file is plotted

len = stop - start;

subplot(211) %Create axes in tiled positions

plot(x(start:stop),'k.'); hold on

plot(x(start:stop),'r'); axis([0 len -2^(Nbits-1) 2^(Nbits-1)])

grid on; title('Original audio') %Display or hide axes grid lines

xlabel('Sample Number'); ylabel('Level')%label x-axis

subplot(212) %Create axes in tiled positions

plot(xq(start:stop),'k.'); hold on %2-d line plot

plot(xq(start:stop)); axis([0 len -2^(Bits-1) 2^(Bits-1)])

grid on; title('Re-quantized audio')%Display or hide axes grid lines

xlabel('Sample Number'); ylabel('Level')%label x-axis

spectrogram(x(start:stop)) %Spectrogram using short-time Fourier transform of the input signal, x

figure(2)

spectrogram(x(start:stop))

figure(3)

spectrogram(x(start:stop))

Kemudian dilakukan percobaan dengan merubah-rubah besar bits untuk re-quantization. Pada re-quantization menjadi 12 bit didapat hasil sebagai berikut:

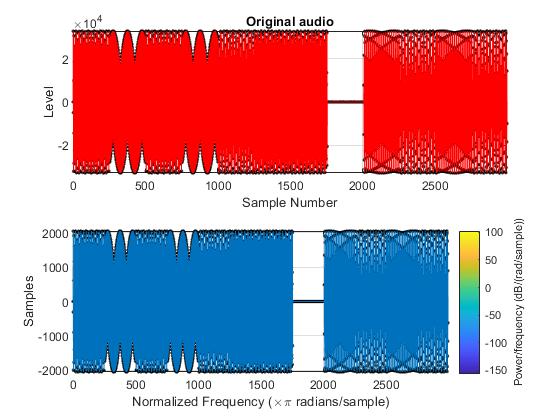


Figure 29 Waveform dari audio sebelum dan sesuda re-quantization.

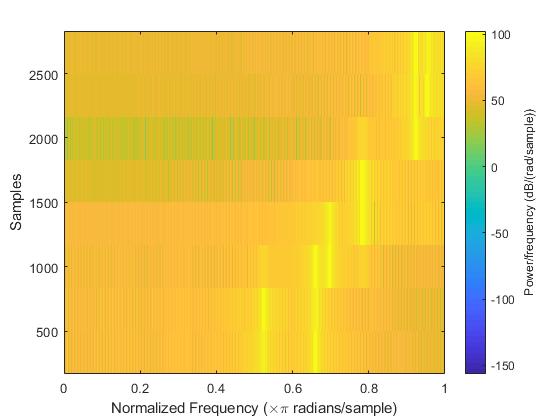
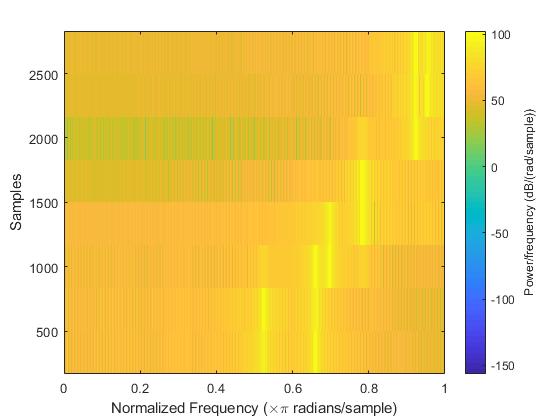


Figure 30 Spectrogram dari audio sebelum dan sesudah re-quantization.

Dari hasil percobaan diketahui bahwa setelah dilakukan re-quantization dengan mengubah bit menjadi lebih rendah, muncul distorsi pada audio yang menyebabkan menurunnya kualitasnya.

# Problem 7

Source code

%problem 7

clc; %clean screen

clear all; %clear removes all variables from the current workspace

close all; %deletes all figures whose handles are not hidden

[y,fs] = audioread('Figura Renata-Elegi.mp3'); %Read audio file Figura Renata-Elegi.mp3

audiowrite('Figura Renata-Elegi.wav',y,16000); %create wav Figura Renata-Elegi

[Y, Fs] = audioread('Figura Renata-Elegi.wav');

Fs = 16000; %frequency sampling

sound(Y, Fs)%play sound Figura Renata-Elegi.wav

Kemudian dilakukan percobaan dengan merubah-rubah nilai Fs. Pada Fs bawaan audio yaitu 44100, audio yang dimainkan identic dengan audio dari file .wav. Ketika dirubah-rubah maka kecepatan dari audio menjadi berubah, semakin berkurang nilai Fs, maka tempo lagu menjadi semakin lambat, sebaliknya maka tempo lagu menjadi semakin cepat.