Guitar Chord Noise Filter using Finite Impulse Response with Low-pass Filter and 48000 Hz Sampling Rate

A Case Study Presented to

Computer Engineering Department

Pamantasan ng Lungsod ng Maynila

In Partial Fulfillment of the Requirements for the Course

Digital Signal Processing

By:

Brix Ian D. Relano

Ravid Phelps M. Reyes

Aaron Jacob A. Sy

Submitted to:

Engr. Eufemia A. Garcia

January 2022

CHAPTER I
The Problem and Its Background

1.1 Background of the Study

Brittanica defines a wave as the propagation of disturbances in a regular and organized way. Waves originate from a certain point and create a disturbance that has a determined frequency and wavelength. Sound waves are classified as mechanical waves due to the fact that they require air as a medium to travel through as opposed to electromagnetic waves that can travel through a vacuum. The human ear can perceive sounds from 20Hz to 20kHz (Purves, Augustine, Fitzpatrick, et. al. 2001), allowing for any sound under that specific frequency range to be heard.

Fink (2019) states that sound and noise are used interchangeably in fields such as acoustics, electronics, and physics, but these terms have different meanings depending on their usage. The vibrations that travel through a certain medium and can be heard when they reach an individual's ears are called sound, and unwanted sound is defined as noise. Because of the wide range of the audible frequency range, wanted sound and unwanted noise can merge together making the sound and the listening experience unpleasant.

Sound waves can also be considered as signals that can be observed digitally in order to gather more information aside from the sound itself. The digitization of sound waves as signals opens the possibility of processing and manipulating recorded sound, and improves these soundwaves through the use of filters and functions.

It is impossible to isolate the desired sound to be recorded. This is especially true in areas with ambient noise and ventilation. Recording artists that do not have the equipment to properly record audio and remove noise find this a very difficult problem to solve. Several practical measures can be done by the artist to lessen the noise in recorded audio but without digital signal processing being applied to the audio it does not entirely eliminate it.

In order to solve this problem, several filtering algorithms had been used to filter recorded audio and remove unwanted noise. With the rise of technology and improvement in computing power, several filtering algorithms became more effective and efficient in filtering noise in different frequency ranges.

1.2 Statement of the Problem

Recorded audio can contain both wanted sound and unwanted noise. It is very difficult to isolate audio to be recorded as not all the variables during audio recording can be controlled therefore the resulting audio output can have a mixture of noise and desired sound. Eliminating noise when recording audio is very difficult without active filtering equipment. Artists who are starting out do not have access to such technology making it very difficult for them to record instrument audio without noise. The development of a Guitar Chord Filter can address this problem by filtering the noise in a recorded audio file leaving out the instrument audio allowing the recording to sound better and provide a cleaner tone to the recorded guitar.

1.3 Objectives

General Objective

To create a Guitar Chord Filter using the Finite Impulse Response and Low-pass filter with a 48000 Hz sampling rate.

Specific Objectives

Specifically, the researchers aim to do the following:

1. Filter high frequency background noise from recorded audio by utilizing the Finite Impulse Response and Low-pass Filter;
2. Output the filtered audio in a .wav file; and
3. Compare the filtered output to recorded audio without noise to determine the effectiveness of the developed algorithm.

1.4 Significance of the Study

A Guitar Chord Noise Filter will be able to reduce background noise from an audio clip with a guitar tune.

Specifically, this study will be beneficial for the following:

Recording Processes

A Guitar Chord Filter will be able to improve the recording process by allowing for cleaner recordings even if there is background noise. This filter may also bring more accessibility to individuals who require a clean recording but have no access to a noise-free recording studio.

Students

This study will help students gain a better understanding of digital signal processing specifically, audio processing. It will also help students understand how Finite Impulse Response and Low-pass Filter work and how it affects signals and sound waves.

Future Researchers

The Fast Fourier Transform and the Finite Impulse Response are techniques done when analyzing and processing digital signals. Future researchers and Digital Signal Processing students will be able to observe the capabilities of these functions and apply them to their own research.

1.5 Scope and Limitations

This study will focus on filtering noise from recorded audio with a .wav file type that contains a recognizable guitar chord. Specifically, the developed algorithm will filter ambient noise above the frequency of 1000 Hz. Speech and other non-guitar sounds from the recorded audio will not be paid attention to and will be filtered to emphasize the sounds made by the guitar.

The algorithm will be developed using the open-source software, Octave together with its built-in and imported functions from *octave* and *signal* packages.

The group only tackled the topics that were discussed in the course Digital Signal Processing such as Fourier transform, frequency response, low-pass filters, finite impulse response, and the likes. Any further complex filters and concepts are not tackled.

CHAPTER II
Review of Related Literature

Audio Processing

According to Zölzer (2008) in his book Digital Audio Signal Processing, this process is done during recording and storing audio signals, for sound mixing and production. These audio signals start as analog signals and are required to be digitized first before any complex processing is done. Recording speech or music first requires the analog signal to be digitized, then mixed in a digital mixing console, and finally stored in a digital storage medium. This is the typical flow of audio and what happens to it when it is recorded.

Advanced audio processing techniques include the usage of different audio filters and methods. Understanding how these filters work and how to properly utilize them requires knowledge of the fundamentals of digital signal processing.

Analog to Digital Signal Conversion

Zölzer (2008) also states in his book that conversion of a function with respect to time into a sequence of numbers is known as analog-to-digital conversion. During AD conversion, 2 main processes are done, time-sampling and quantization. Time sampling is done by setting a fixed time interval and obtaining the voltage level of the signal at each time interval which can now be stored discretely as vectors. Quantization comes after wherein the sampled values are converted into a signal that allows the analog signal to be processed digitally.

Fast Fourier Transform

Fast Fourier Transform (FFT) is a more efficient method in solving the Discrete Fourier Transform (DFT) of a wave. It is a better alternative to using continuous Fourier transform as FFT is able to compute values that DFT computes while consuming less computing time (Bergland, 1969). Compared to DFT which involves O(n2) operations, FFT is able to reduce these operations into O(n log(n)) enabling it to be used for a wide variety of applications that include audio and image compression, video streaming, satellite and cellular network communications. Given the ability of FFT to reduce the computing time of complex signals, FFT libraries have been built into almost every device and operating system executing digital signal processing operations (Brunton and Kutz, 2017).

In the study by Liu and Morgan (2006), FFT was used to remove systemic noise in Enhanced Thematic Mapper Plus (ETM+) Image pairs on an earthquake event to identify earthquake-caused terrain displacements. FFT was an essential part of imageodesy which is a technique used for detecting and measuring feature shifts between two images. Images obtained from the earthquake event were contaminated with vertical and horizontal stripping noise which made the process of comparing two images very difficult. In the study, a semiautomatic selective filtering procedure was designed which isolates and masks the diagnostic frequency features of

the noise through FFT frequency-domain analysis. An FFT frequency-domain adaptive filter was then also designed to remove vertical wavy striping patterns present in the images to remove noise. The study concluded with successfully removing the noise in the images which enabled the images to be studied through imageodesy.

Finite Impulse Response Low-pass Filter

According to Salih (2017), filters are defined as networks that process signals through frequency-dependent conditions. It can be compared to the frequency-dependent nature of the impedance of capacitors and inductors. Whenever the frequency is changed, the reactive impedance and the voltage diverter ratio also changes. A filter produces a frequency-dependent change in the input/output transfer function defined as the frequency response.

Moreover, digital filters are used to remove noise present in a digital signal. It implements a mathematical algorithm in order to obtain the desired filter transfer function. Digital filters are the basic filters used in Digital Signal Processing. It is also known as a Finite Impulse Response (FIR) Filter as the output of the filter settles to zero within a finite time (Kumar et al., 2017). FIR filters operate opposite to the Infinite Impulse Response (IIR) Filter as the IIR filter operates for a long period of time without the output reaching zero (Salih, 2017).

Based on the study by Salih (2017), FIR filters offer various advantages over IIR filters. These advantages include its ability to have an exactly linear phase, its stability, its design methods being mostly linear, its ability to be realized efficiently in hardware and the filter start-up transients have a finite duration. Salih (2017), utilized the FIR filter along with the low-pass filter in order to reduce noise in audio recordings specifically from signals with different frequencies and ripple factors. A low pass filter filters out signals with a frequency above the cutoff frequency and passes signals with a frequency below the cutoff frequency. Using the low-pass filter proved to be a very effective filter in removing noise from audio as concluded by Salih (2017).

2.2 Synthesis

Title	Reference	Summary	Synthesis
Audio Noise Reduction Using Low Pass Filters	Salih, A.O.M. (2017) Audio Noise Reduction Using Low Pass Filters. Open Access Library Journal, 4: e3709. https://doi.org/10.4236/oalib.1103709	In the study by Salih (2017), Low pass filters which are components of FIR filters were used to remove noise from audio. The study also utilized using .wav format in recording audio filtering two recorded audio samples. Throughout the study, it was concluded that utilizing a low pass filter removes noise effectively.	The study that the researchers conducted also utilized the low pass filter in removing noise from the audio recordings. The researchers gathered a larger sample size and specified the subject by obtaining audio recordings of the basic chords of a guitar in .wav format. The researchers also recorded audio recordings without noise to compare the filtered audio with and measure the effectiveness of the developed filter.
FFT Selective and Adaptive Filtering for Removal of Systematic Noise in ETM+ Imageodesy Images	J. G. Liu and G. L. K. Morgan, "FFT Selective and Adaptive Filtering for Removal of Systematic Noise in ETM+ Imageodesy Images," in IEEE Transactions on Geoscience and Remote Sensing, vol. 44, no. 12, pp. 3716-3724, Dec. 2006, doi: 10.1109/TGRS.2006.881752.	In the study, Liu and Morgan (2006) utilized Fast Fourier Transform (FFT) to design a filter that will isolate and mask diagnostic frequency features of the noise and isolate vertical wavy striping patterns in an image. The study concluded with successfully filtering the noise in the images enabling them to be studied and compared.	In the study that the researchers conducted, Fast Fourier Transform was utilized to convert the signal from the time domain to frequency domain which enabled it to be filtered eliminating the noise in the recorded audio.

Table 2.1. Table of Syntheses

2.3 Conceptual Framework

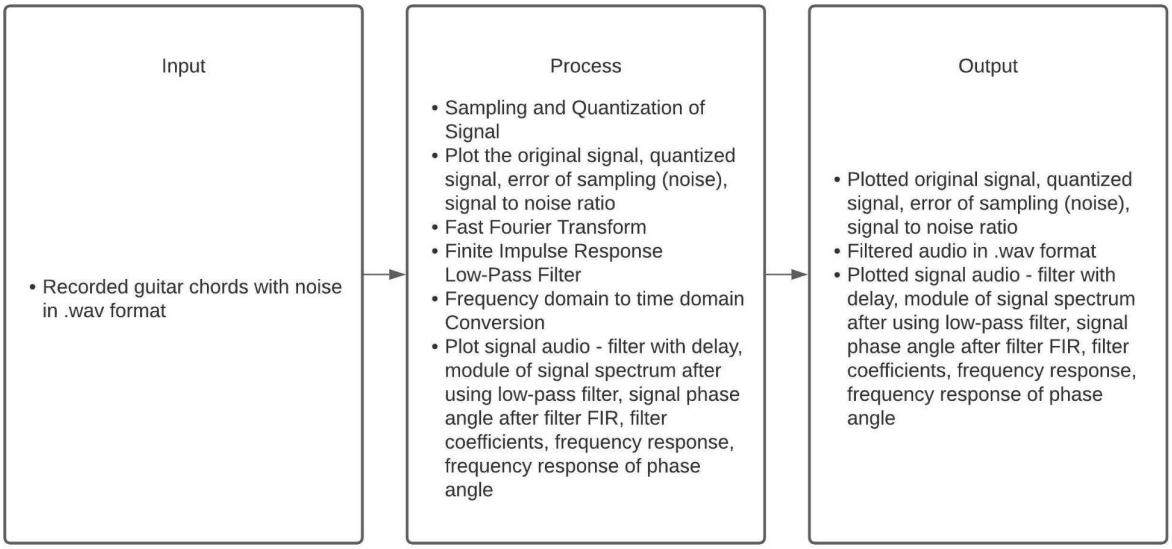


Figure 2.1. Input Process Output

The input stage of the study includes the recorded guitar chords with noise in .wav format. The recorded audio will be filtered throughout the process as noise will be removed through the filtering algorithm developed by the researchers.

The recorded audio goes through different processes in order to successfully filter the noise. First, it goes through sampling and quantization then a fast fourier transform before it is filtered through the use of a finite impulse response low-pass filter. After the filtering of the noise, the signal is converted back to the time domain from the frequency domain. Several outputs are plotted through figures during the process in order to illustrate the changes in the signal during the filtering process.

The output of the study includes the plotted signals during the filtration process. It also includes the filtered guitar chord audio in a .wav file. The output, along with its spectrogram and waveform is compared to a recorded guitar chord audio without noise to compare the effectiveness of the filtering algorithm.

CHAPTER III
Methodology

3.1 Research Design

The noise filtration process is composed of three (3) main processes:

Signal Sampling and Quantization

The signal is quantized iteratively by looping its process to determine the appropriate resolution for the sampling of the signal. The process tries to determine the needed quantization level of the audio signal using the formula 2^n-1 with n being the counter in the loop. The quantization level is also used to round the signal quantized value. This process also shows noise compared to the original signal the filter is currently generating. Consequently, the signal-to-noise ratio of the quantization process can be determined and will serve as a basis if the error of sampling is acceptable. Then it is visualized by plotting the original signal, quantized signal, the error of sampling which is the noise, and the signal to noise ratio in different color representations and in time and amplitude axes.

Signal Conversion (Time to Frequency Domain)

Using the MATLAB function *fft*, the Fast Fourier Transform of the signal on the frequency from 0 to 48000 hertz is processed through the frequency range parameter. Then its absolute value is taken for the values to be real numbers, therefore can be plotted to the frequency domain.

Application of Finite Impulse Response Low-pass Filter

The Finite Impulse Response Low-pass filter is utilized to allow a certain amount of frequency (in this case it is 1000 hertz), and attenuate higher frequencies to 6 decibels. It is done by the *fir1* function together with its parameters, the order (101), the normalized frequency (0.04), and the type of filter (low). Then the *filter* function is used to convert it back to a time-invariant difference equation, to be plotted with the original signal for comparison.

Alongside these main processes are the sub-processes and functions used to handle the intricacies of the noise filtration process.

Auload

The auload function loads the .wav audio file and stores its audio signal and the sampling frequency in each respective variable.

Linspace

Stores the vector of samples from the base length zero (0) to the limit length (48000) to a variable for quick access in plotting the signal.

Ausave

The ausave function saves the .wav audio file by creating a new audio file with its respective file name and file type.

Signal Amplification

The signal was amplified by 10 to increase the intensity of the signal for a more accurate and more rounded calculation.

Signal Deviation

The signal to noise ratio was calculated through this formula: signal strength divided by the signal noise, then converted to decibels using logarithm.

Frequency Response

The frequency response of the signal is obtained by using the *freqz* function together with its parameters, filter coefficients, and sampling frequency.

3.2 Audio File Recording Process

The audio files were recorded in two sets, the first is the guitar chords with noise (electric fan background) and the next is the clean version without the noise. These will be used as an input for the noise filtration process and a comparison between the filtered audio file and the clean audio file.

3.3 Program Flowchart

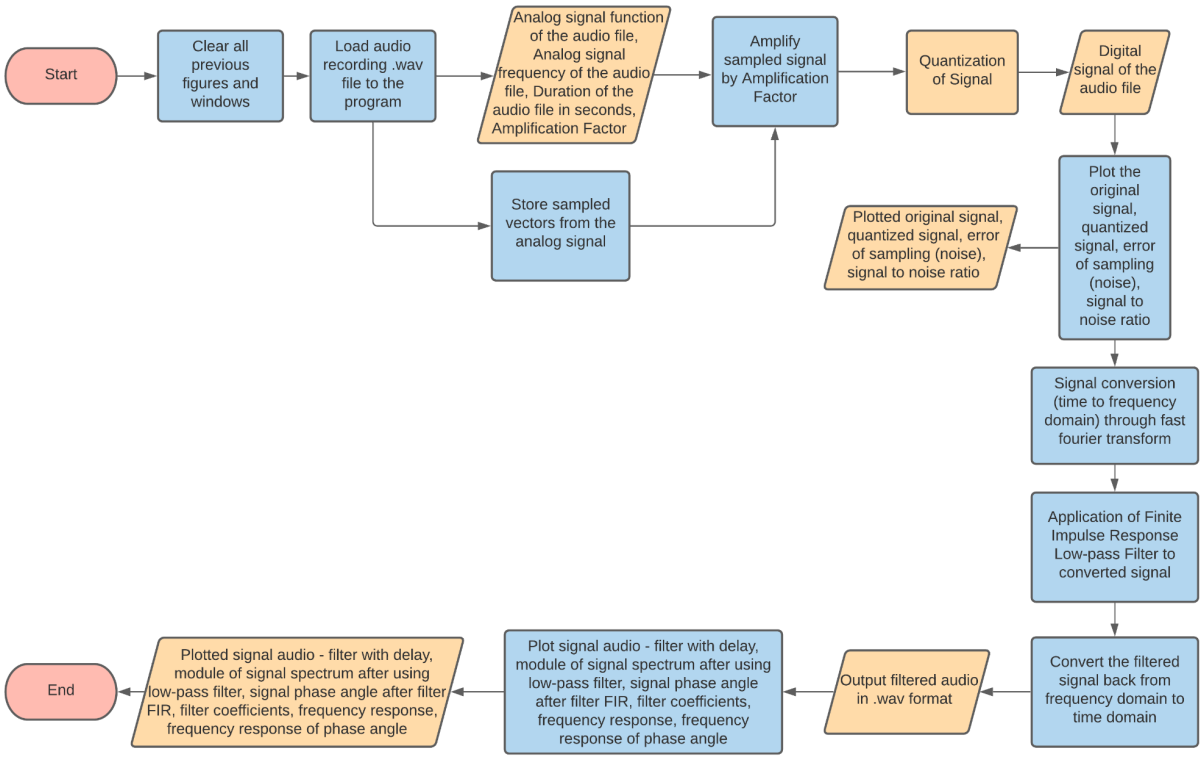


Figure 3.1. Program Flowchart

3.4 Block Diagram

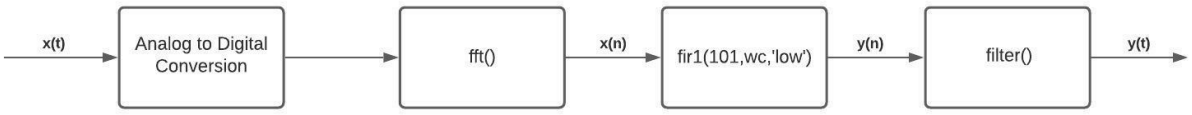


Figure 3.2 Block Diagram

3.5 Program Listing

The figures below contain the program listing for the A chord Guitar Noise Filter. The code for other chords is similar except for naming conventions based on the name of the chord or the input and output file names of the program.

FIR_A.m

```
## beginning - clearing all previous figures, and windows etc...
clear all;
close all;
clf;
pkg load signal;


## loading of an audiofile to project
## s- signal## fp - frequency of a file
[s, fp] = auload('A.wav');


##  audio file is ~1,78s long
## 1,78 * 44100 ~78.5 kHz

t_length = length(s);

## vector of next samples from 0 to ~78500
t = linspace(0, t_length/fp, length(s) );


## amplification factor
amp = 10.0;
## multiply figure by amp factor
s(1:t_length) = amp * s(1:t_length);


## number of bits of filter - iterative loop
BITS = 5;

##for loop describing good enough resolution of sampling
for n = 1:BITS;


  a =2^n - 1;

  ## rounding a signal quantized value using "a" parameter
  signal_quantized = round( s(1:t_length) * a) / a;


  ## noise signal
  noise = s(1:t_length) - signal_quantized;


  ## ploting signals: original, quantized and error of sampling
  ## -b: blue - original signal
  ## -g: green - quantized signal
```

Figure 3.3. Code for "A Chord" Audio Filtering 1st Part

FIR_A.m

```
48    ## -r: red - noise signal
49
50    plot(t, s(1:t_length), '-b; Original signal;',
51          t, signal_quantized, '-g; Quantized signal;',
52          t, noise, '-r; Error of sampling - noise;');
53
54    grid on;
55    xlabel ('Time [s]');
56    ylabel('Amplitude [-]');
57    title({"Singular step of quantisation number:", int2str(n)});
58
59
60
61    ## (n, :) - all "n" rows will be read in their entirety
62    ## std - standard deviation
63    ## logarithmic scale multiplied by 20
64    signal_deviation(n,:) = 20 * log10( std( s(1:t_length) ) / std(noise) );
65
66
67
68    ##int2str - converts integers to strings
69    print(["A_Quantized_signal ", int2str(n), ".png"], "-dpng", "-color");
70
71    signal_rounded = round(s * a) / a;
72
73    ## audio save function from folder
74    ausave(['A', int2str(n), '.wav'], signal_rounded, fp);
75
76
77  end
78  ##
79
80
81
82  ################################################################################
83  ################################################################################
84  ##SECOND FRAME WITH PLOTS
85
86
87  ## plot of signal in time multiplied by amp
88  ## figure = new window with plot
89
90   subplot(3,3,1);
91   plot(t, s(1:t_length), '-b');
92
93   title('Original signal');
94   xlabel('Time [s]');
```

Figure 3.4. Code for "A Chord" Audio Filtering 2nd Part

FIR_A.m

```
95  ylabel('Amplitude [-]');
96  grid on;
97  hold on;
98
99
100
101
102 subplot(3,3,3);
103 plot(signal_deviation, 'r*');
104 hold on;
105 plot(signal_deviation ,'-b');
106 hold on;
107 grid on;
108
109 title({'SNR signal to noise ratio', 'dependent on bits of filter'});
110 xlabel({'Bits of filter -', 'resolution of quantization'});
111 ylabel({'value of SNR', 'in logaritmic scale [dB]'});
112
113
114
115 n_poly = 1:1:BITS;
116
117
118 ## method polyfit returns roots of equation with best fit
119 ## symbol ' reverts horizontal vector to vertical matrix
120 ## 1 means logiczal vector setting usage of roots to true
121 polyfit(n_poly', signal_deviation, 1);
122
123
124
125 ## overwriting (copying) final audio file to next one
126 ## one with o number higher of 1
127 file1 = ['A' , int2str(BITS), '.wav'];
128 file2 = ['A' , int2str(BITS + 1), '.wav'];
129 copyfile(file1, file2);
130
131
132 ## spectrum analisys of audio file after quantization and sampling
133 [s_final, fp] = auload(file2);
134
135 ## first half of Fourier spectrum
136 N = fp / 2;
137
138
139 ## function determines  to which top to the power of raise a number to
140 ## be greater than N
141 ## you could use logarithm in this place
```

Figure 3.5. Code for "A Chord" Audio Filtering 3rd Part

FIR_A.m

```
142     Nf_help = 1;
143     help = 2;
144     while(help < N)
145         help = help* 2;
146         Nf_help++;
147     endwhile
148     ##
149
150 ## power two to a number - FFT is faster that way
151 Nf = 2^Nf_help;
152
153 ## Nf2~22kHz
154 Nf2 = Nf/2 + 1;
155
156
157 ## vector of equaly set probes
158 f = linspace(0, fp/2, Nf2);
159
160
161 ##subplot of only a signal after iterative sampling
162 subplot(3,3,4);
163 plot(t, s_final(1:t_length));
164 title({'Signal quantized','-without noise'});
165 xlabel('Time [s]');
166 ylabel('Amplitude [-]');
167 hold on;
168 grid on;
169
170
171 ## fourier transfomate of final signal on frequanzy from 0 to 48 kHz
172 s_fft = fft(s_final, Nf);
173
174
175 ## absolute value of our final signal after fft
176 s_fft_abs = abs(s_fft);
177
178
179 subplot(3,3,5);
180 plot(f, s_fft_abs(1:Nf2) );
181 title({"Module of a signal after", "iterative sampling"});
182 xlabel('Frequency [Hz]');
183 ylabel({'Module of a signal after', 'iterative sampling [-]'});
184 box off; grid on;
185
186
187
188 ## phase of signal iterative sampling
```

Figure 3.6. Code for "A Chord" Audio Filtering 4th Part

FIR_A.m

```
s_fft_angle = angle(s_fft);

subplot(3,3,6);
plot(f, s_fft_angle(1:Nf2));
title({"Phase of signal after", "iterative sampling"});
xlabel('Frequency [Hz]');
ylabel('Phase angle of a signal [rad]');
box off; grid on; axis tight;



## noise signal plot
subplot(3,3,7);
plot(t,  noise(1:t_length) );
title('Noise of sampling');
xlabel('Time [s]');
ylabel('Amplitude [-]');
hold on; grid on;


## FFT for noise signal
noise_fft = fft(noise, Nf);

noise_fft_abs = abs(noise_fft);


## Module of signal spectrum
subplot(3,3,8);
plot(f, noise_fft_abs(1:Nf2));
title("Module of noise spectrum");
xlabel('Frequency [Hz]');
ylabel('Module of noise spectrum [-]');
grid on; hold on;


## angle of noise signal
noise_ftt_angle = angle(noise_fft);

subplot(3,3,9);
plot(f, noise_ftt_angle(1:Nf2));
title("Phase angle of noise");
xlabel('Frequency [Hz]');
ylabel('Phase angle of signal [rad]');
grid on; hold on;
```

Figure 3.7. Code for "A Chord" Audio Filtering 5th Part

FIR_A.m

```
## saving picture with 1600x900 pixels
print(['A_Fig. 1 - Plots of signals original sampled and noise.png'], "-dpng", '-S1600,900');

################################################################################
################################################################################
##THIRD PICTURE WITH PLOTS

## FILTR FIR
freqCut = 1000;

## frequency normalized, fp ~48kHz
wc = freqCut / (fp/2);

## FIR filter coefficients
firCoeff = firl(101, wc, 'low');


figure;

subplot(2,3,4);
## filar-type plot
stem(firCoeff);
title("Filter coefficients");
hold on;

## signal audio after sampling - filtered with delay
s_filtered = filter(firCoeff, 1, s_final);


subplot(2,3,1);
plot(t, s_filtered, '-r',
     t, s_final, '-b');

title("Signal audio - filter with delay: \n red - sig. original, \n blue - sig. filtered]");
xlabel('Time [s]');
ylabel('Signal [-]');
grid on; hold on;


## FFT of filtered signal
s_filtered_fft = fft(s_filtered, Nf);


## signal module after filter FIR
s_filtered_fft_abs = abs(s_filtered_fft);

subplot(2,3,2);
```

Figure 3.8. Code for "A Chord" Audio Filtering 6th Part

FIR_A.m

```
283 plot(f, s_filtered_fft_abs(1:Nf2));
284 title({"Module of signal spectrum after" , "using low-pass filer"});
285 xlabel('Frequency [Hz]');
286 ylabel('Module of signal spectrum after reduction [-]');
287 grid on; hold on;
288
289
290
291 ## Signal phase angle after filter FIR
292 s_filtered_fft_angle = angle(s_filtered_fft);
293
294 subplot(2,3,3);
295 plot(f, s_filtered_fft_angle(1:Nf2));
296 title({"Signal phase angle", "after filter FIR"});
297 xlabel('Frequency [Hz]');
298 ylabel('Signal phase angle [rad]');
299 grid on; hold on;
300
301 s_filtered_mono = s_filtered(:,1);
302 ausave(['A_filtered.wav'], s_filtered_mono, fp);
303
304
305
306
307 ## frequency response of filter FIR
308 [H,f] = freqz(firCoeff, 1, 2^Nf_help, fp);
309
310 subplot(2,3,5);
311 plot(f, abs(H));
312 title({"Frequency", "response"});
313 xlabel('Frequency [Hz]')'
314 ylabel('Frequency response [-]');
315 grid on; hold on;
316
317 ## frequency response phase angle of filter FIR
318 subplot(2,3,6);
319 plot(f, angle(H));
320 title({"Frequency response of", "phase angle"});
321 xlabel('Frequency [Hz]');
322 ylabel('Phase angle of signal [rad]');
323 grid on; hold on;
324
325
326 ## save plots as a picture
327 print(['A_Fig. 2 - Plots of filter FIR.png'],"-dpng", '-S1600,900');
```

Figure 3.9. Code for "A Chord" Audio Filtering 7th Part

```
##############################################################################
##############################################################################
## LAST FRAME WITH PLOTS


figure;
freqz(firCoeff, 1, 2^Nf_help, fp);

print(['A_Fig. 3 - Plots of frequency response of filter FIR.png'],"-dpng", '-S1600,900');
```

Figure 3.10. Code for "A Chord" Audio Filtering 8th Part

CHAPTER IV
Results and Discussion

The following are the plots and figures showing several steps to achieve desired results of noise filtration using FIR Low-pass filter, the comparison between the original signal and filtered signal, phase angle, signal to noise ratio, and the frequency response of signals. The group found out that the results are almost identical for the chords as they have similar waveforms. Therefore, the group decided to show one chord (A Chord) for the results and the other plots can be referred to the appendices.

A Chord Noise Filtration - Quantization

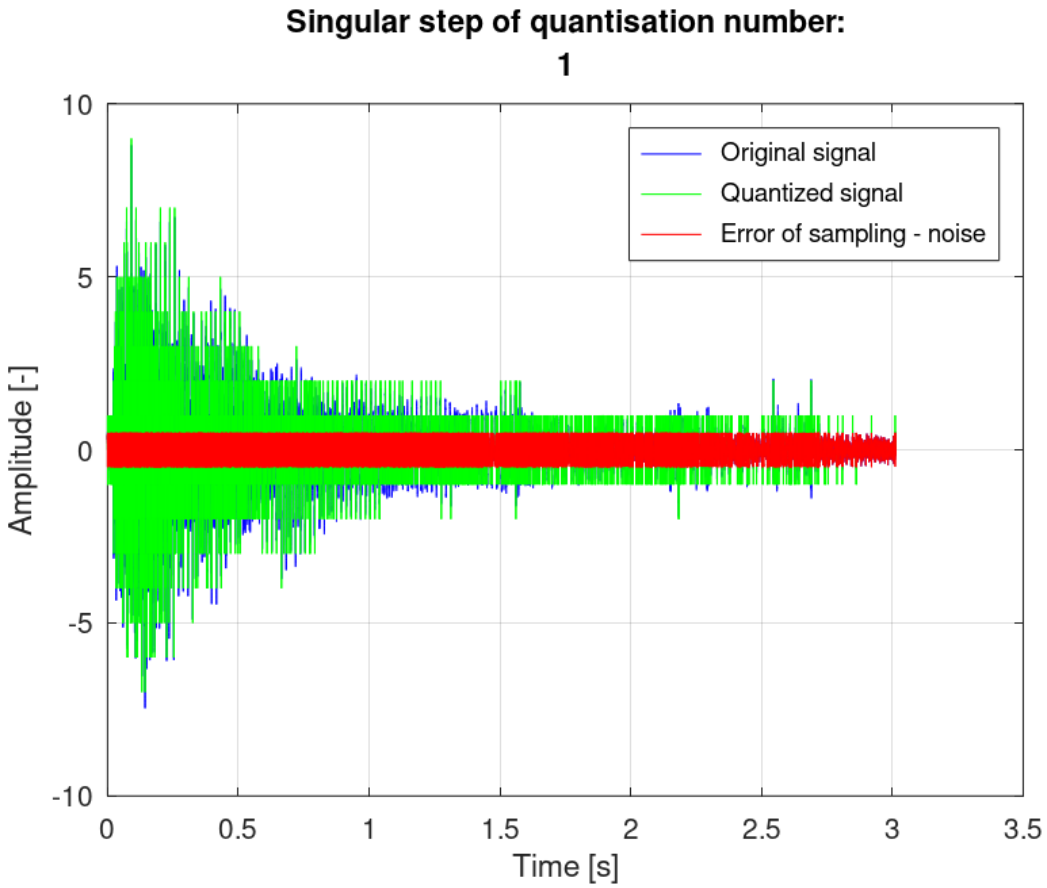


Figure 4.1. First iteration of Quantization Process

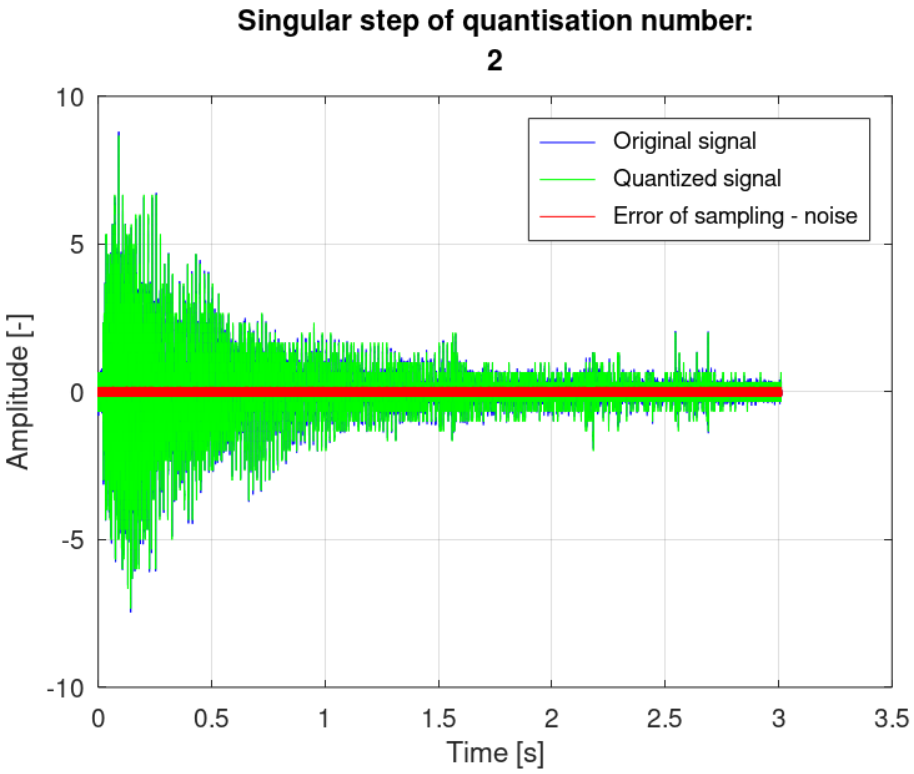


Figure 4.2. Second iteration of Quantization process

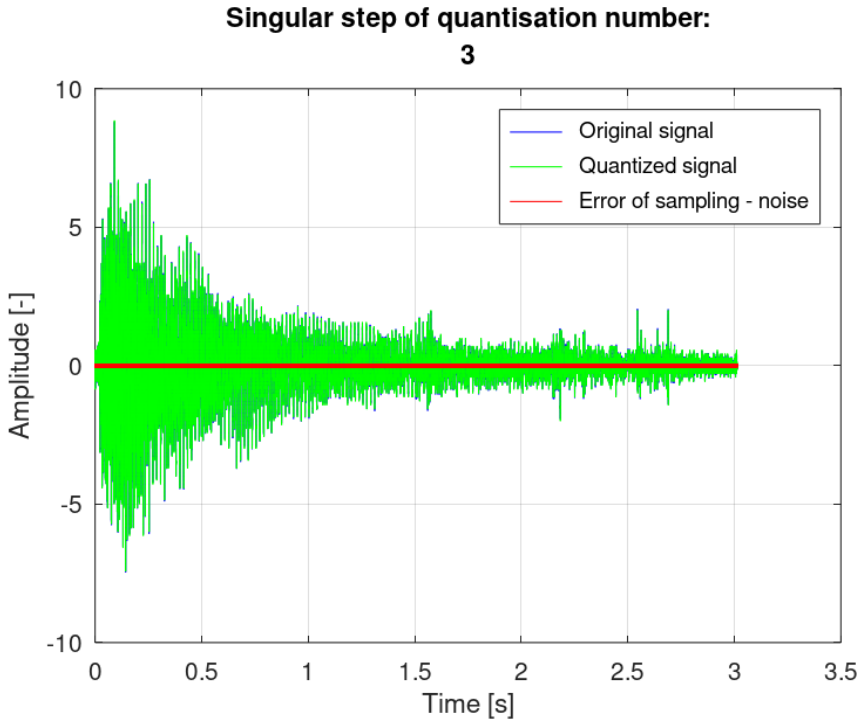
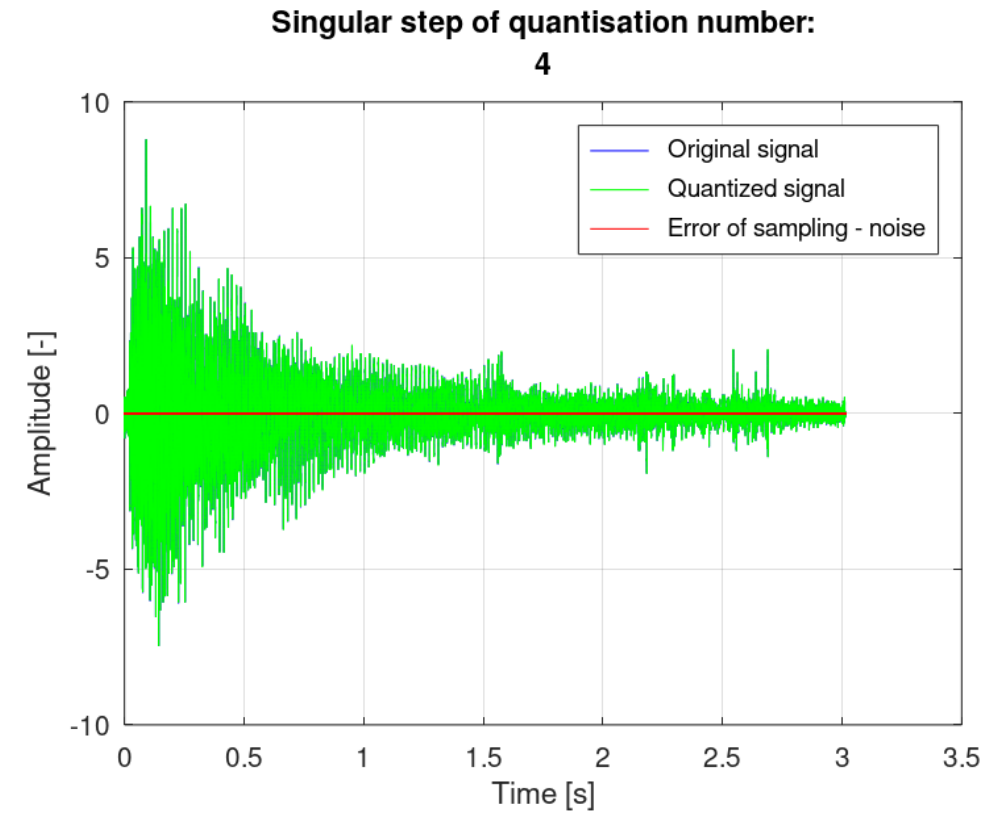
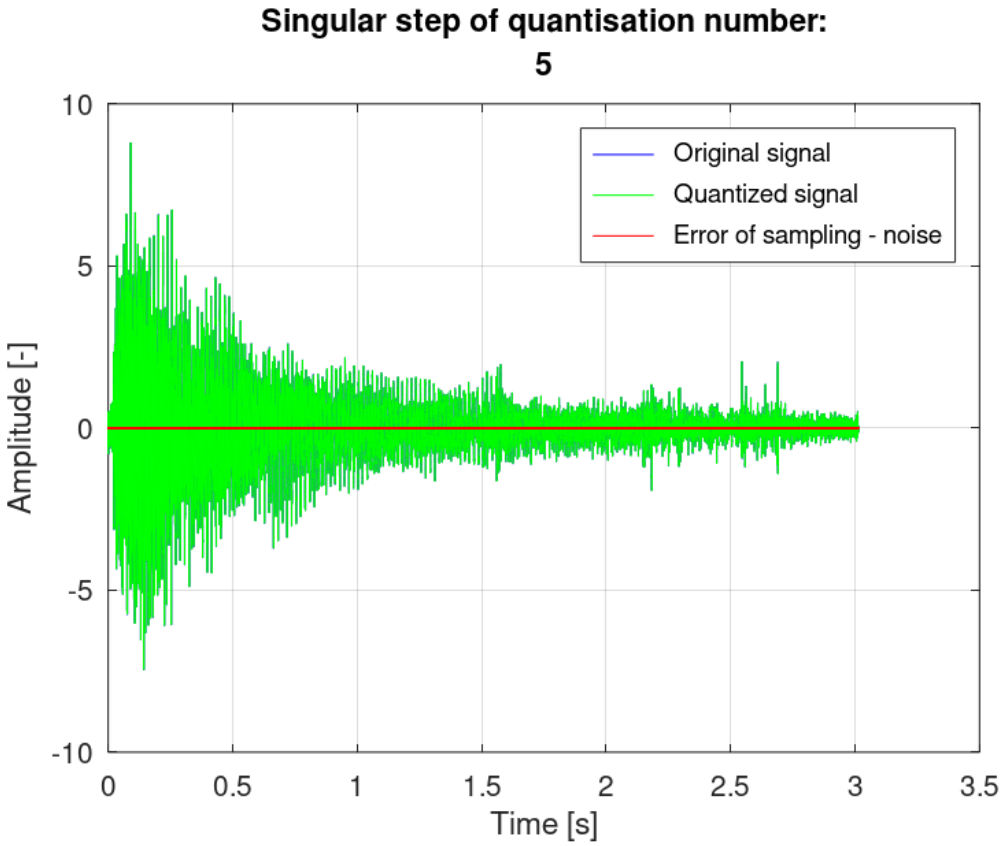
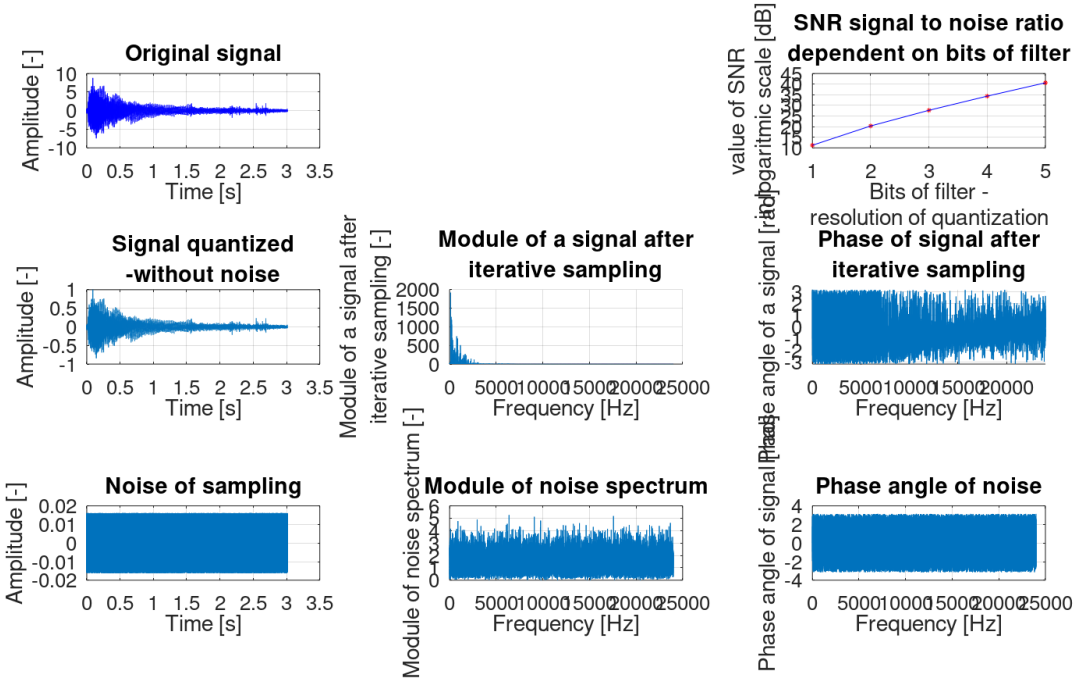


Figure 4.3. Third iteration of Quantization process

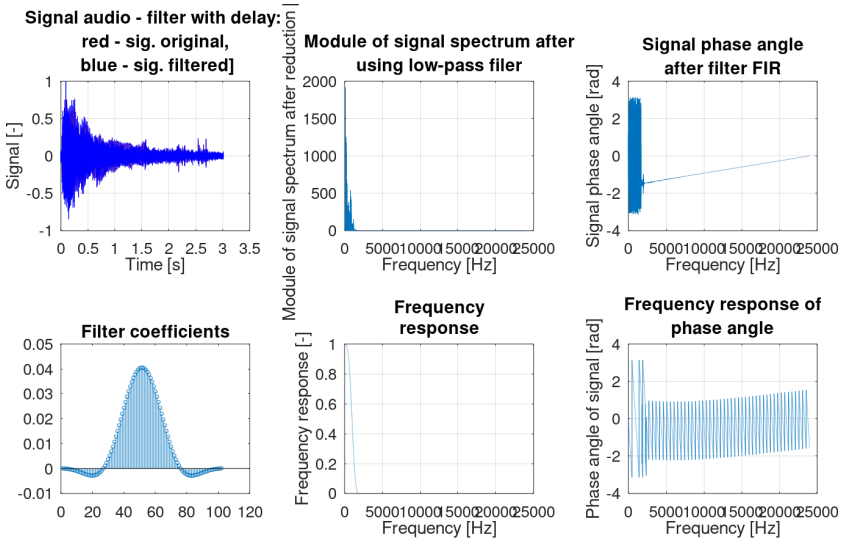

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5


Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

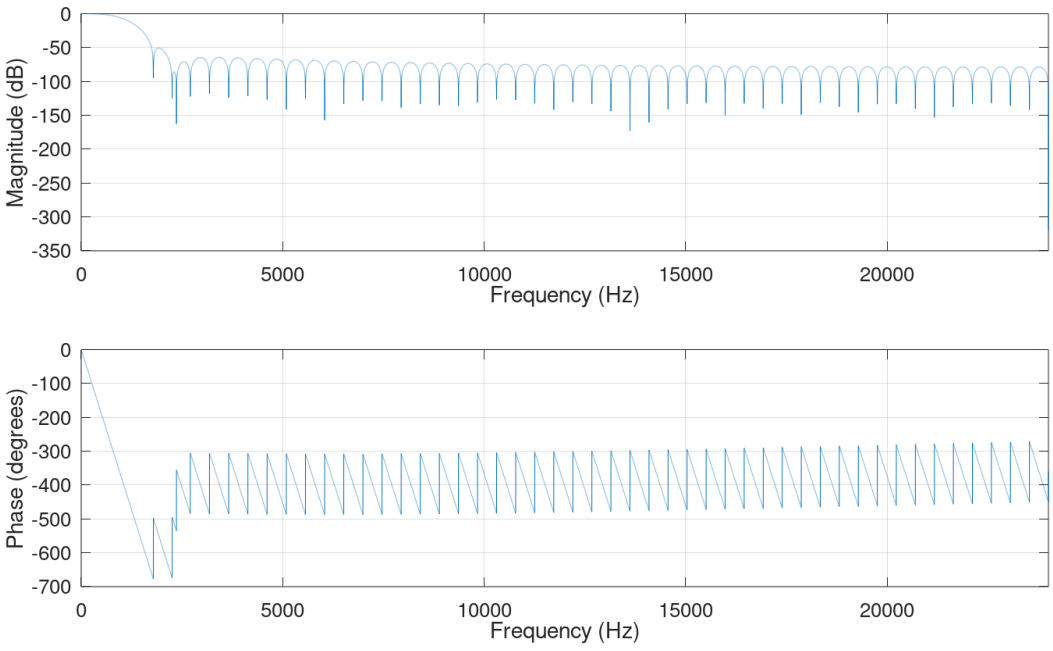
It is observable that through the iterative process of quantization, the error of sampling (noise) is depleting until it reaches the acceptable range which is justified by the signal to noise ratio.



This shows the first set of plots labeled as Figure 1 in the code. It is observable that the original signal and the quantized signal are almost identical, therefore the quantization process is a success. Following through, the SNR value increases every iteration of the process, achieving 40 db which is a good sign that the noise is greatly reduced. Some plots of the signals are also shown such as the noise of sampling, module of the signal after iterative sampling, module of the noise spectrum, the phase angle of signal, and phase angle of noise.

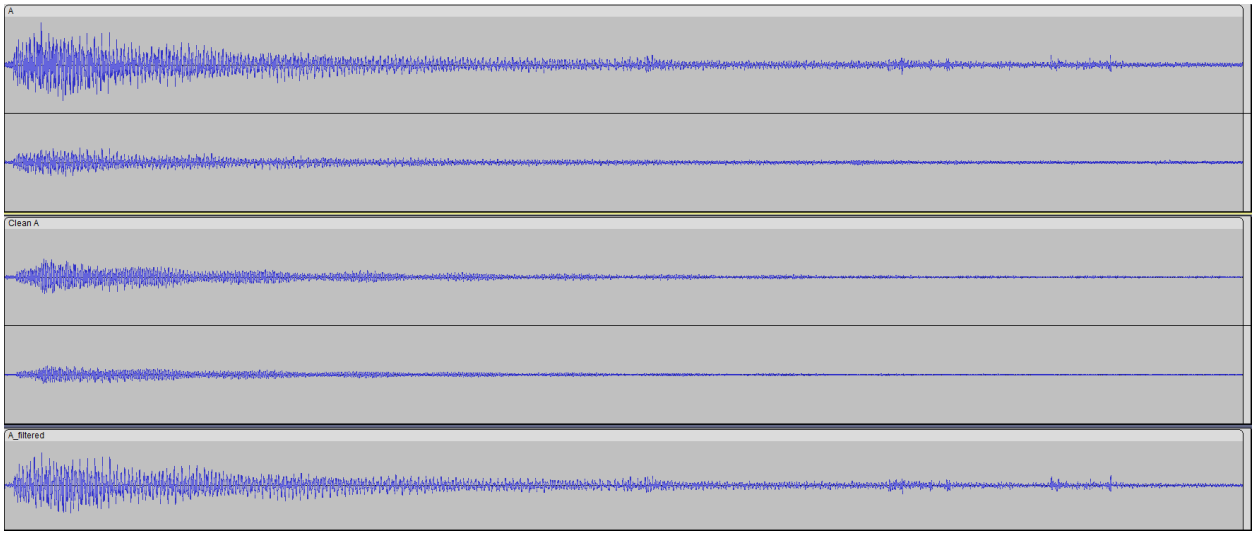


This shows the comparison of the signal audio with respect to delay through overlapping graphs. Other plots are also shown such as the filter coefficients, module of signal spectrum after filtration, frequency response of the signal, phase angle of the signal after filtration, and frequency response of the phase angle of the signal.



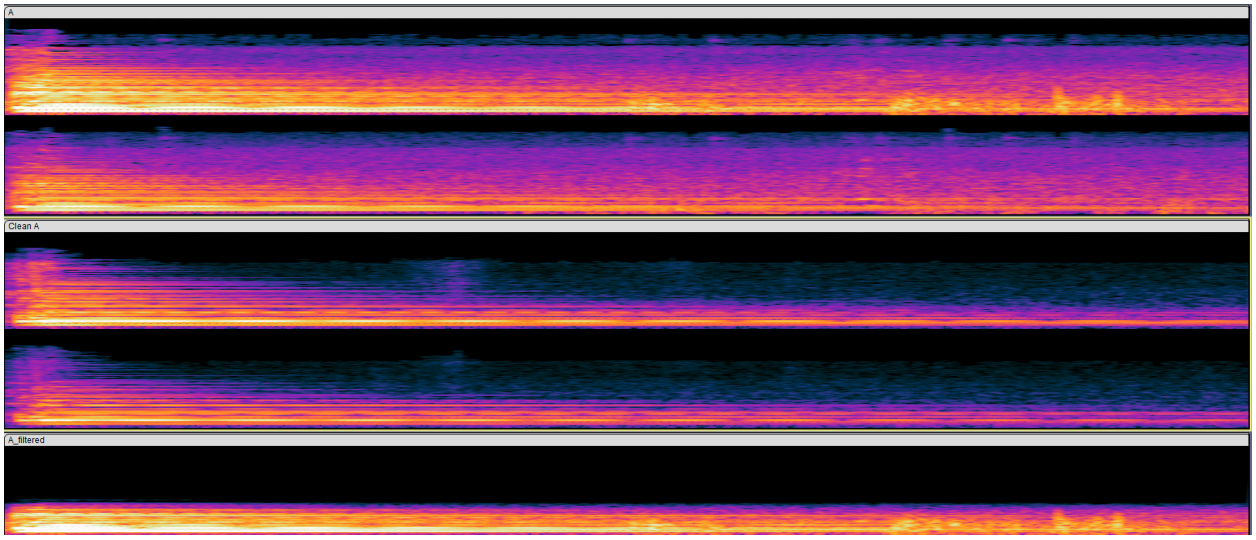
This shows the last figure of plots, which represents the frequency response of magnitude and phase angle of the signal.

A Chord (Stereo) | Clean A Chord (Stereo) | A_filtered Chord (Mono) Waveforms



This shows the waveforms of the noisy A chord, clean A chord, and filtered A chord respectively. It is observable that the filtered A chord waveform and the noise A chord waveform have the same structure, but the difference in frequencies are noticeable. However, the clean A chord waveform has lower amplitude.

A Chord (Stereo) | Clean A Chord (Stereo) | A_filtered Chord (Mono) Spectrograms



This shows the spectrogram of the noisy A chord, clean A chord, and filtered A chord respectively. It is noticeable that the frequency is cut in 1000 hertz from the filtered A chord, therefore eliminating a significant amount of background noise when compared to the higher frequencies of noisy and clean A chord. This however also eliminated higher frequencies completely, damaging the integrity of the original audio signal.

The overall results of the program gives us a filtered audio file that resembles the clean version of the chord recording. However, the researchers used a low-pass filter that cut off frequencies at above 1kHz. This means that guitar sounds at the highest frequencies are also cut off thus impacting the overall sound of the recording and specifically, the guitar. Furthermore, in some parts of the recording, there were sounds that passed through and were not filtered out due to the ambient noise not being at a high frequency.

CHAPTER V
Conclusion and Recommendation

The group used Octave as a digital signal processing tool to reduce high frequency background noise from the recorded samples of basic guitar chords. With the concepts of sampling and quantization, Fast Fourier Transform, finite impulse response, and low-pass filter, the objective was executed and accomplished with plots and the filtered audio file as proof. The frequency response of the signal resembles the usual frequency response when a low-pass filter is used and the difference is also observable from the given waveforms, spectrograms, and audio files. The filtered audio however, was not filtered finely as some of the guitar chord audio with high frequency was also filtered along with the background noise therefore reducing the audio quality of the overall filtered guitar chord audio. The group aims to learn more about the course to better apply the concepts in real-life situations, such as in audio file noise filtration.

The group recommends further analysis and utilization of other audio signal noise filters to achieve different results and identify more efficient algorithms. The group also recommends the use of better and more efficient filtering algorithms such as the band-pass filter to retain the quality of the audio being filtered and specifically filter the background noise in the recorded audio. Furthermore, the group recommends the recording of the noise as a separate recording in order to better compare the effectiveness of the algorithm that the future researchers will develop. MATLAB is also recommended as a digital signal processing tool since it has a built-in digital filter tool that allows users to create their own filter quickly based on their desired parameters and specifications.

BIBLIOGRAPHY

Encyclopedia Britannica, inc. (n.d.). *Wave*. Encyclopedia Britannica. Retrieved January 30, 2022, from https://www.britannica.com/science/wave-physics

Fink, D. (2019). A new definition of noise: Noise is unwanted and/or harmful sound. noise is the new 'secondhand smoke'. *Proceedings of Meetings on Acoustics*. https://doi.org/10.1121/2.0001186

Purves D, Augustine GJ, Fitzpatrick D, et al., editors. Neuroscience. 2nd edition. Sunderland (MA): Sinauer Associates; 2001. The Audible Spectrum. Available from: https://www.ncbi.nlm.nih.gov/books/NBK10924/

Zölzer Udo. (2008). Introduction. In *Digital Audio Signal Processing* (2nd ed.). Chapter, Wiley.

Salih, A.O.M. (2017) Audio Noise Reduction Using Low Pass Filters. Open Access Library Journal, 4: e3709. https://doi.org/10.4236/oalib.1103709

Karki, J. (2000). Active low-pass filter design. *Texas Instruments application report*.

R. Kumar, R. Kumar, D. Samanta, M. Paul and V. Kumar, "A combining approach using DFT and FIR filter to enhance impulse response," 2017 International Conference on Computing Methodologies and Communication (ICCMC), 2017, pp. 134-137, doi: 10.1109/ICCMC.2017.8282660.

G. D. Bergland, "A guided tour of the fast Fourier transform," in IEEE Spectrum, vol. 6, no. 7, pp. 41-52, July 1969, doi: 10.1109/MSPEC.1969.5213896.

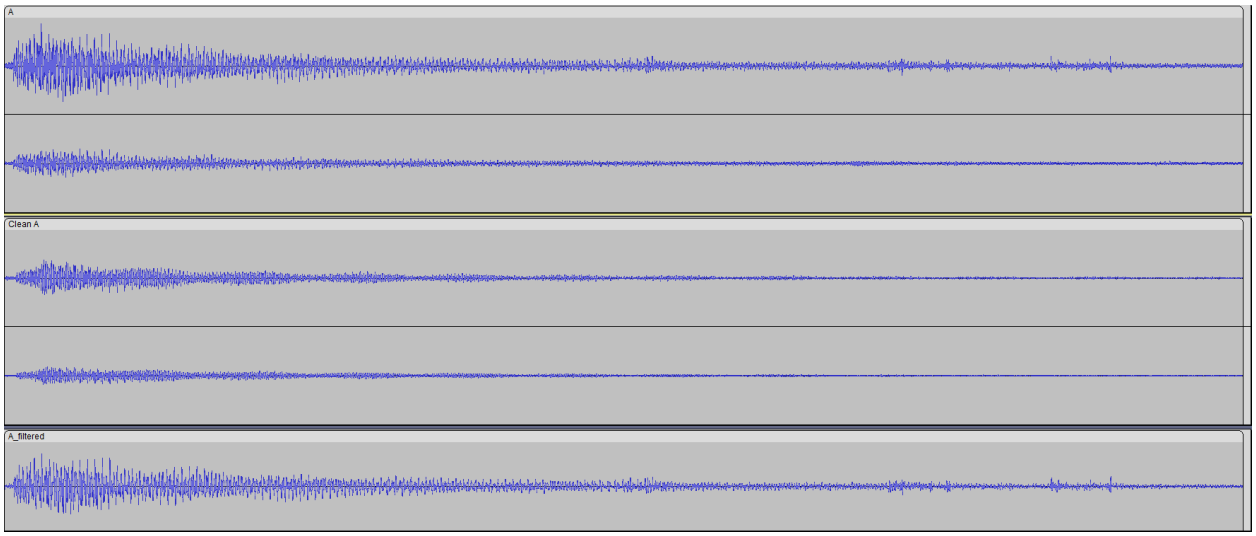
Brunton, S. L., & Kutz, J. N. (2017). *Data Driven Science & Engineering Machine Learning, Dynamical Systems, and Control* [E-book], pp. 65-68.

J. G. Liu and G. L. K. Morgan, "FFT Selective and Adaptive Filtering for Removal of Systematic Noise in ETM+ Imageodesy Images," in IEEE Transactions on Geoscience and Remote Sensing, vol. 44, no. 12, pp. 3716-3724, Dec. 2006, doi: 10.1109/TGRS.2006.881752.

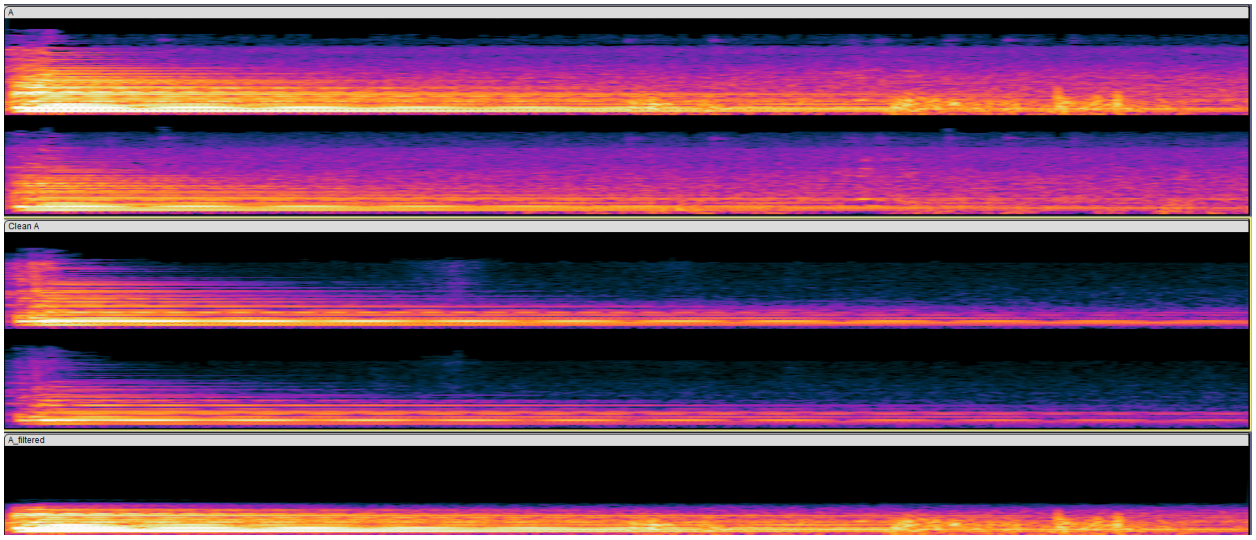
APPENDICES

Appendix A - Audio Waveforms and Spectrograms

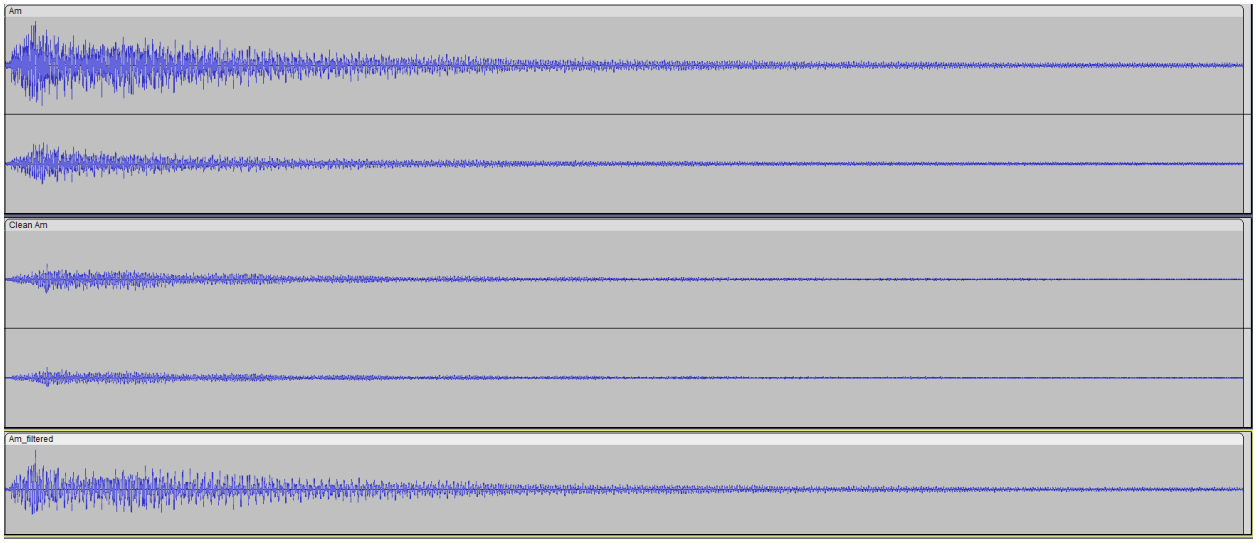
A Chord (Stereo) | Clean A Chord (Stereo) | A_filtered Chord (Mono) Waveforms



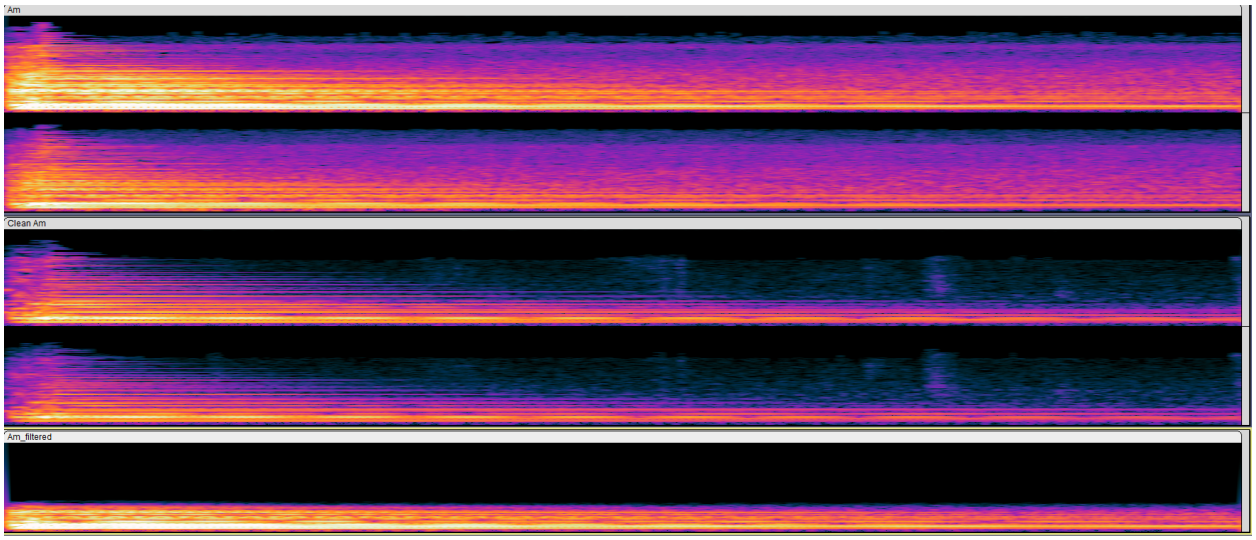
A Chord (Stereo) | Clean A Chord (Stereo) | A_filtered Chord (Mono) Spectrograms



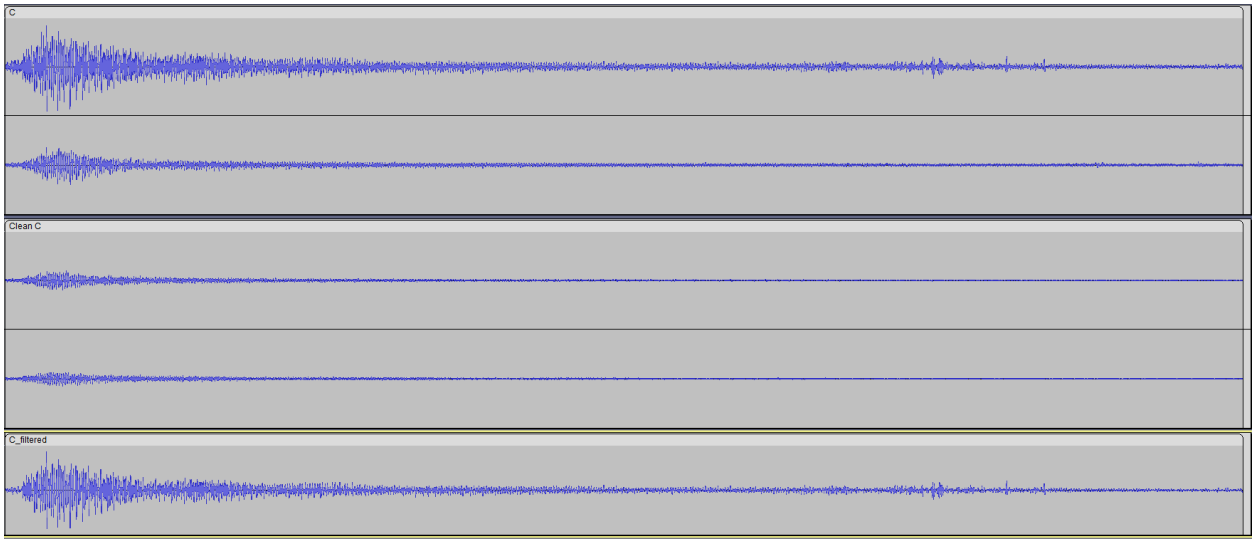
Am Chord (Stereo) | Clean Am Chord (Stereo) | Am_filtered Chord (Mono) Waveforms



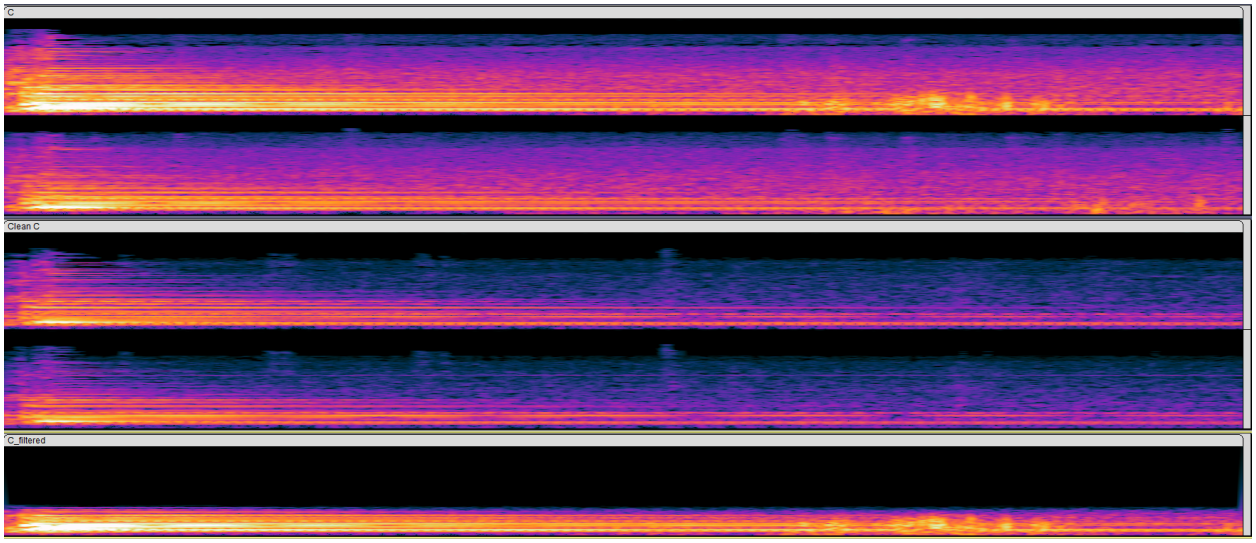
Am Chord (Stereo) | Clean Am Chord (Stereo) | Am_filtered Chord (Mono) Spectrograms



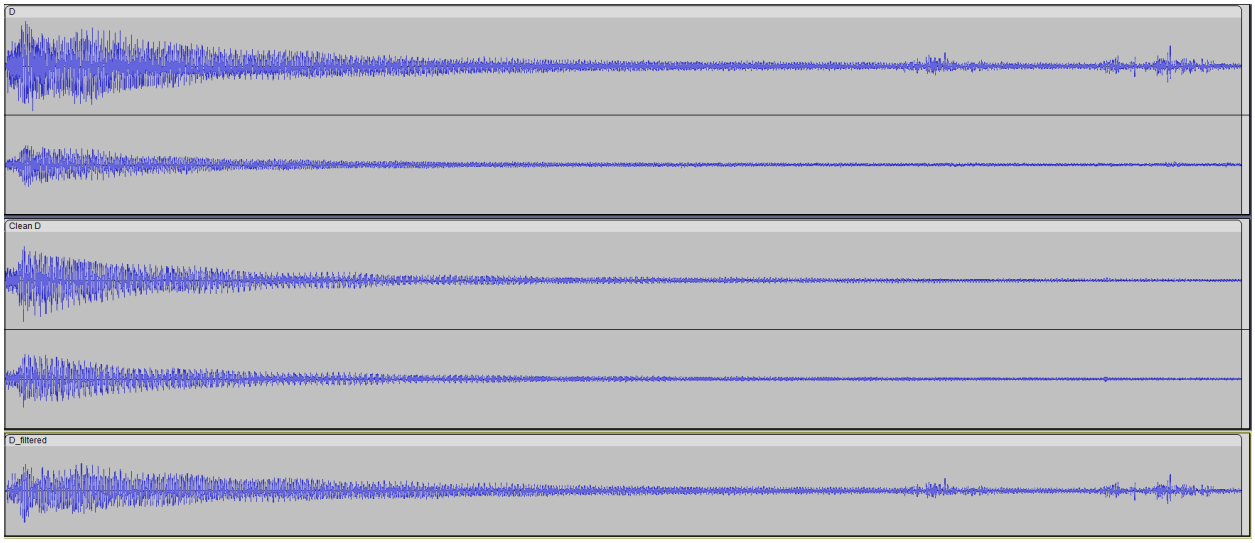
C Chord (Stereo) | Clean C Chord (Stereo) | C_filtered Chord (Mono) Waveforms



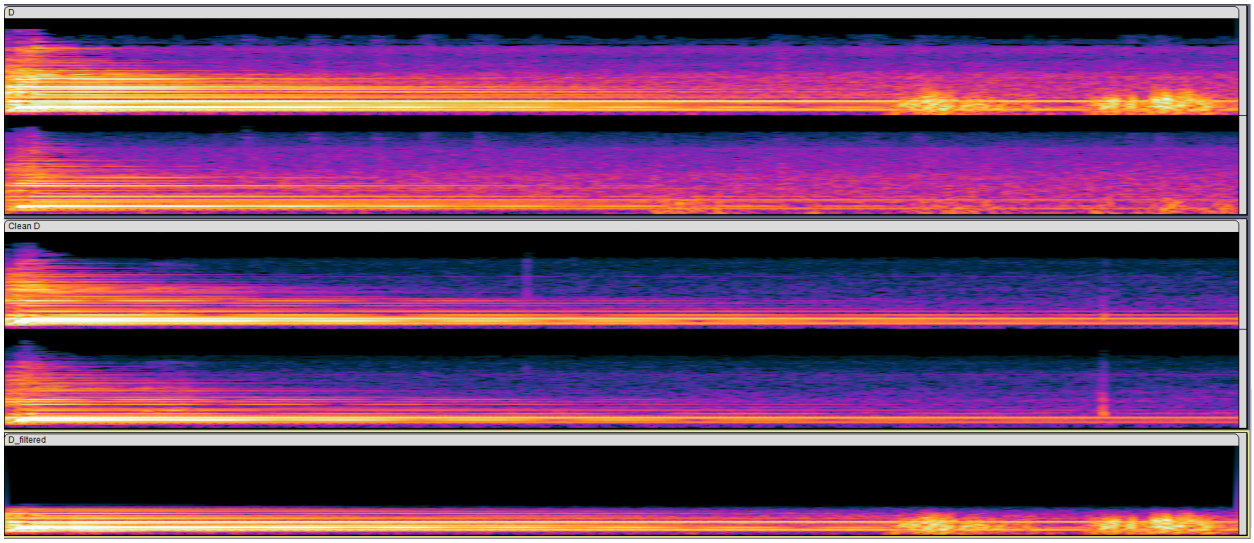
C Chord (Stereo) | Clean C Chord (Stereo) | C_filtered Chord (Mono) Spectrograms



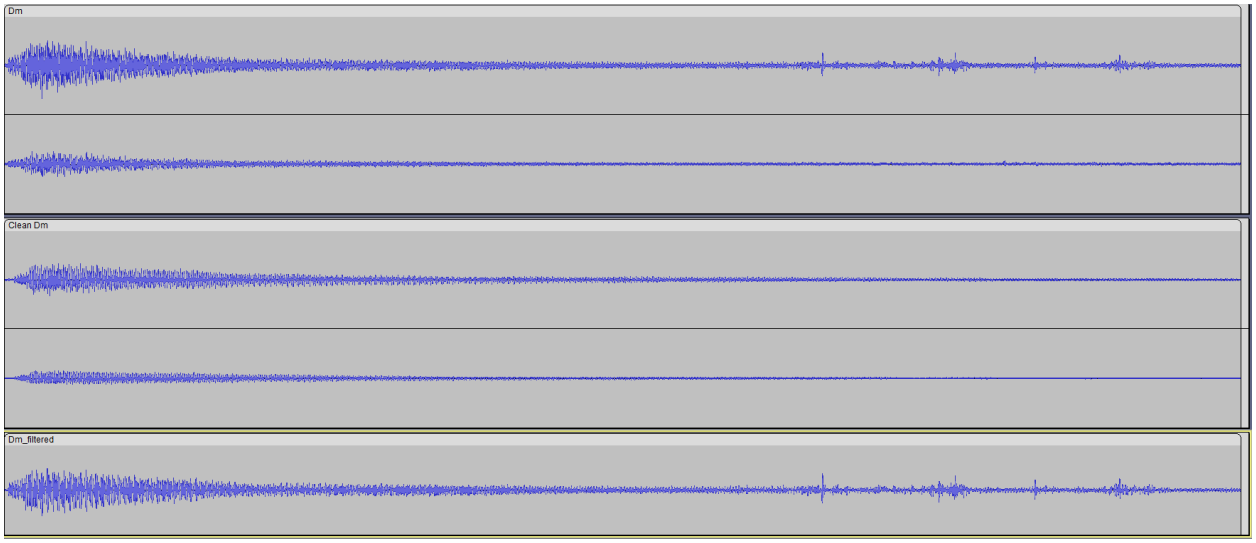
D Chord (Stereo) | Clean D Chord (Stereo) | D_filtered Chord (Mono) Waveforms



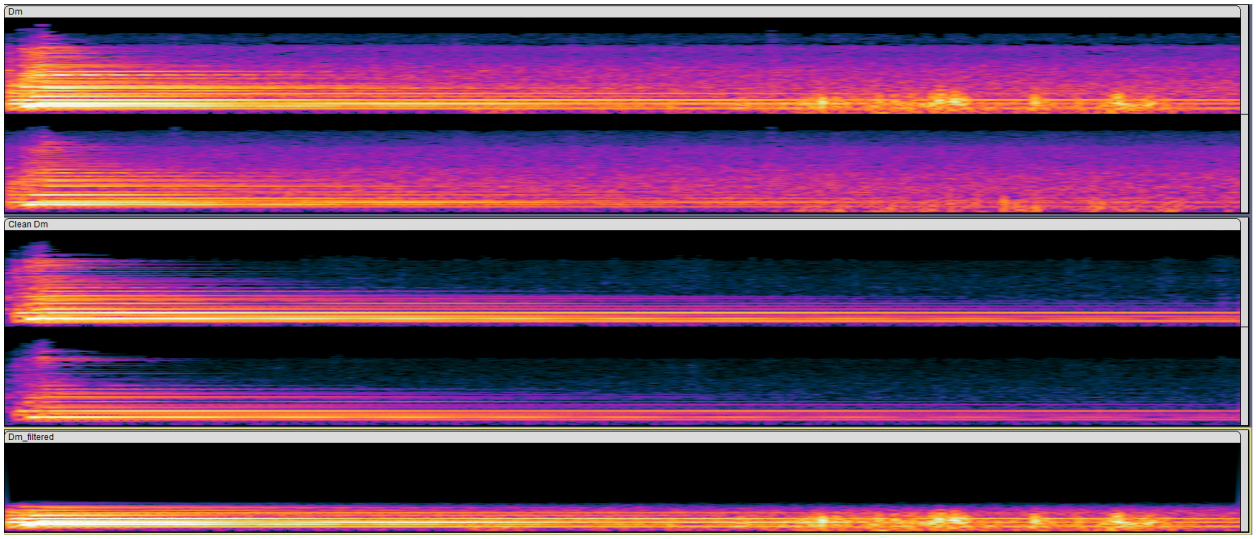
D Chord (Stereo) | Clean D Chord (Stereo) | D_filtered Chord (Mono) Spectrograms



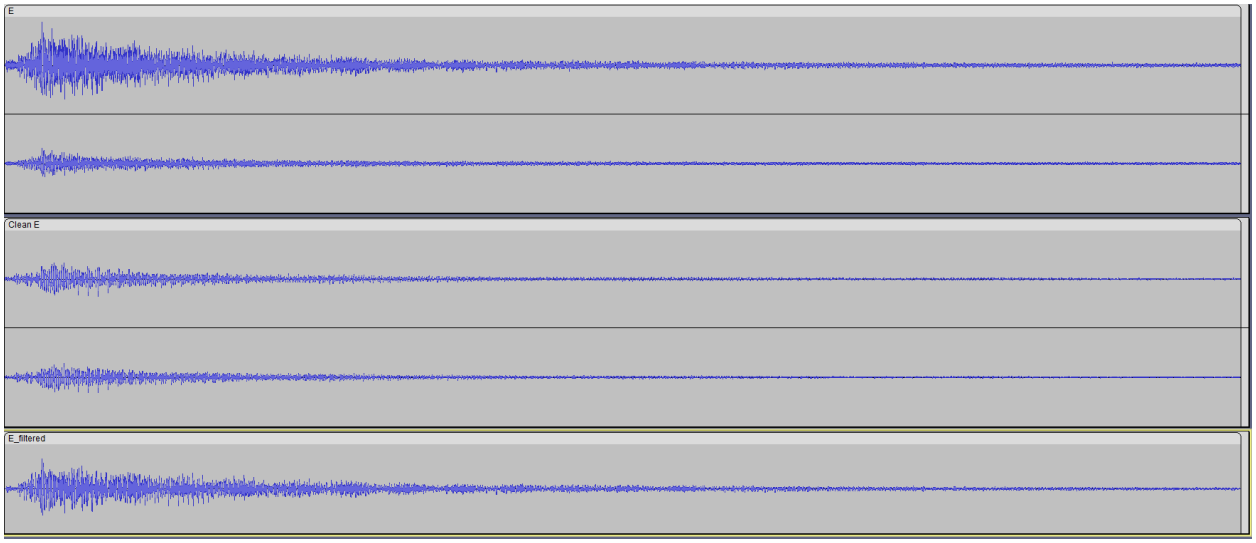
Dm Chord (Stereo) | Clean Dm Chord (Stereo) | Dm_filtered Chord (Mono) Waveforms



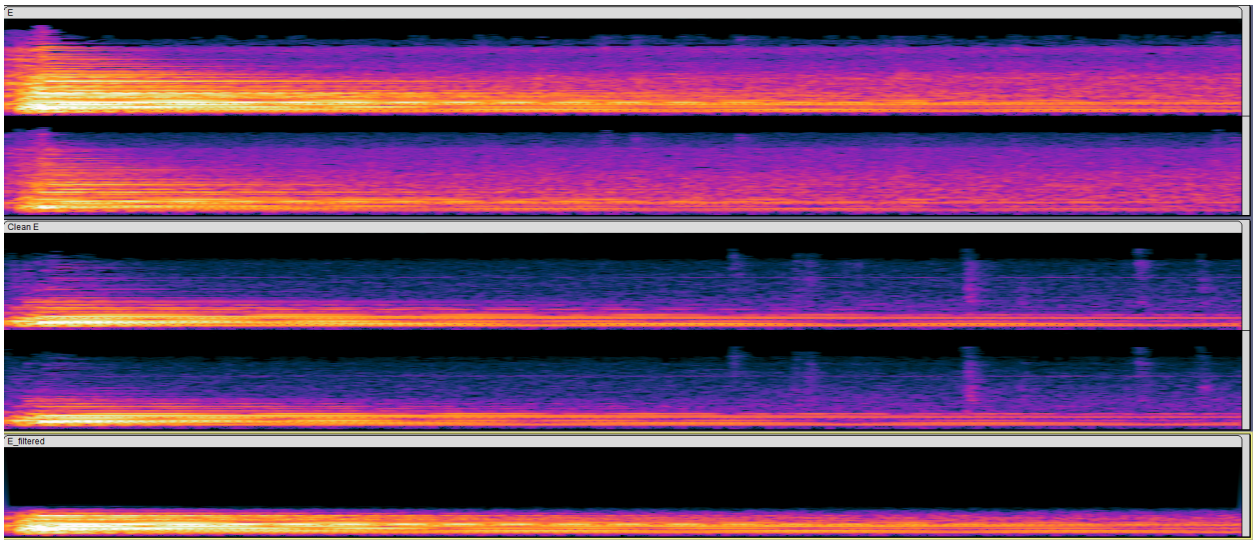
Dm Chord (Stereo) | Clean Dm Chord (Stereo) | Dm_filtered Chord (Mono) Spectrograms



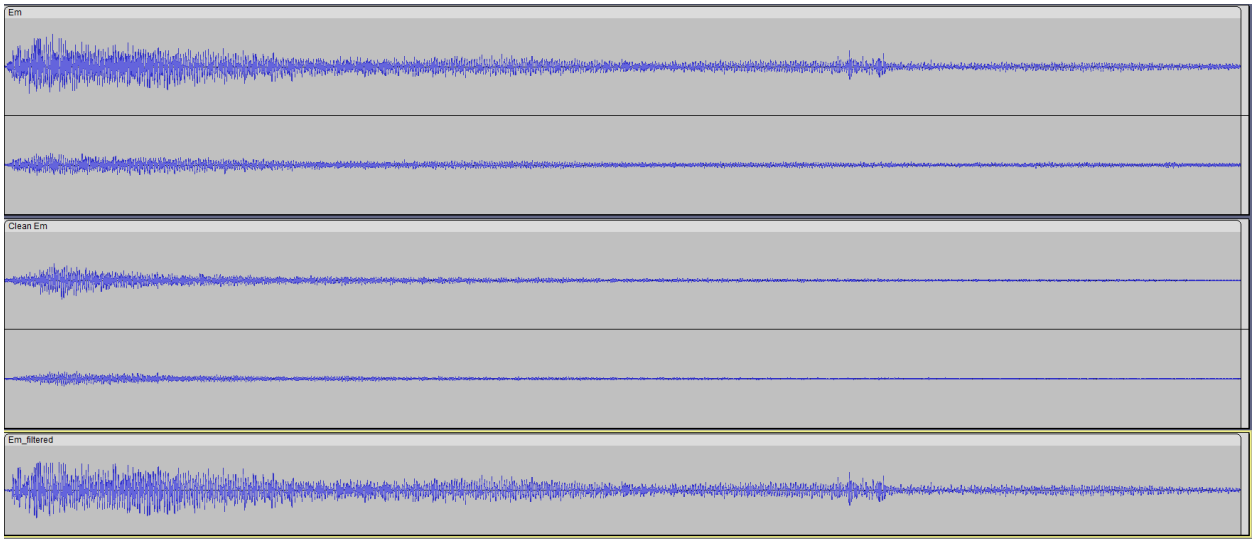
E Chord (Stereo) | Clean E Chord (Stereo) | E_filtered Chord (Mono) Waveforms



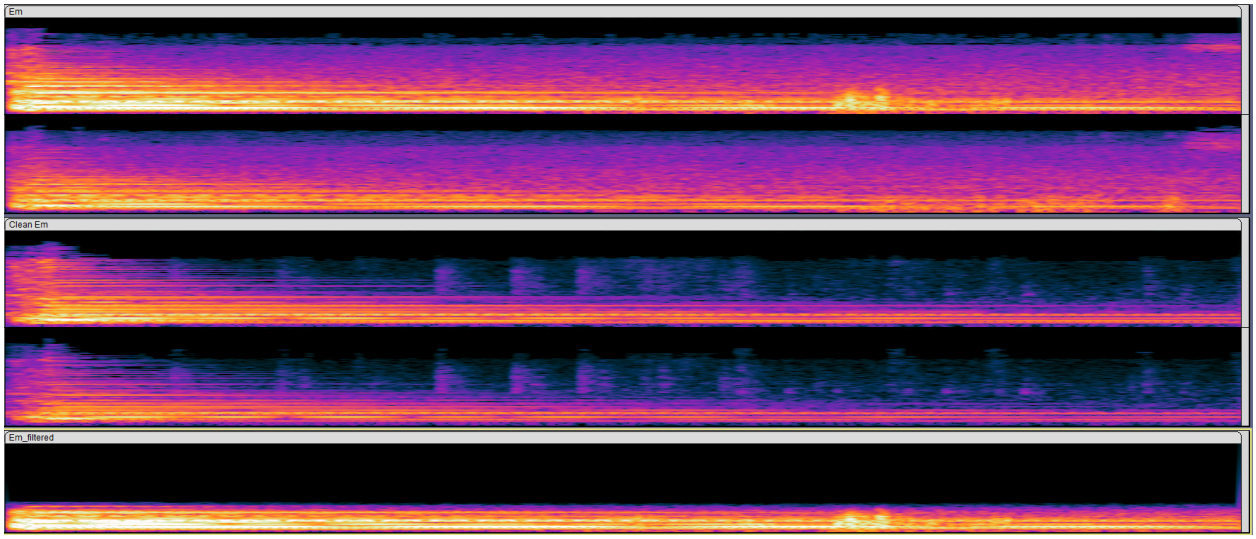
E Chord (Stereo) | Clean E Chord (Stereo) | E_filtered Chord (Mono) Spectrogram



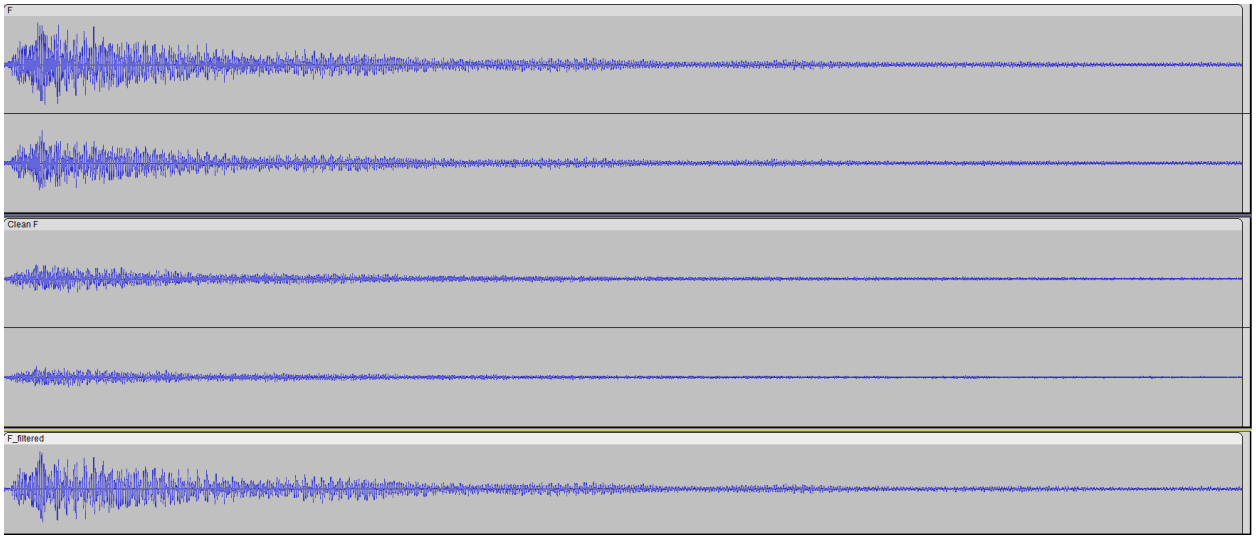
Em Chord (Stereo) | Clean Em Chord (Stereo) | Em_filtered Chord (Mono) Waveforms



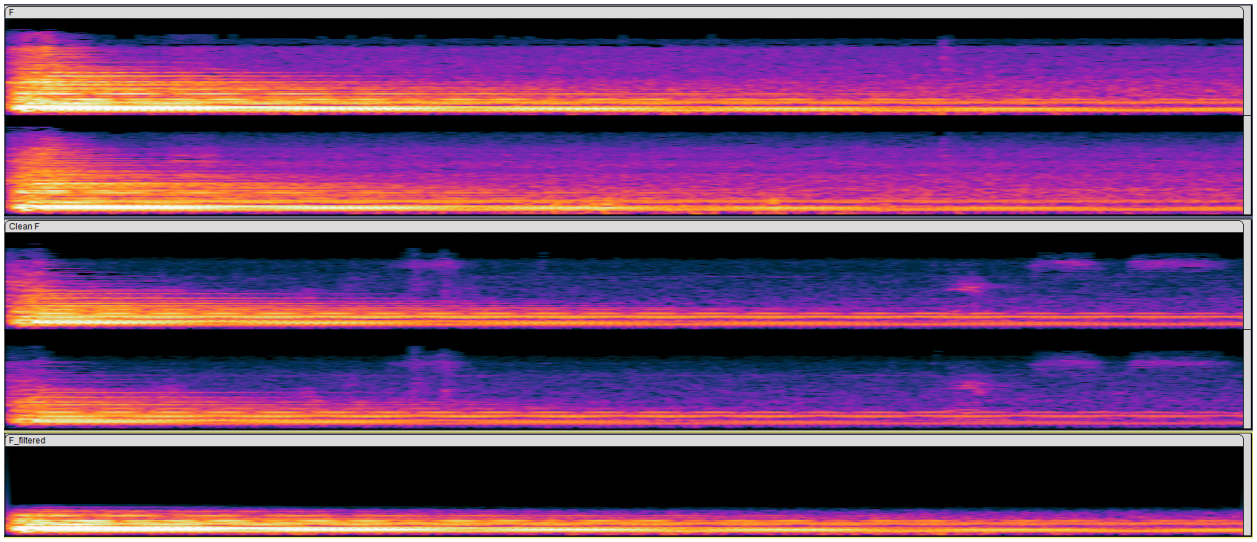
Em Chord (Stereo) | Clean Em Chord (Stereo) | Em_filtered Chord (Mono) Spectrogram



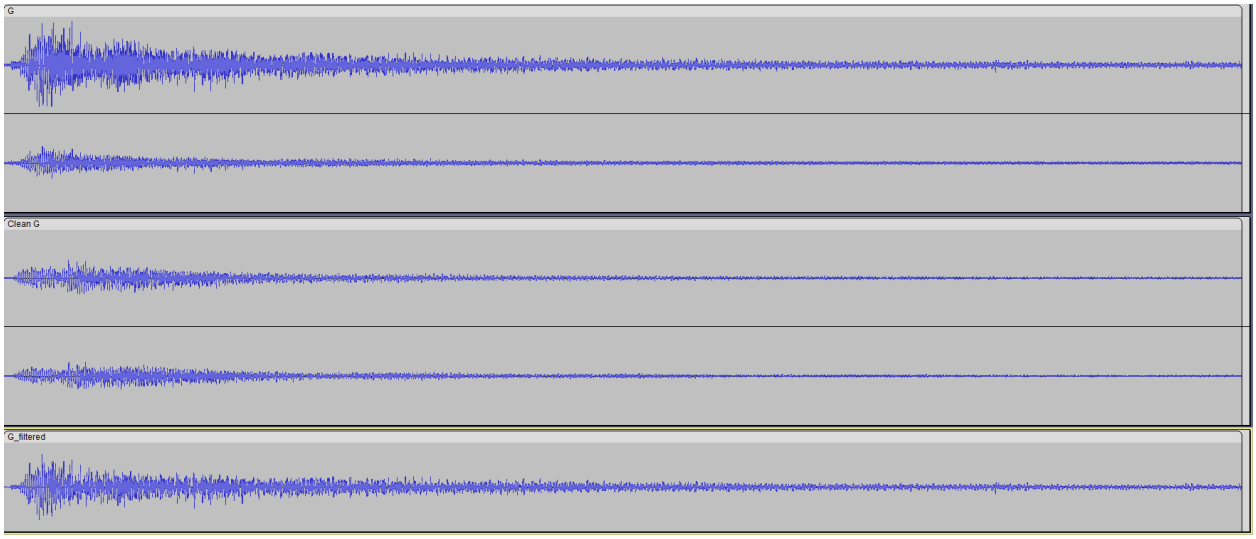
F Chord (Stereo) | Clean F Chord (Stereo) | F_filtered Chord (Mono) Waveforms



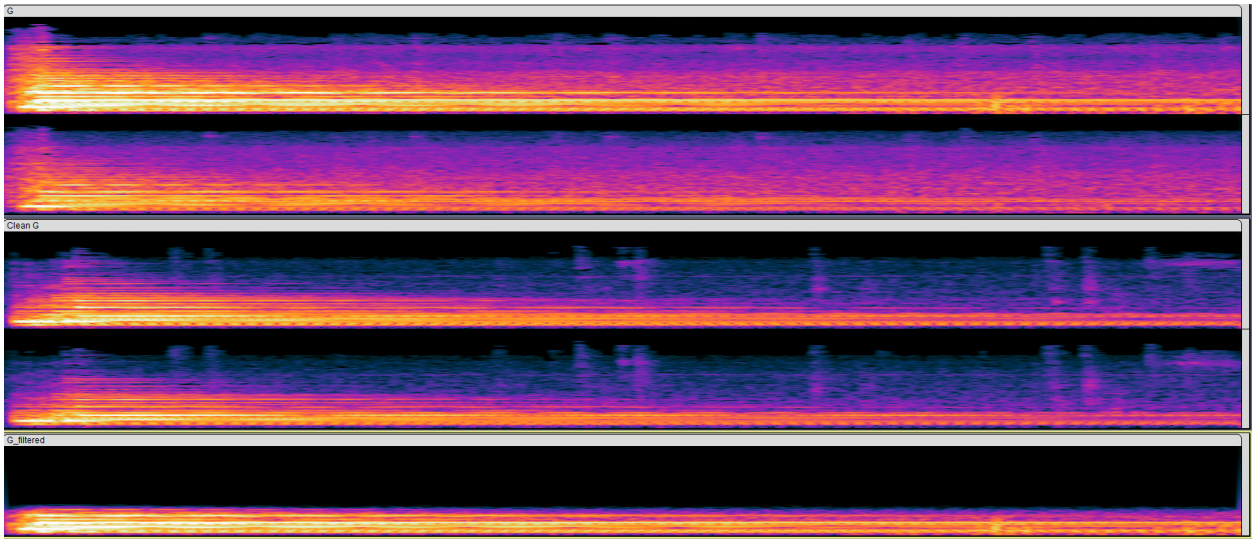
F Chord (Stereo) | Clean F Chord (Stereo) | F_filtered Chord (Mono) Spectrogram



G Chord (Stereo) | Clean G Chord (Stereo) | G_filtered Chord (Mono) Waveforms

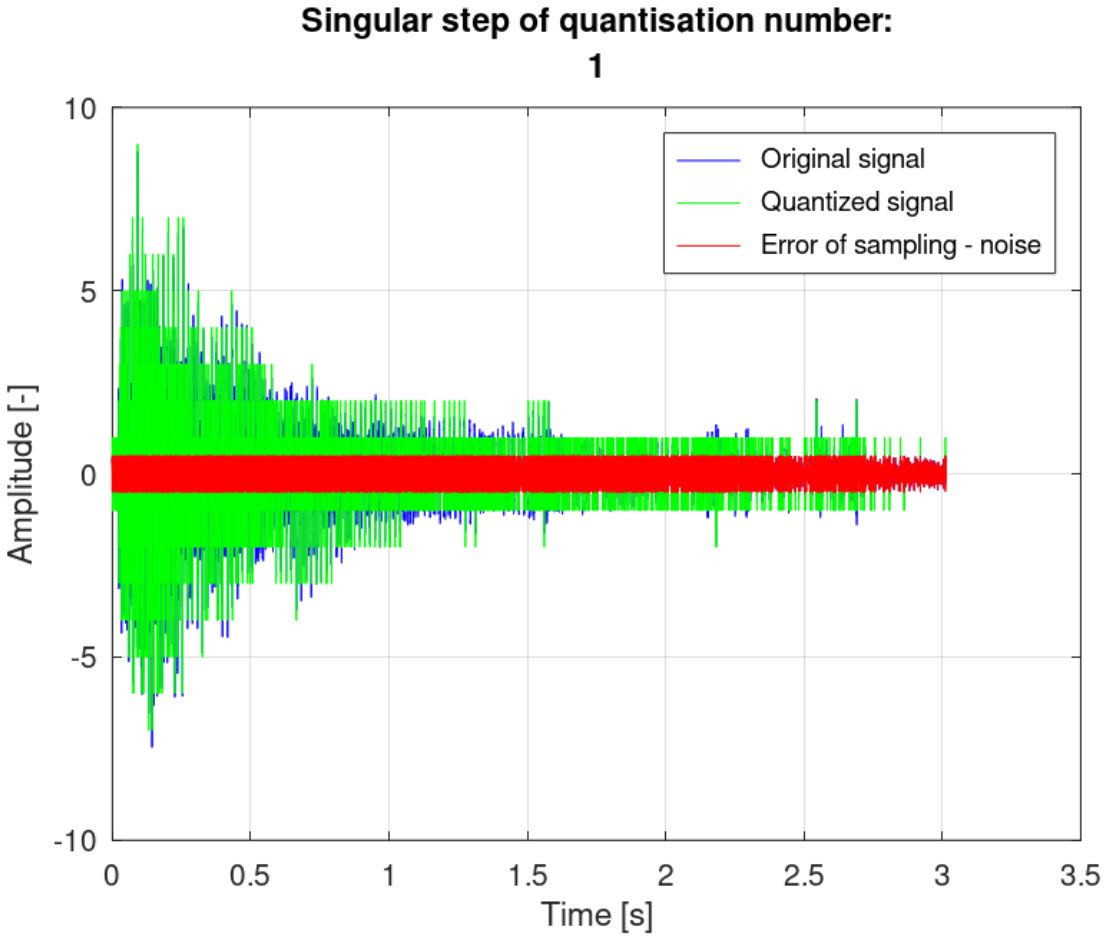


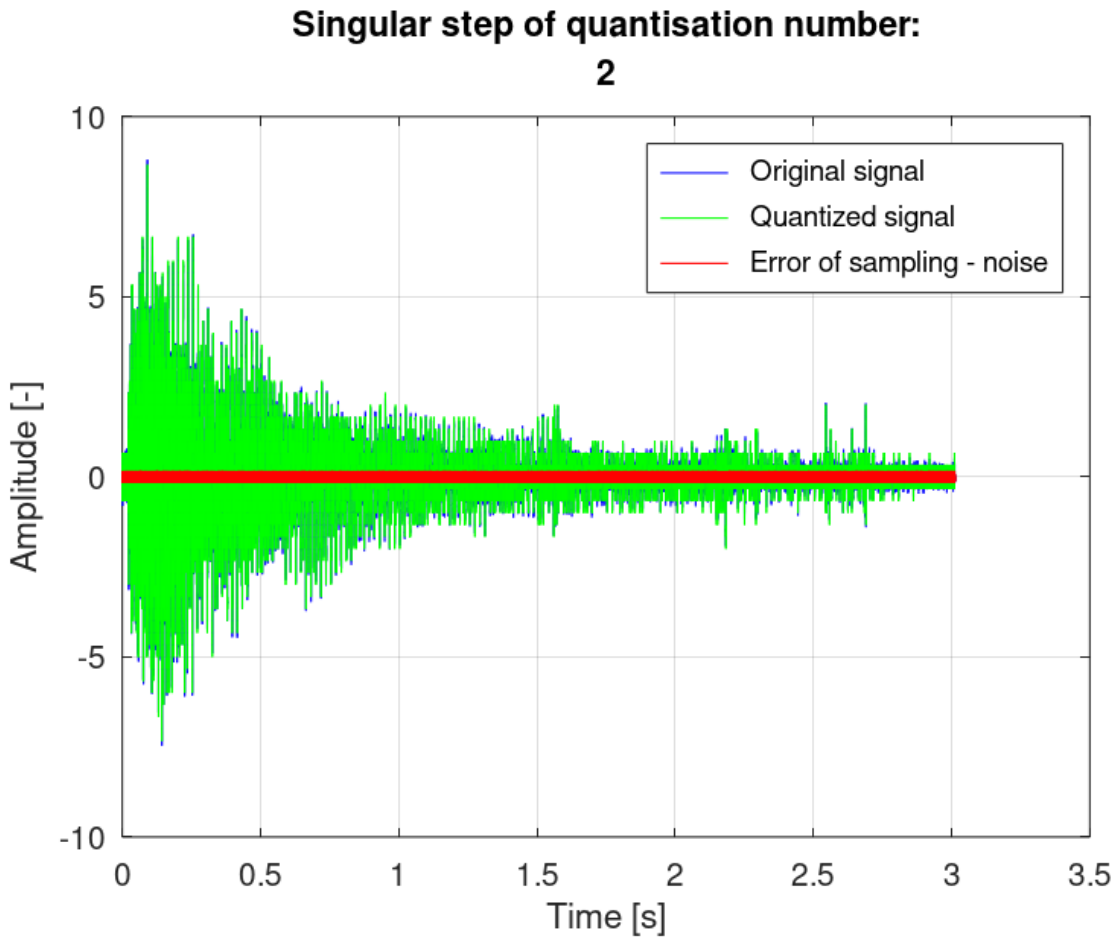
G Chord (Stereo) | Clean G Chord (Stereo) | G_filtered Chord (Mono) Spectrogram

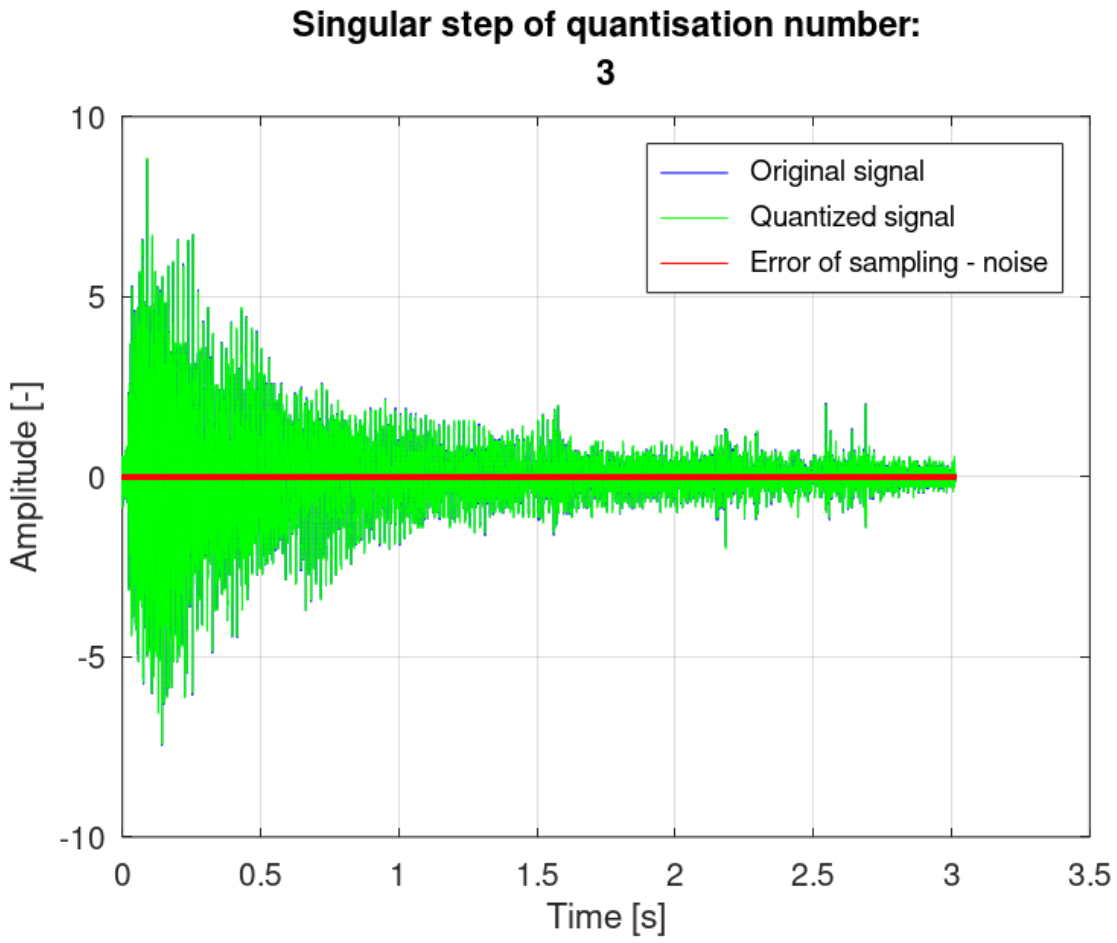


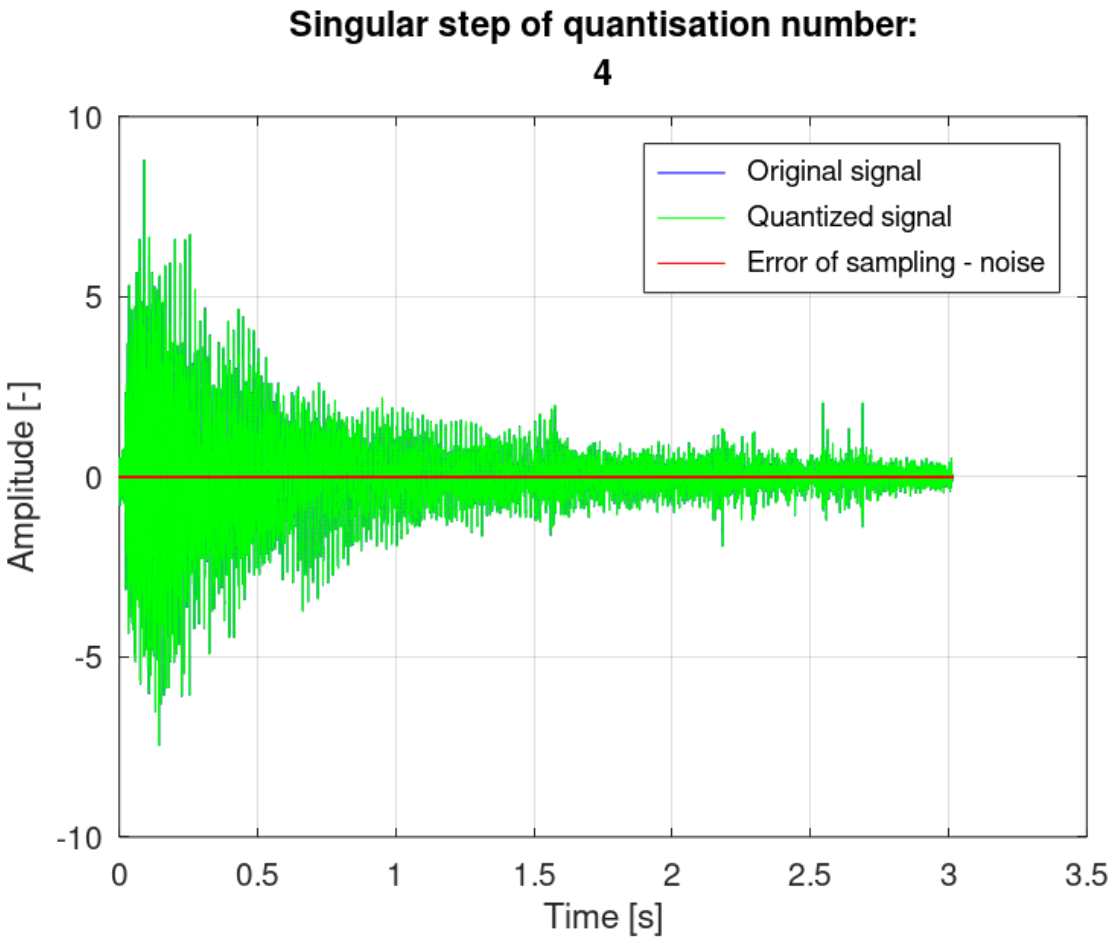
Appendix B - Plots and Figures

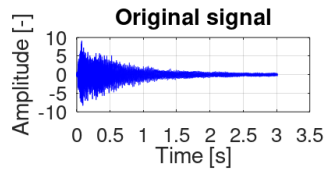
A Chord

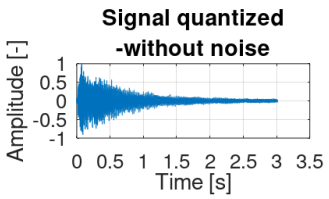


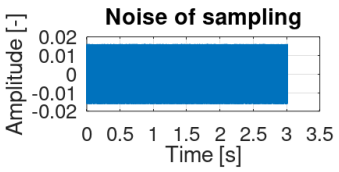

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

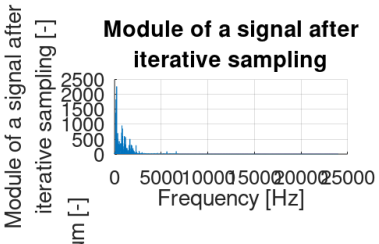

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

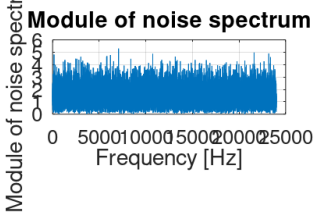

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

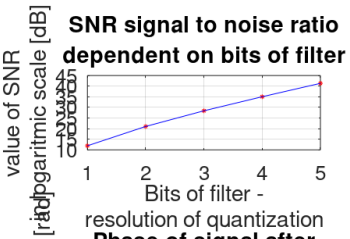

Original signal
Amplitude [-]
Time [s]

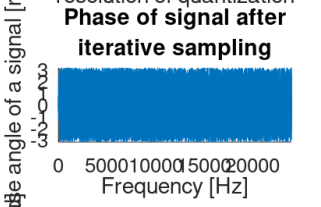

Signal quantized
-without noise
Amplitude [-]
Time [s]

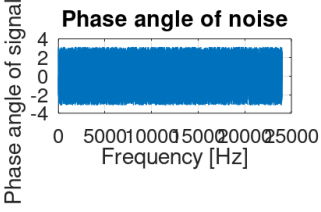

Noise of sampling
Amplitude [-]
Time [s]

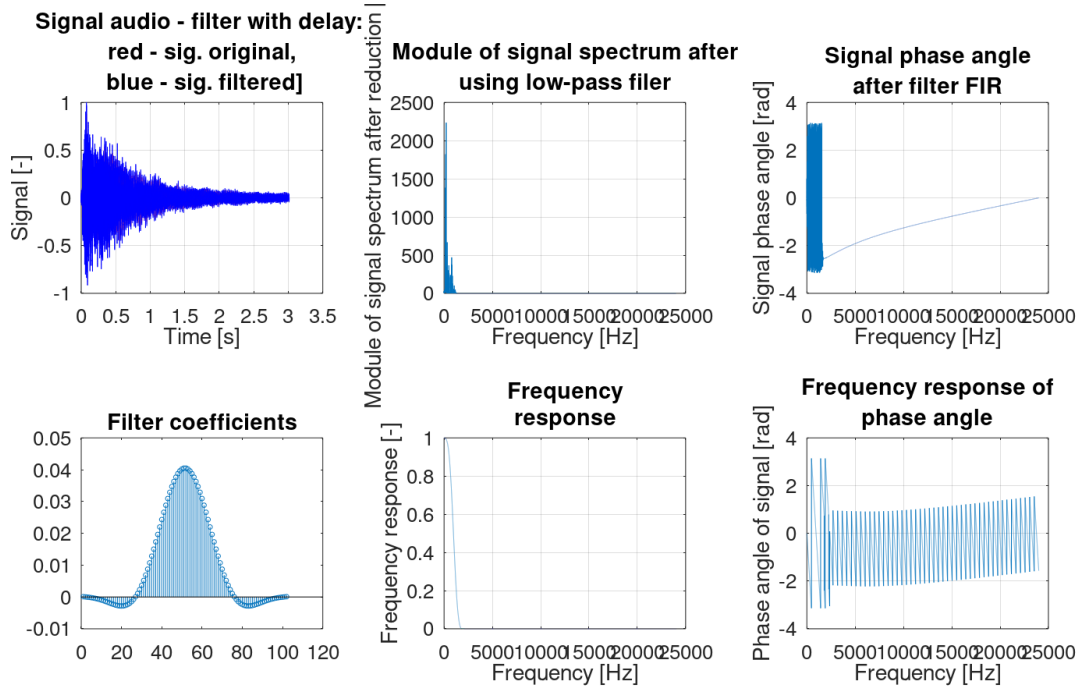

Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

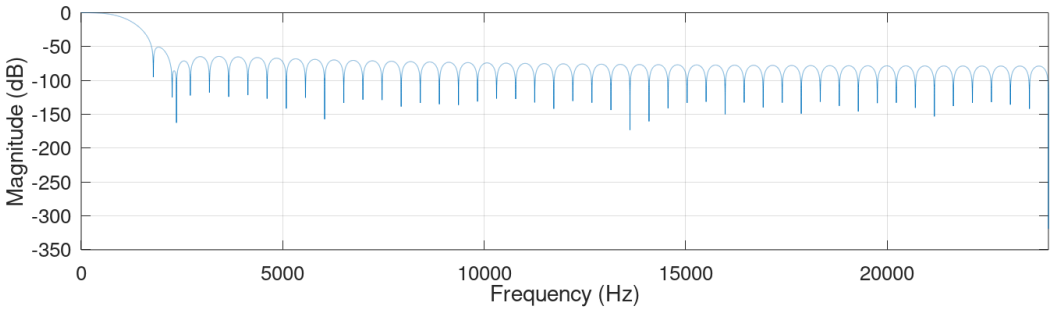

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

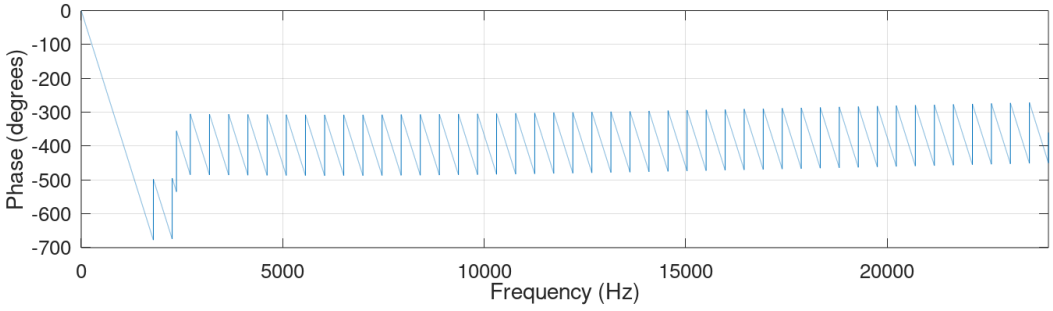

SNR signal to noise ratio
dependent on bits of filter
value of SNR
logaritmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]

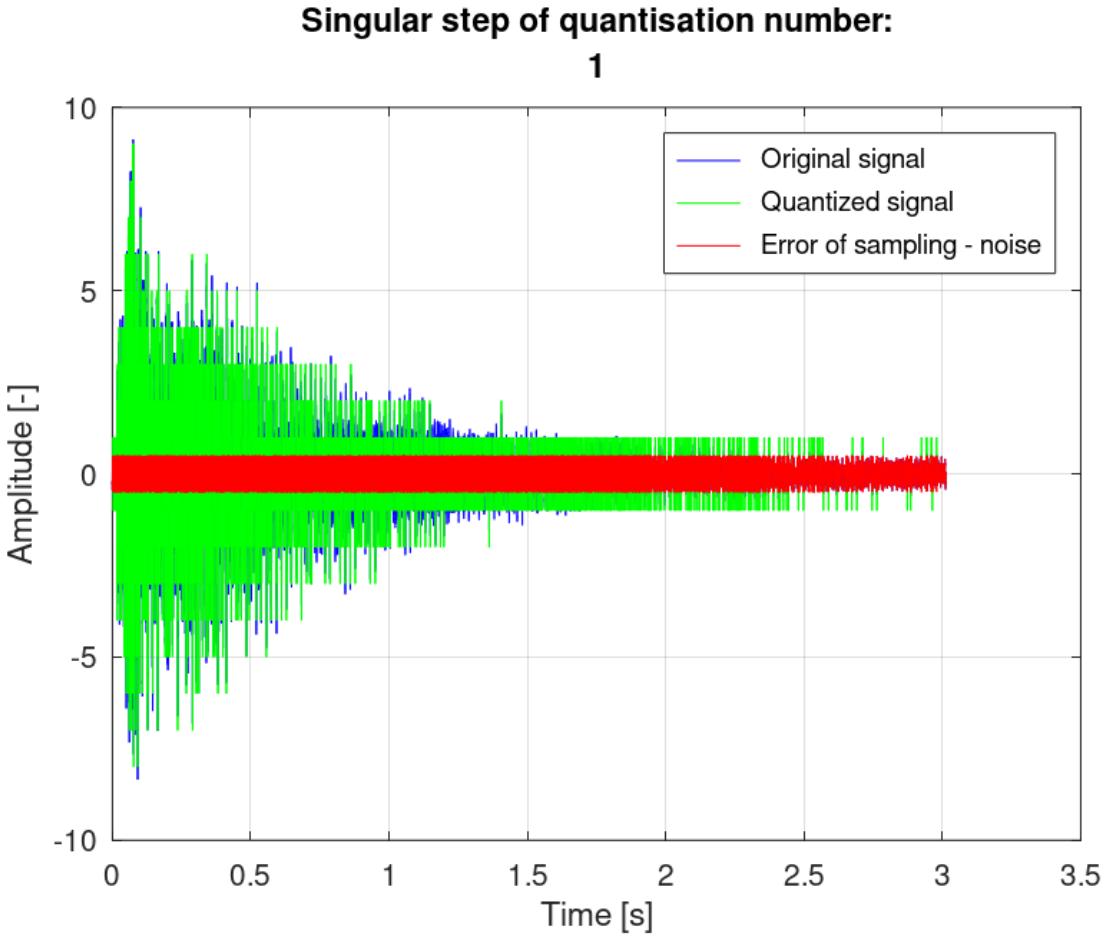

Phase angle of noise
Frequency [Hz]

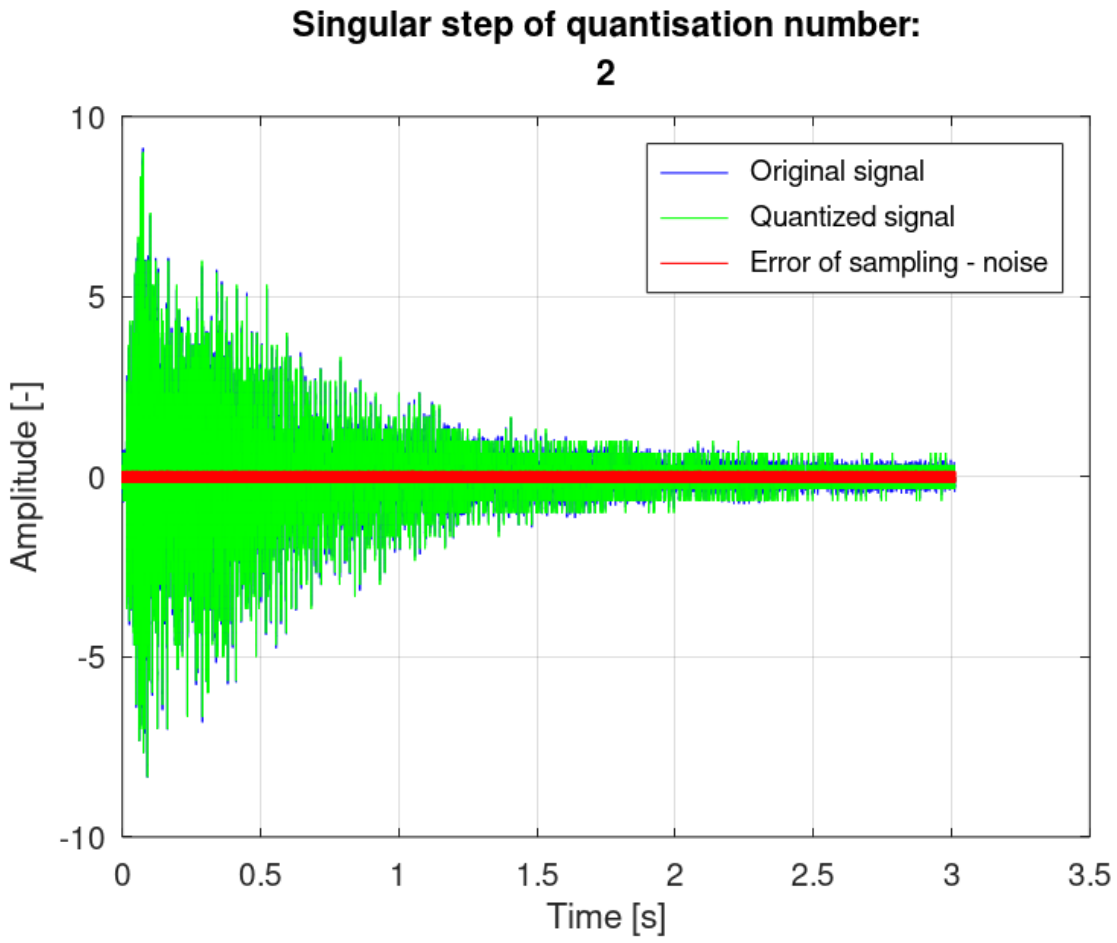

Signal audio - filter with delay: red - sig. original, blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after using low-pass filer
Module of signal spectrum after reduction
Frequency [Hz]
Signal phase angle after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency response
Frequency response [-]
Frequency [Hz]
Frequency response of phase angle
Phase angle of signal [rad]
Frequency [Hz]

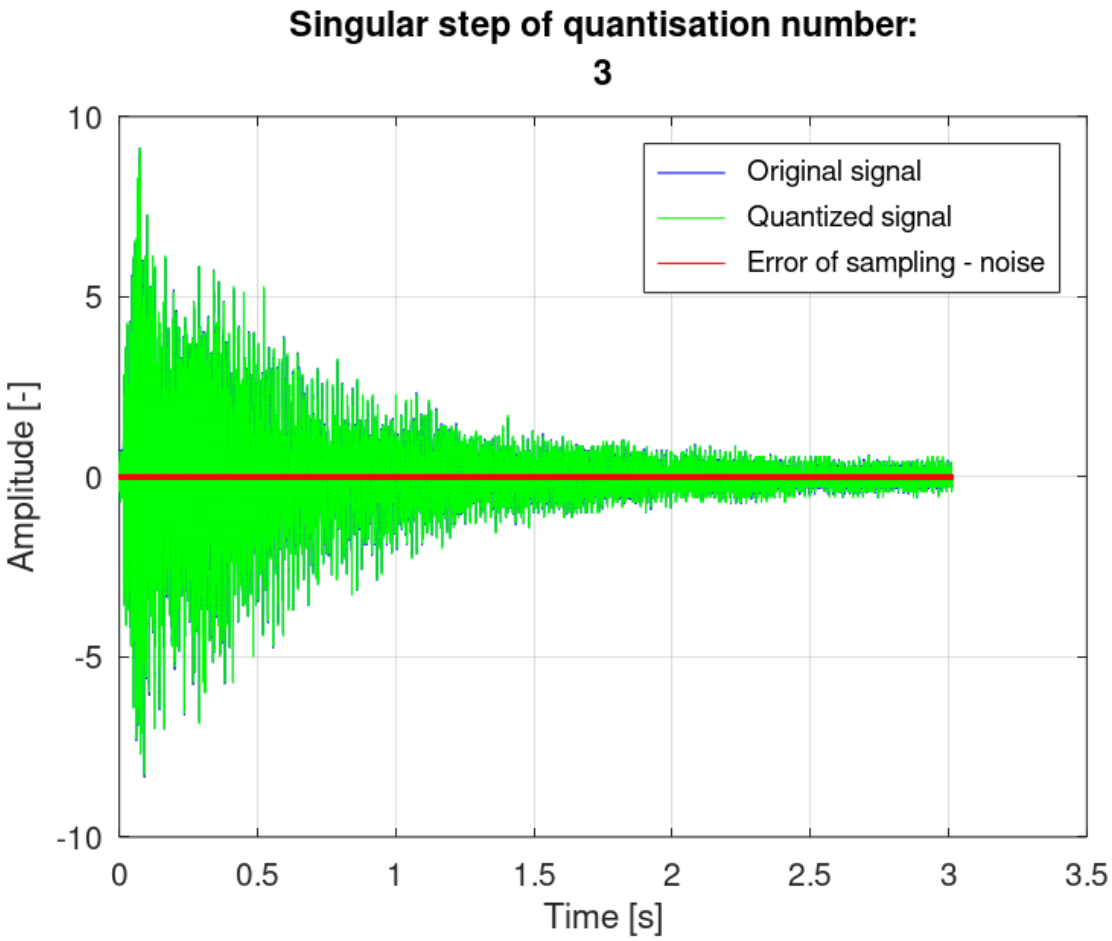

Magnitude (dB)
Frequency (Hz)

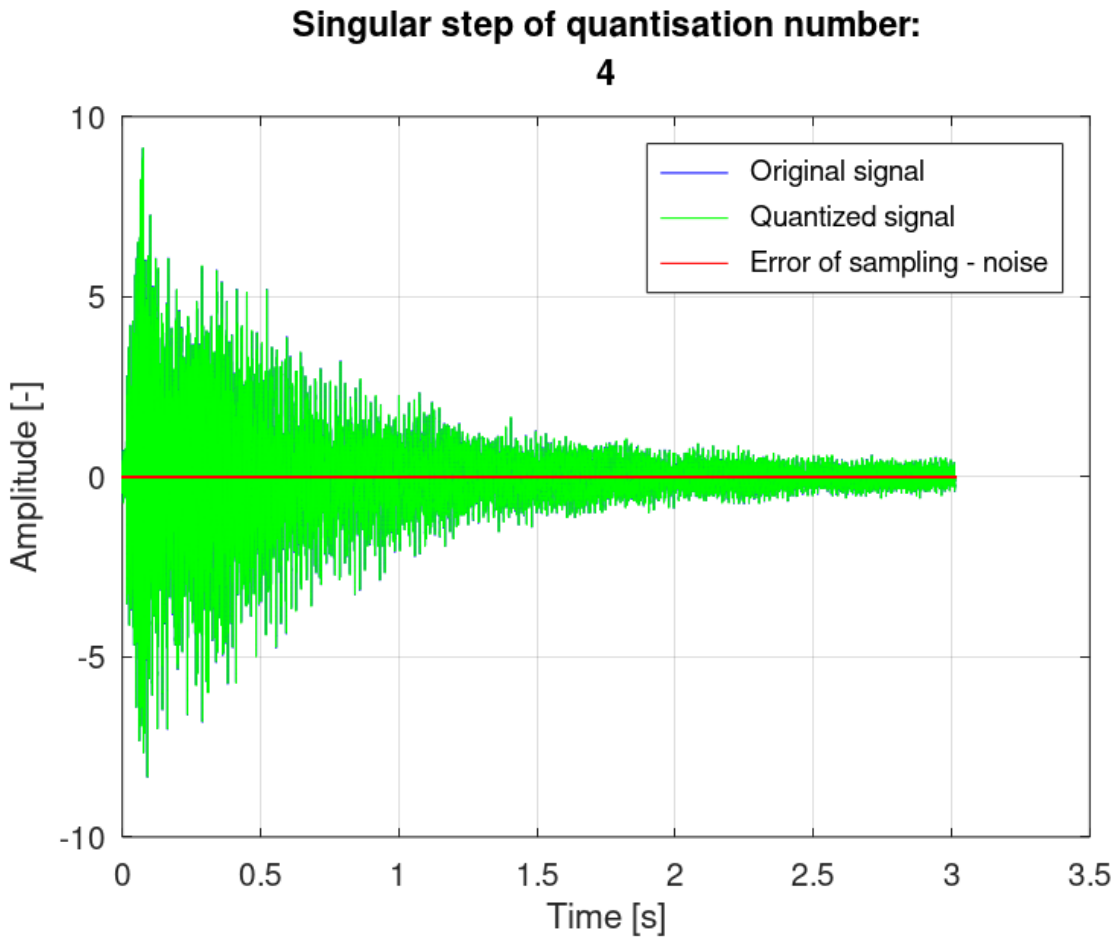

Phase (degrees)
Frequency (Hz)

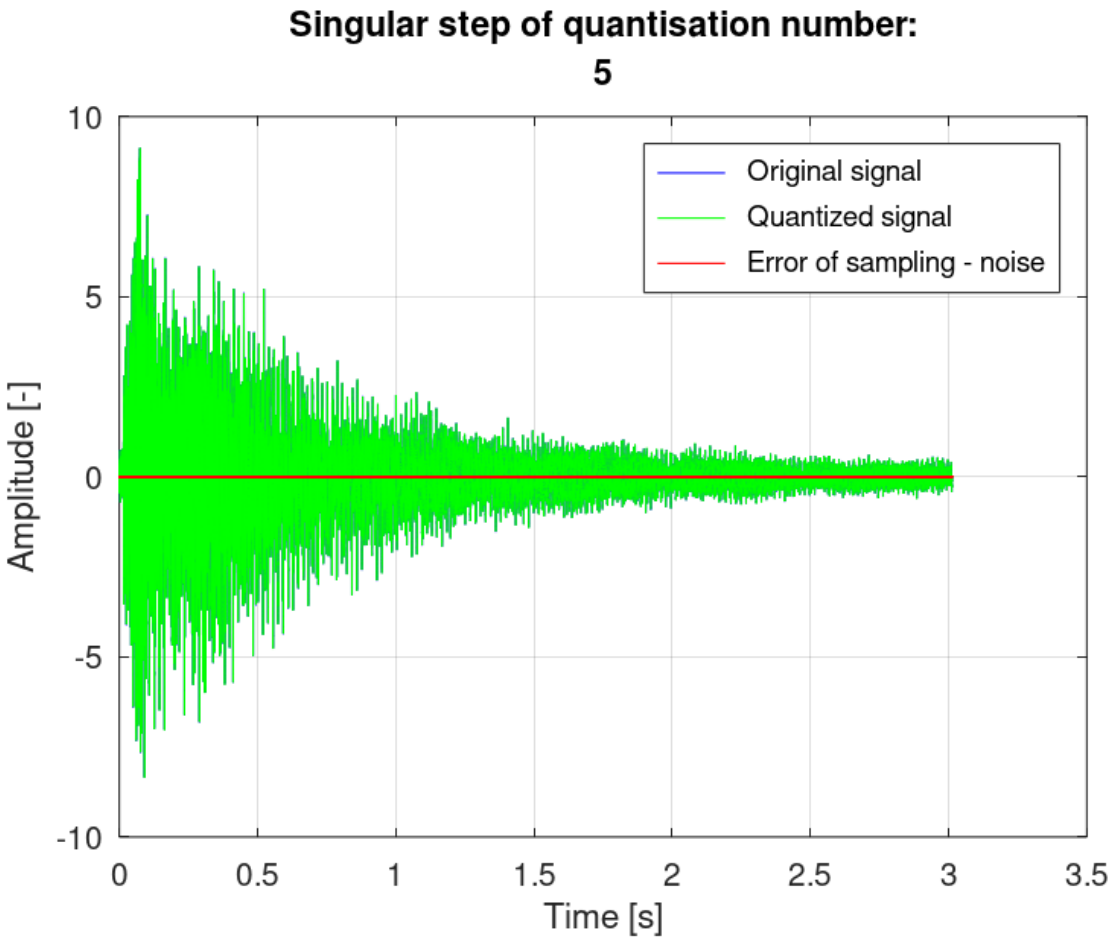
Am Chord

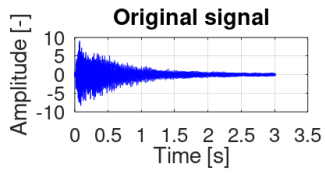

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

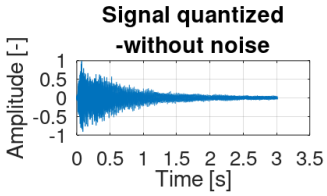

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

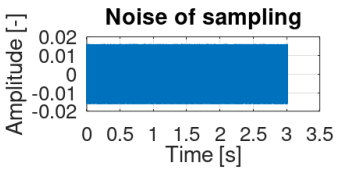

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

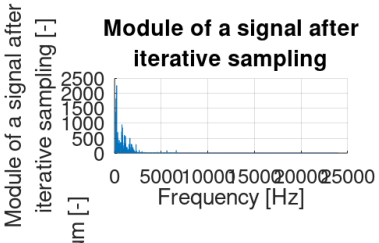

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

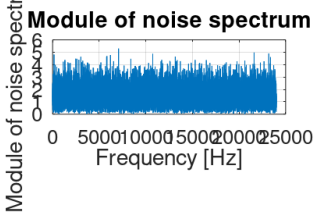

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

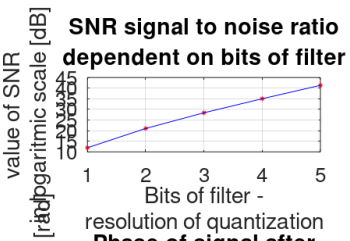

Original signal
Amplitude [-]
Time [s]

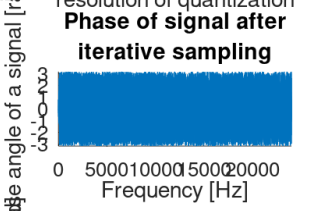

Signal quantized
-without noise
Amplitude [-]
Time [s]

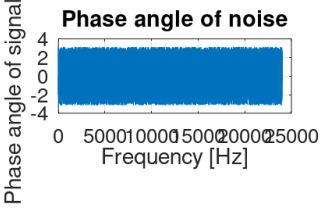

Noise of sampling
Amplitude [-]
Time [s]

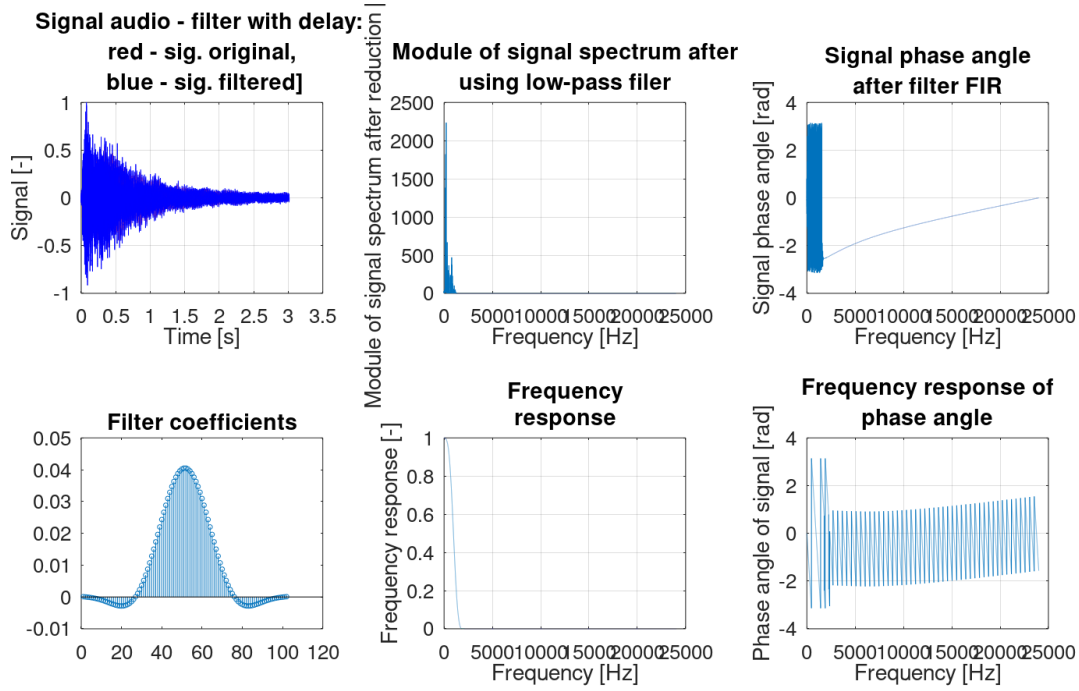

Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

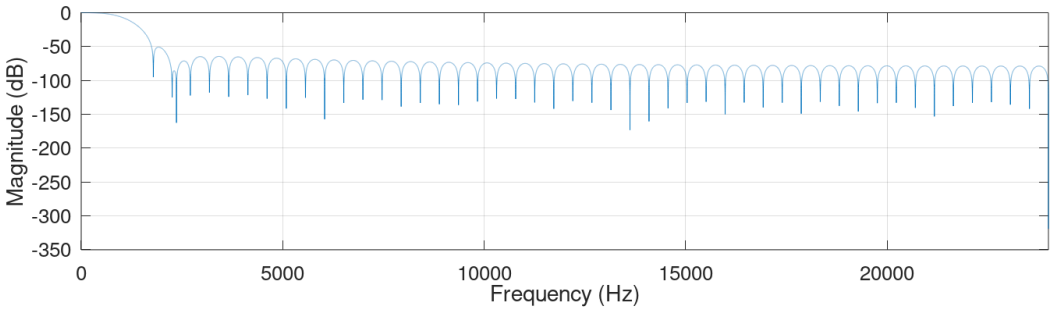

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

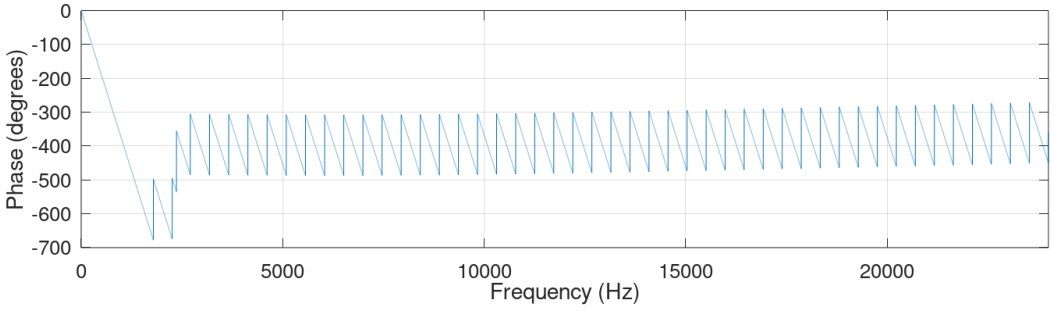

SNR signal to noise ratio
dependent on bits of filter
value of SNR logaritmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]

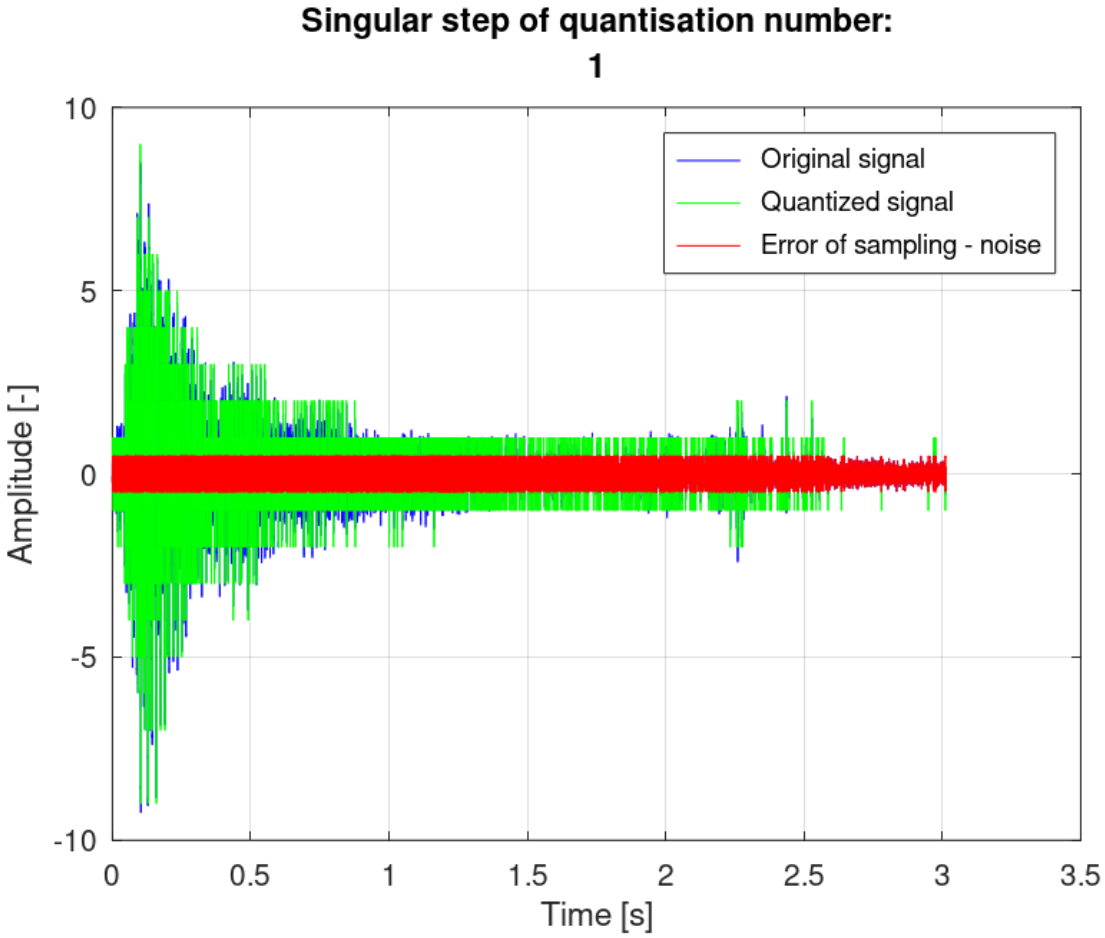

Phase angle of noise
Frequency [Hz]

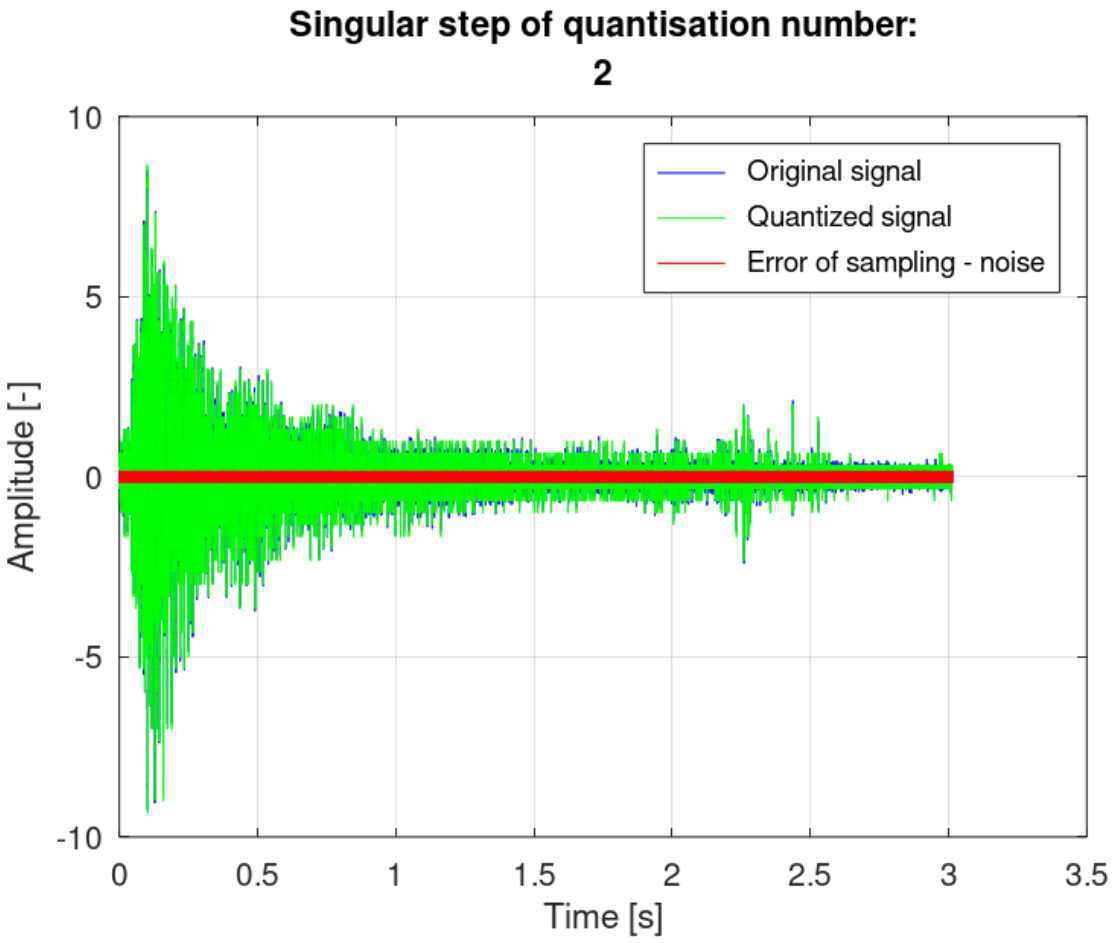

Signal audio - filter with delay: red - sig. original, blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after using low-pass filer
Module of signal spectrum after reduction
Frequency [Hz]
Signal phase angle after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency response
Frequency response [-]
Frequency [Hz]
Frequency response of phase angle
Phase angle of signal [rad]
Frequency [Hz]

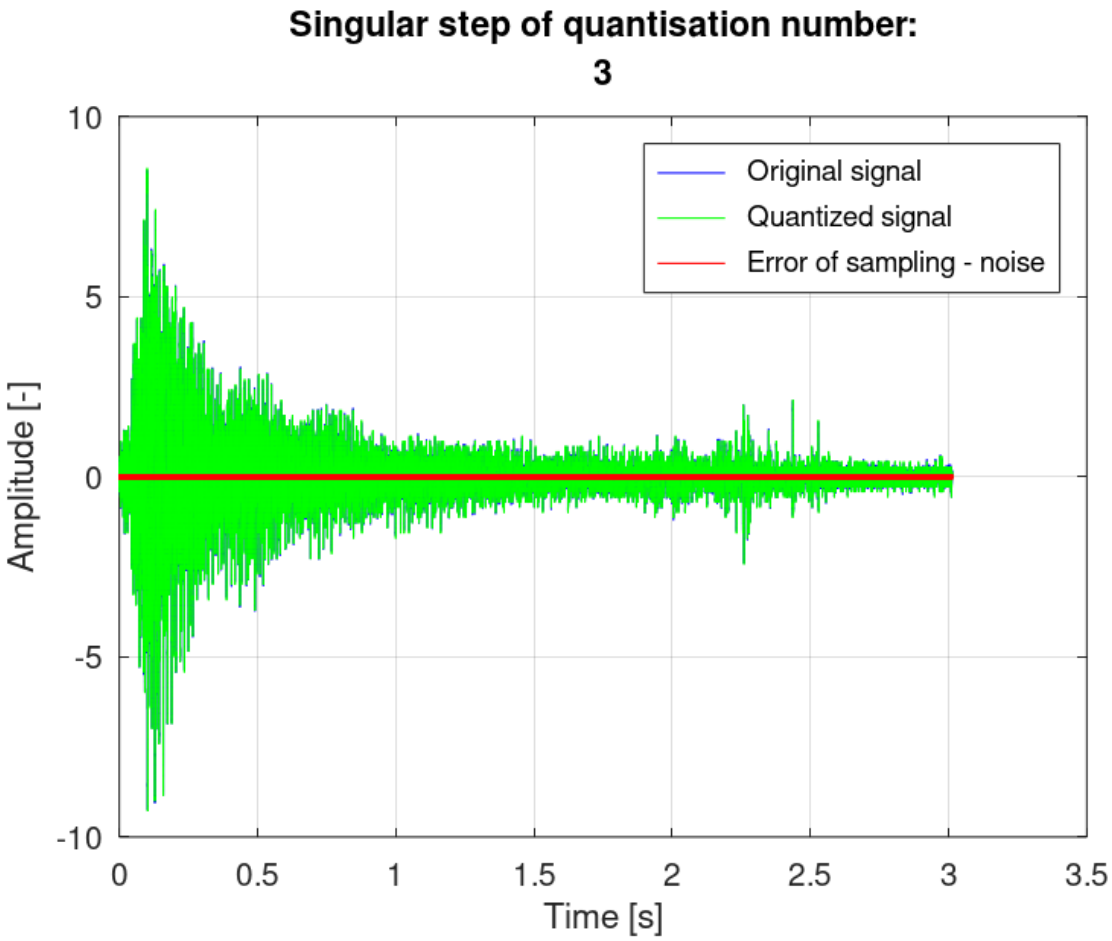

Magnitude (dB)
Frequency (Hz)

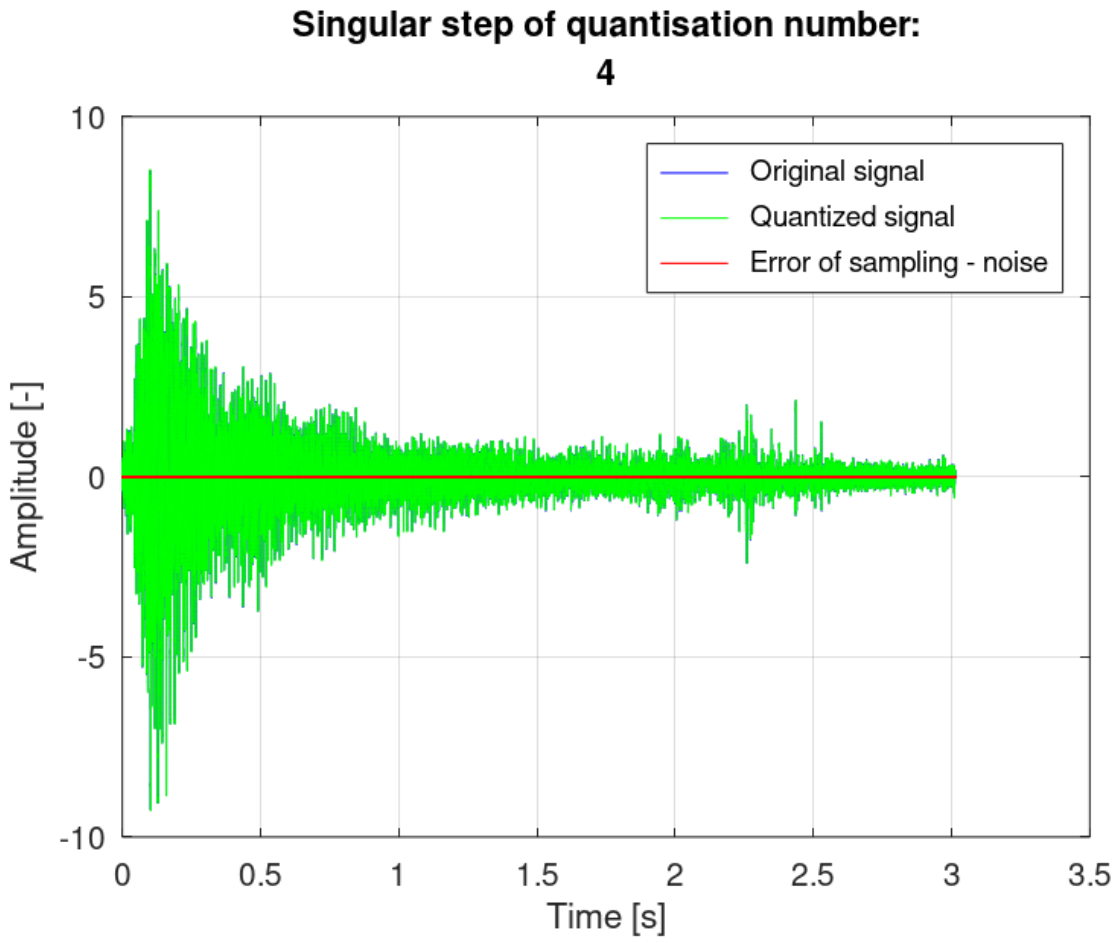

Phase (degrees)
Frequency (Hz)

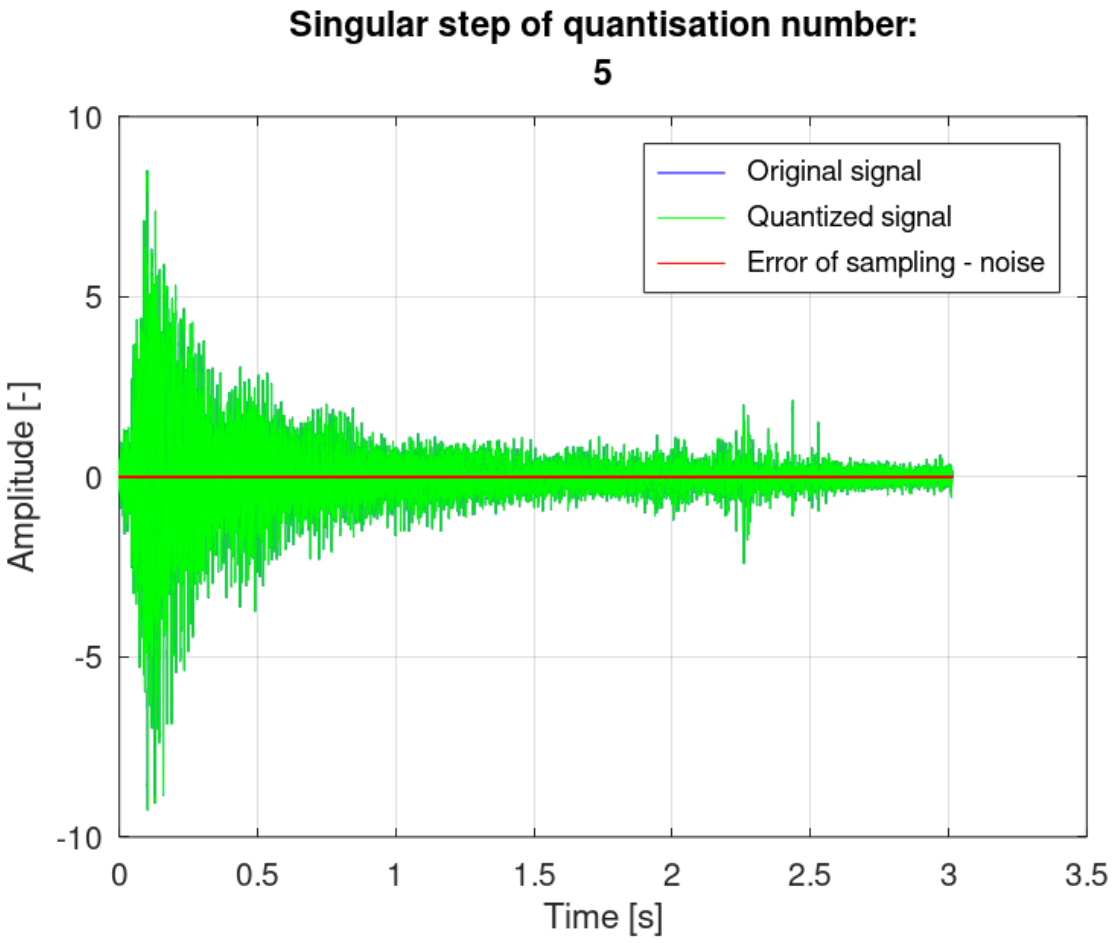
C Chord

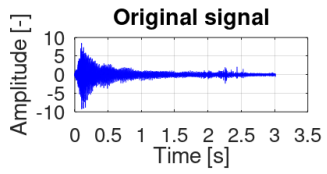

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

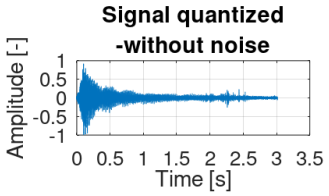

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

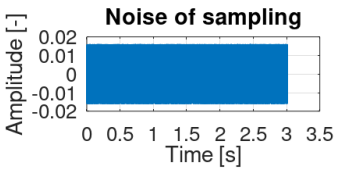

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

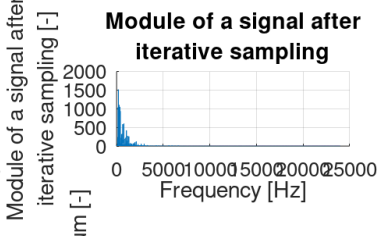

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

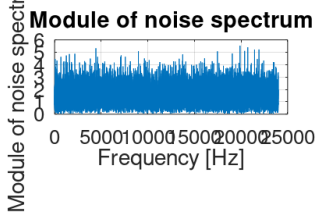

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

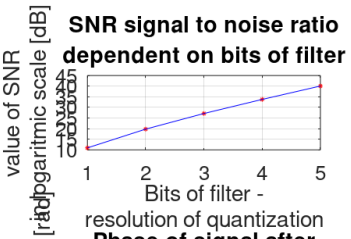

Original signal
Amplitude [-]
Time [s]

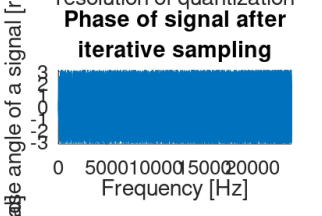

Signal quantized
-without noise
Amplitude [-]
Time [s]

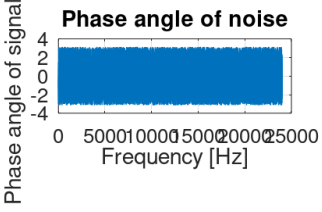

Noise of sampling
Amplitude [-]
Time [s]

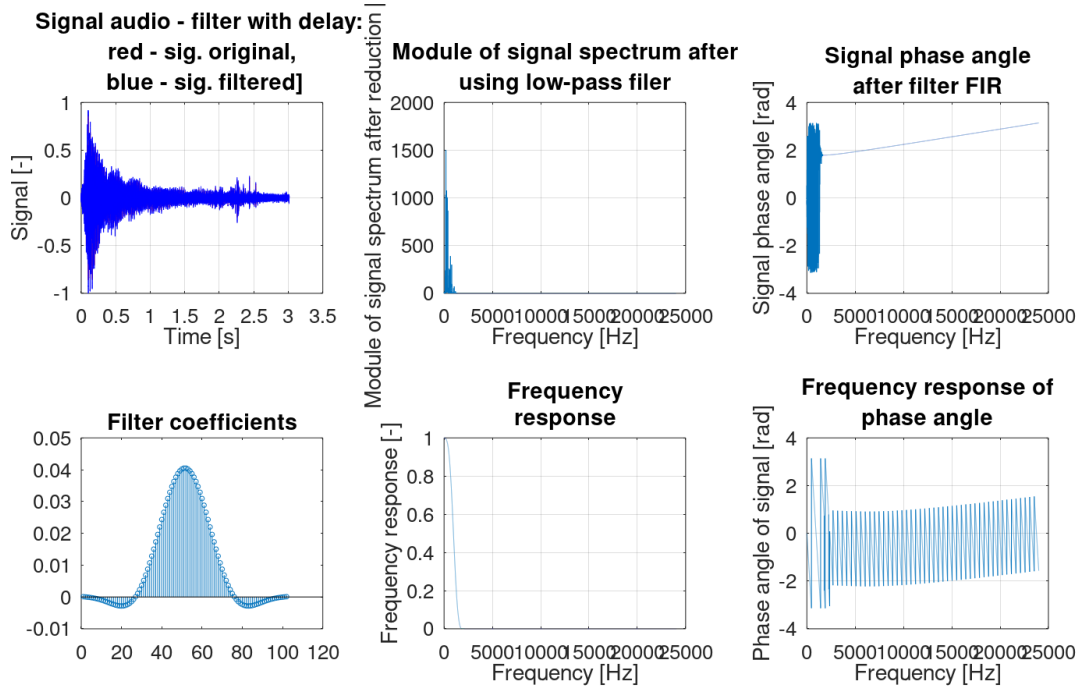

Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

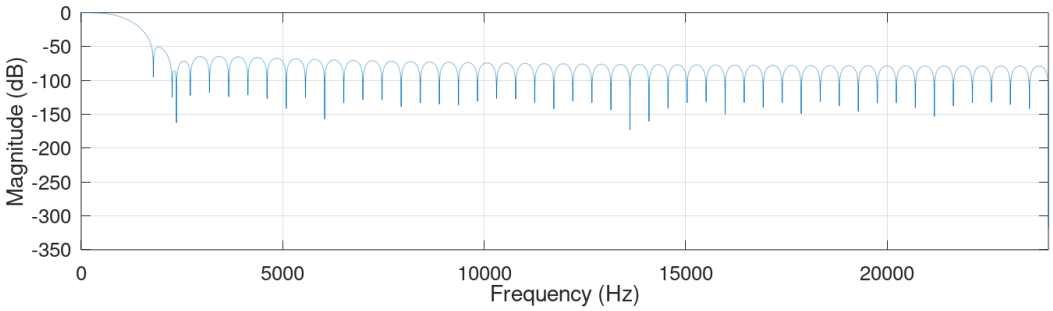

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

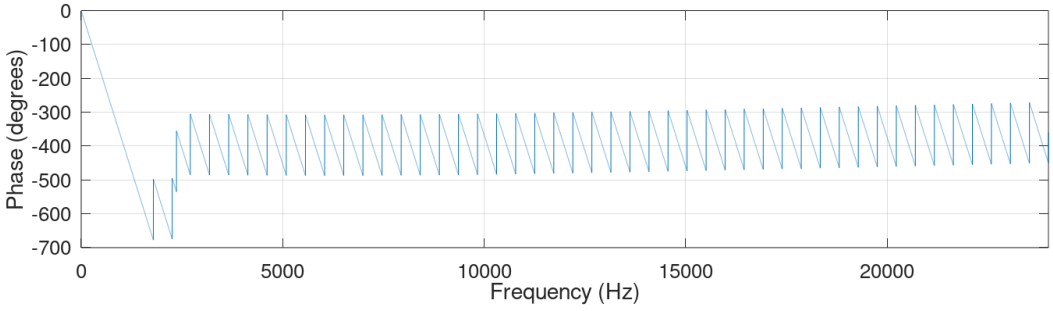

SNR signal to noise ratio
dependent on bits of filter
value of SNR logaritmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]

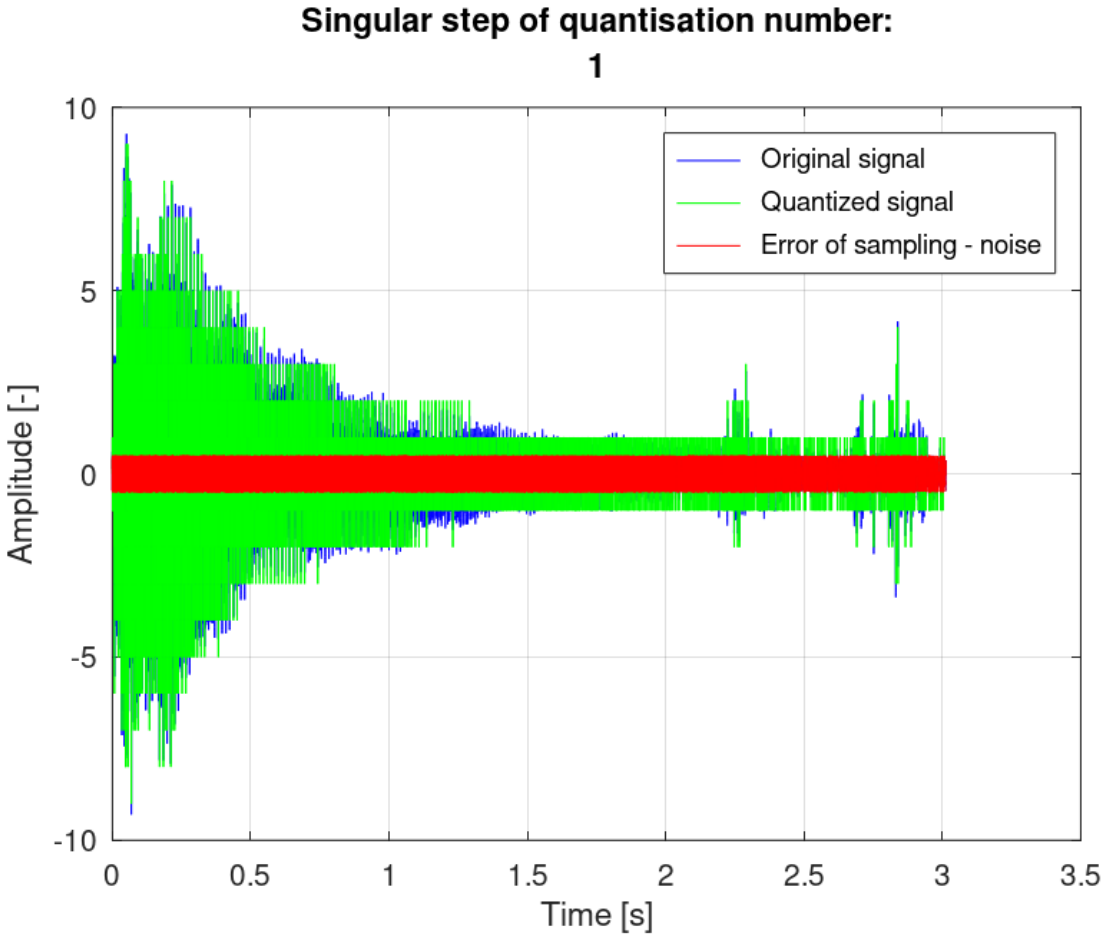

Phase angle of noise
Frequency [Hz]

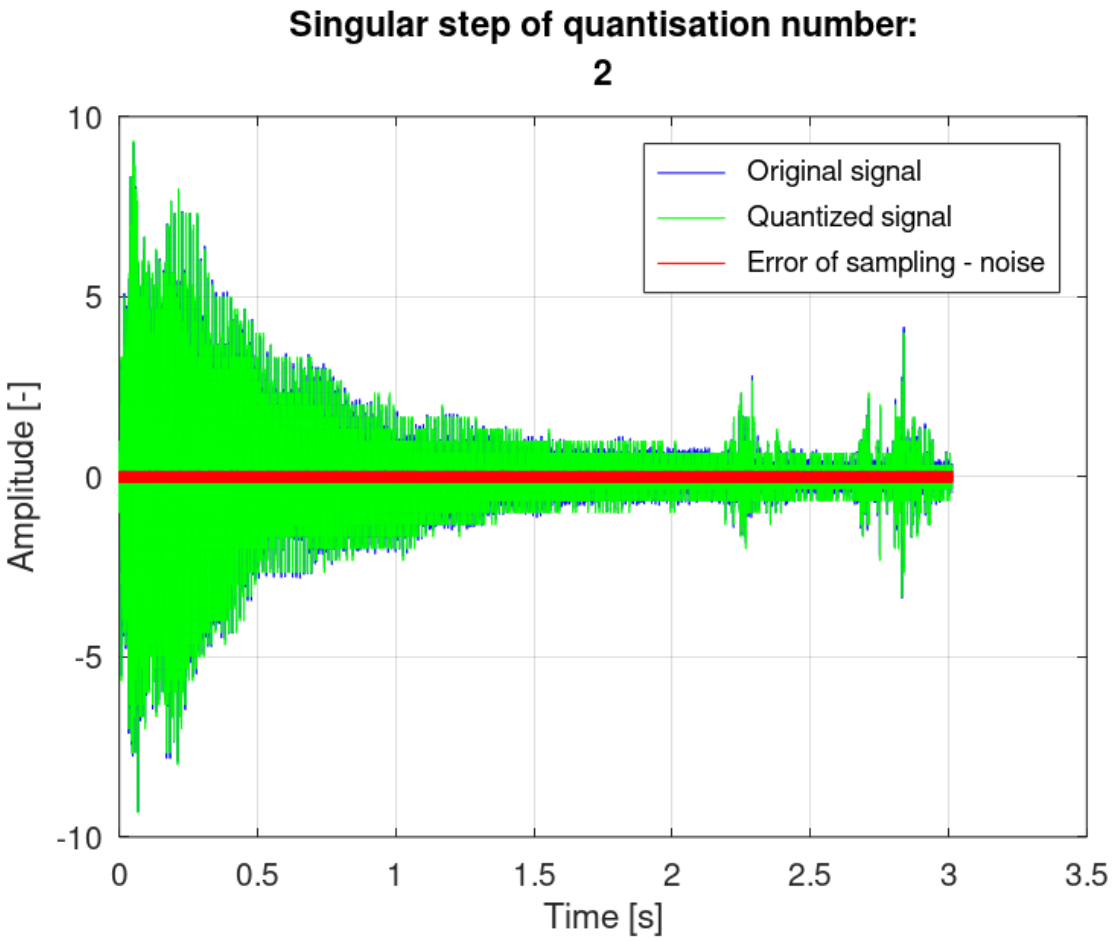

Signal audio - filter with delay:
red - sig. original,
blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after reduction
using low-pass filer
Module of signal spectrum after reduction |
Frequency [Hz]
Signal phase angle
after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency
response
Frequency response [-]
Frequency [Hz]
Frequency response of
phase angle
Phase angle of signal [rad]
Frequency [Hz]

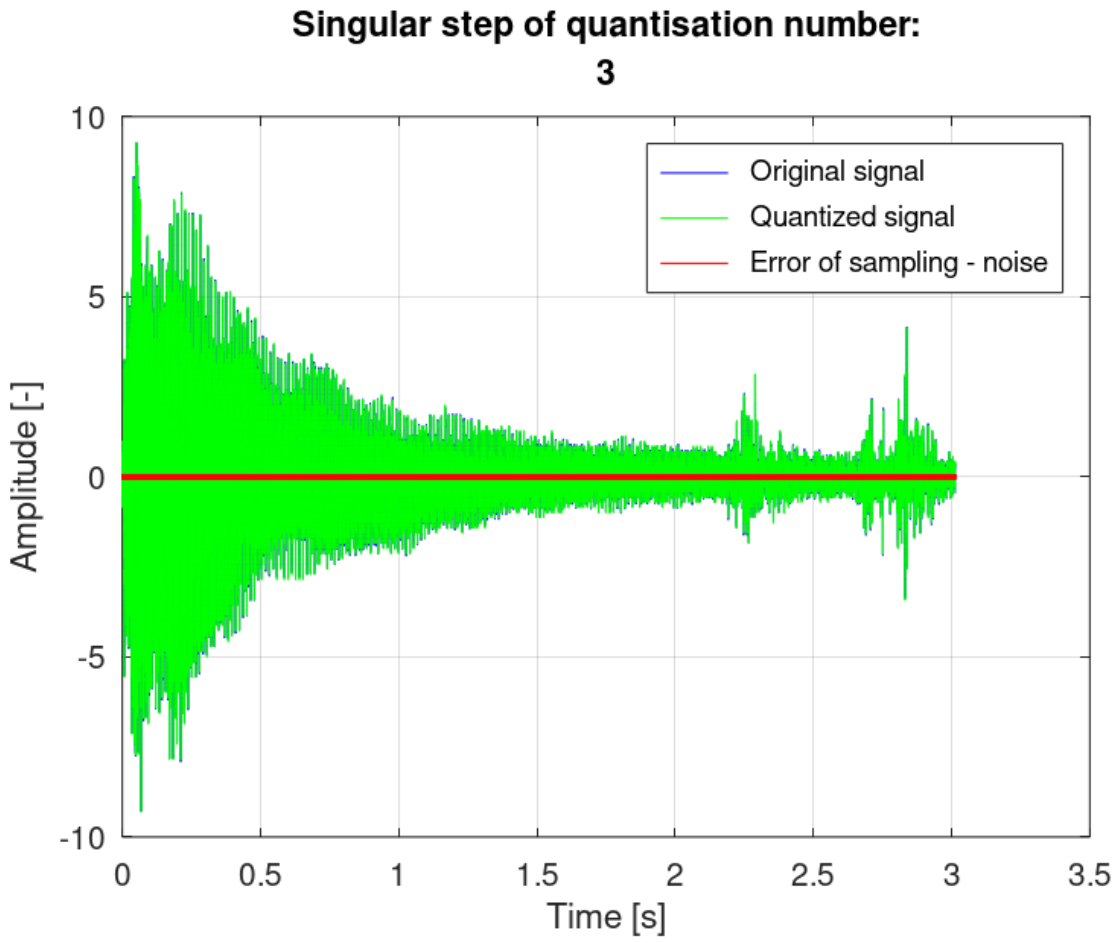

Magnitude (dB)
Frequency (Hz)

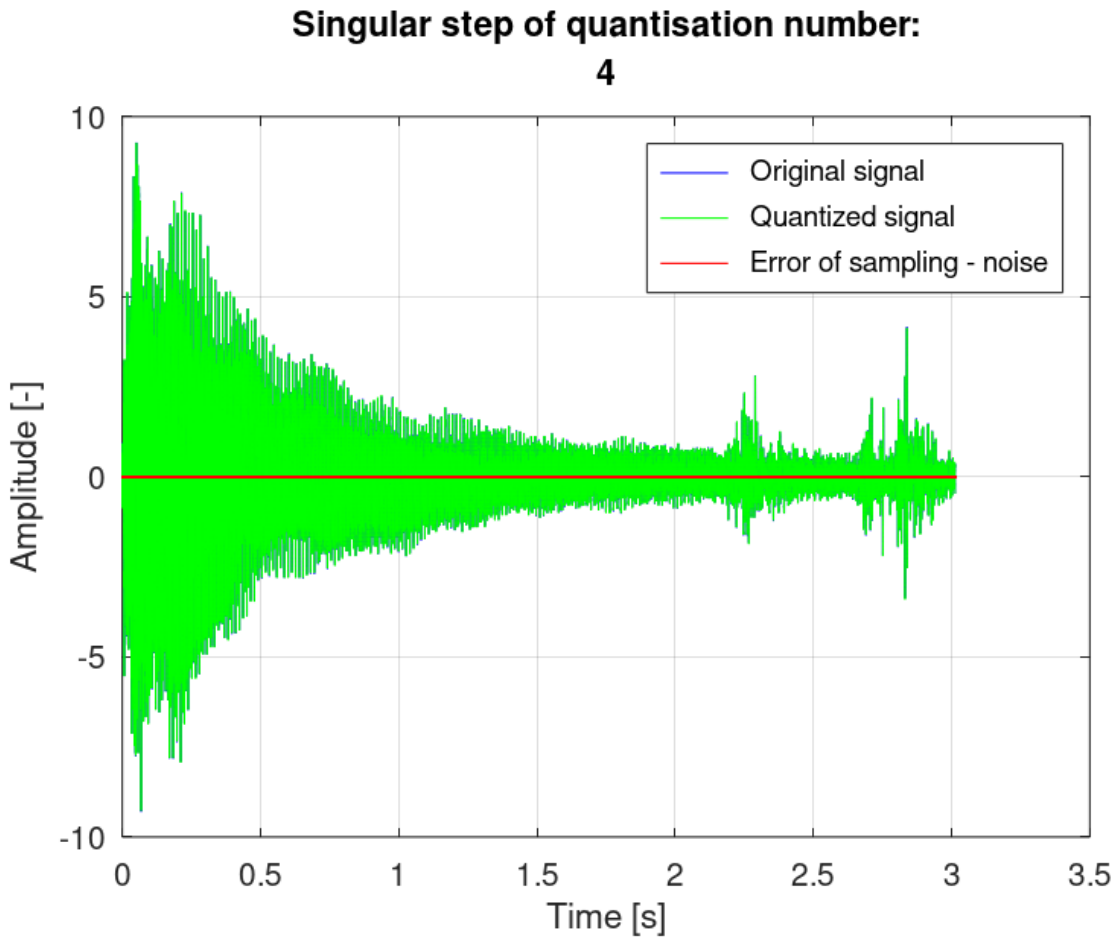

Phase (degrees)
Frequency (Hz)

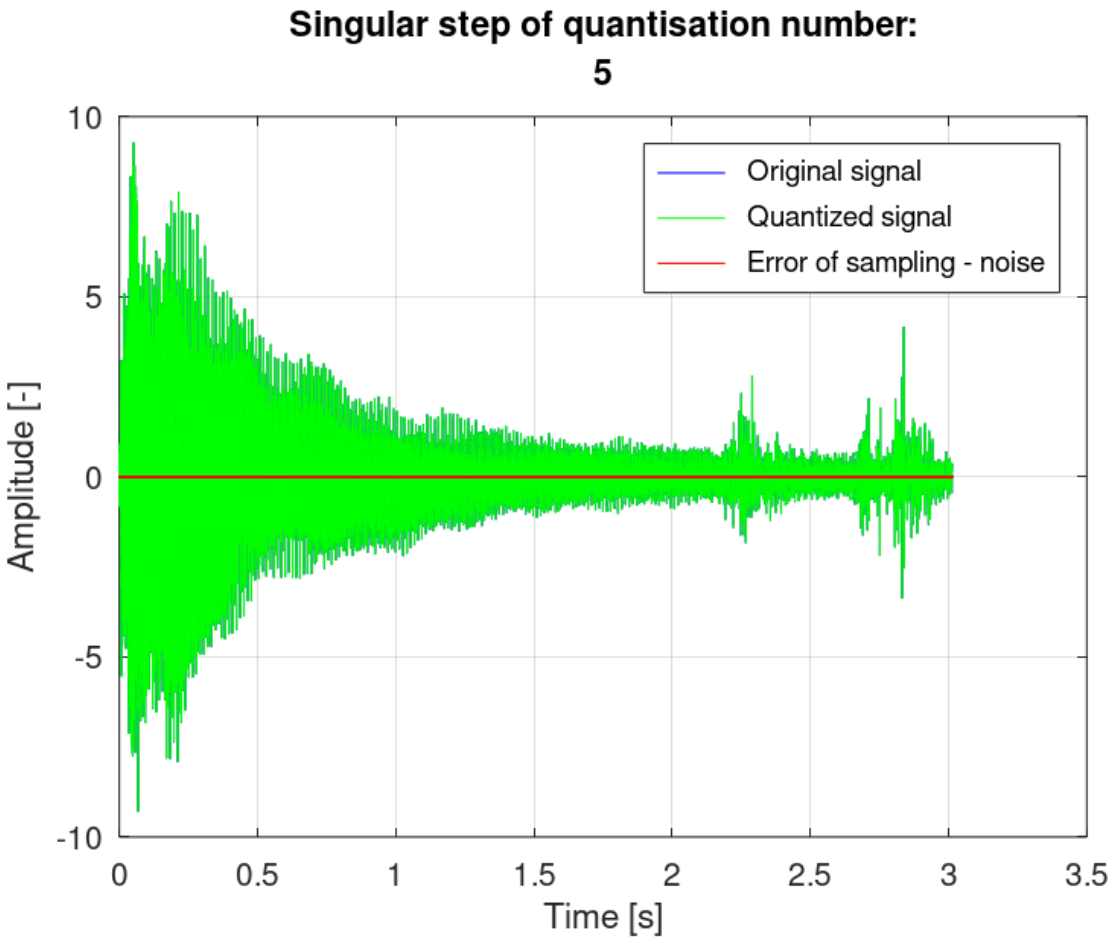
D Chord

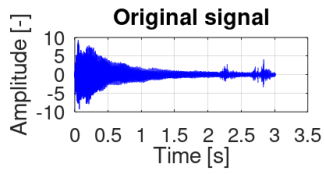

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

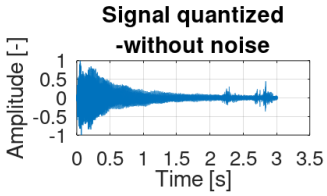

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

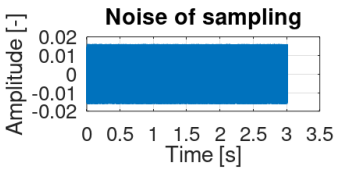

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

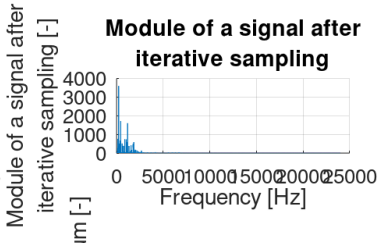

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

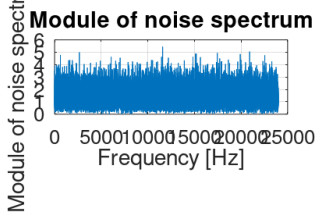

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

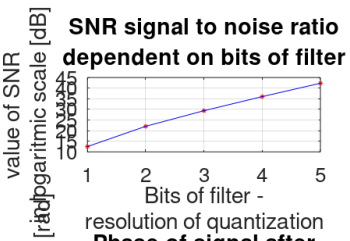

Original signal
Amplitude [-]
Time [s]

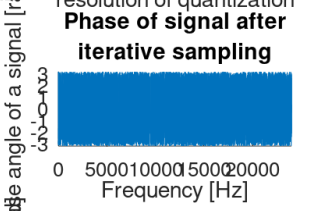

Signal quantized
-without noise
Amplitude [-]
Time [s]

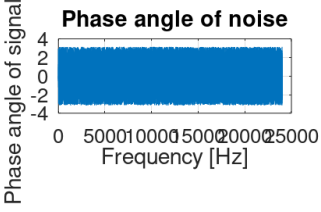

Noise of sampling
Amplitude [-]
Time [s]

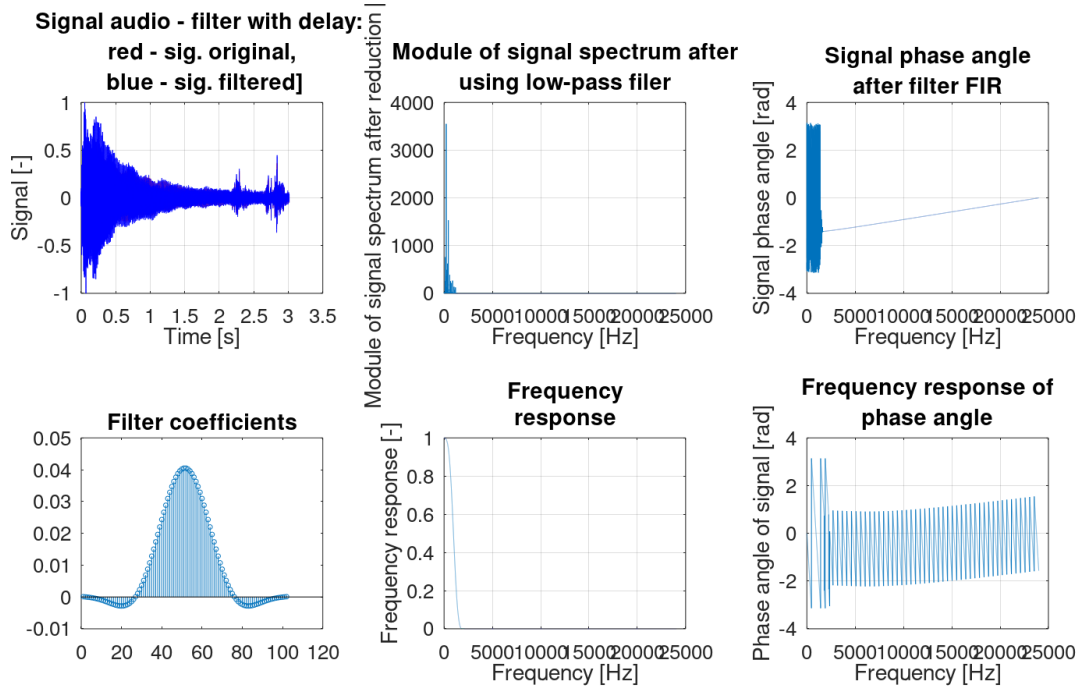

Module of a signal after iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

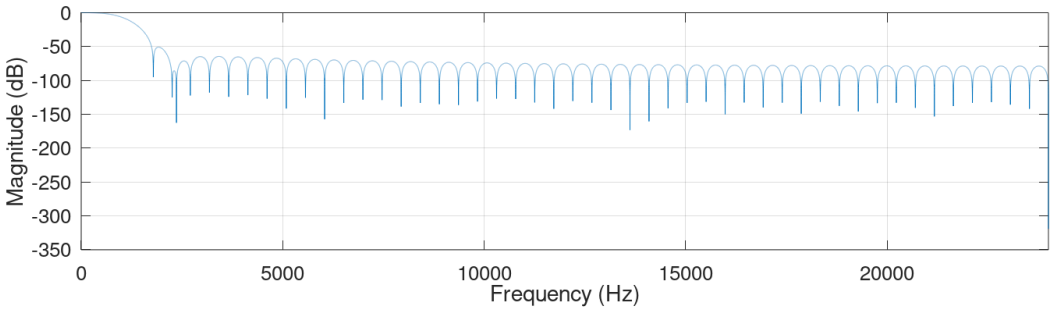

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

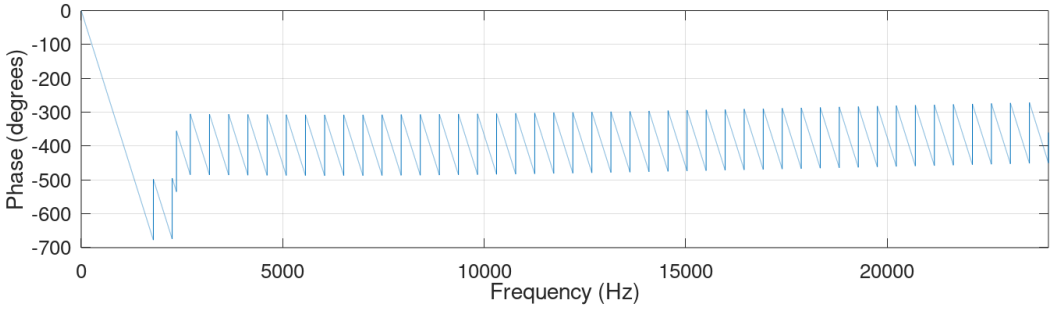

SNR signal to noise ratio
dependent on bits of filter
value of SNR logaritmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]

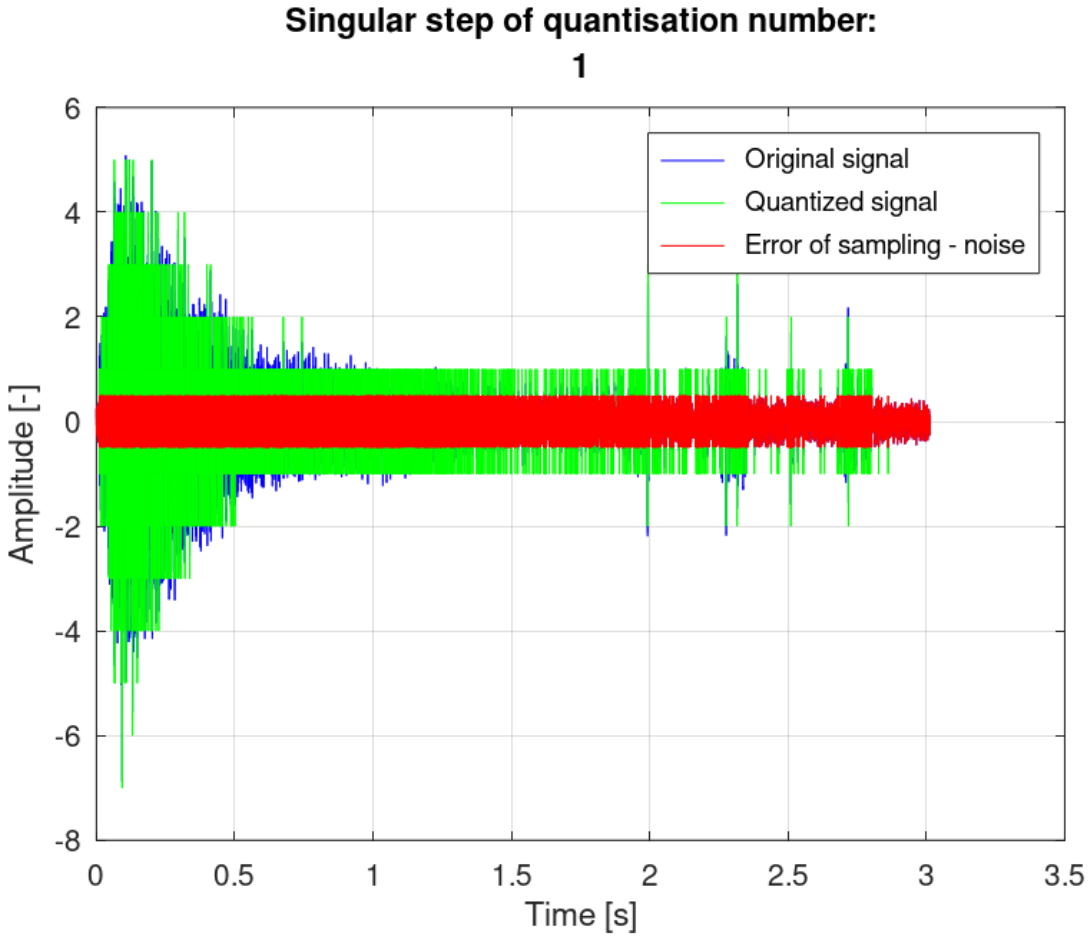

Phase angle of noise
Frequency [Hz]

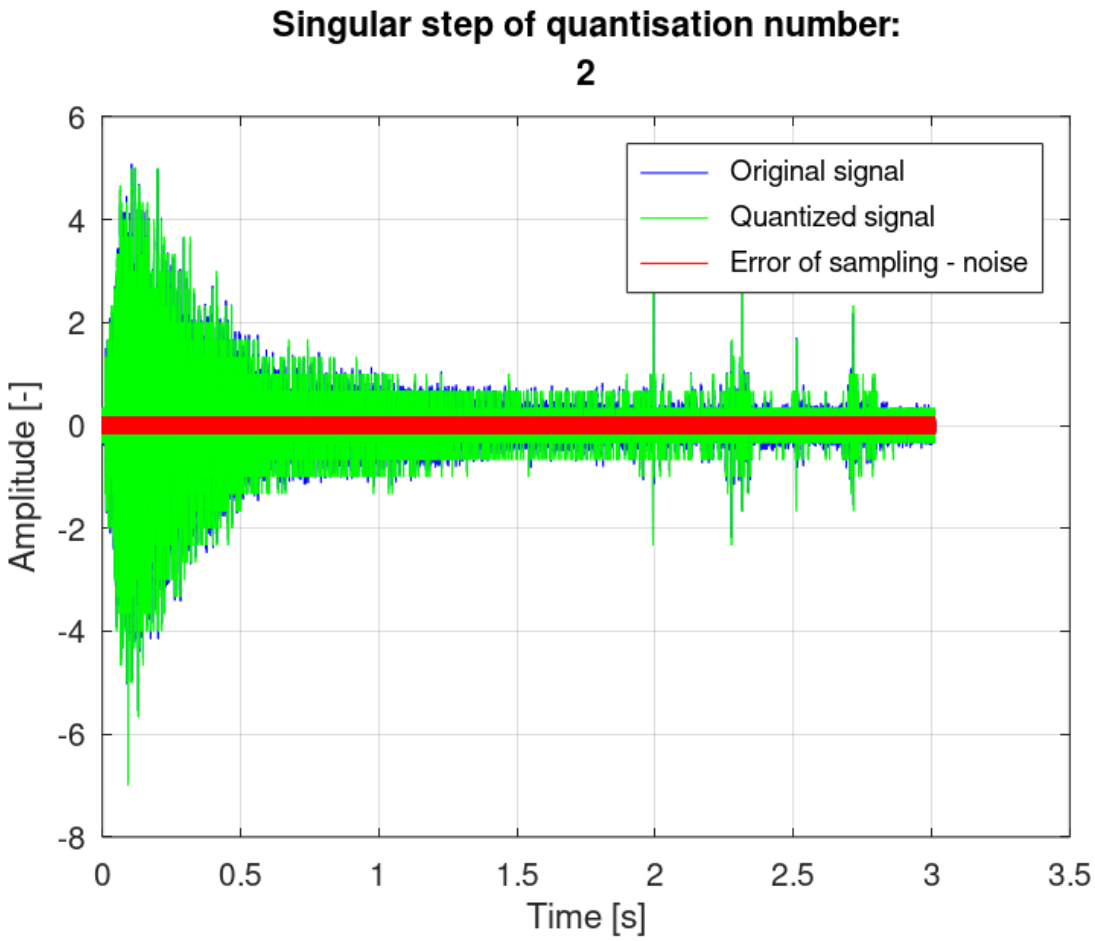

Signal audio - filter with delay:
red - sig. original,
blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after reduction
using low-pass filer
Module of signal spectrum after reduction |
Frequency [Hz]
Signal phase angle
after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency
response
Frequency response [-]
Frequency [Hz]
Frequency response of
phase angle
Phase angle of signal [rad]
Frequency [Hz]

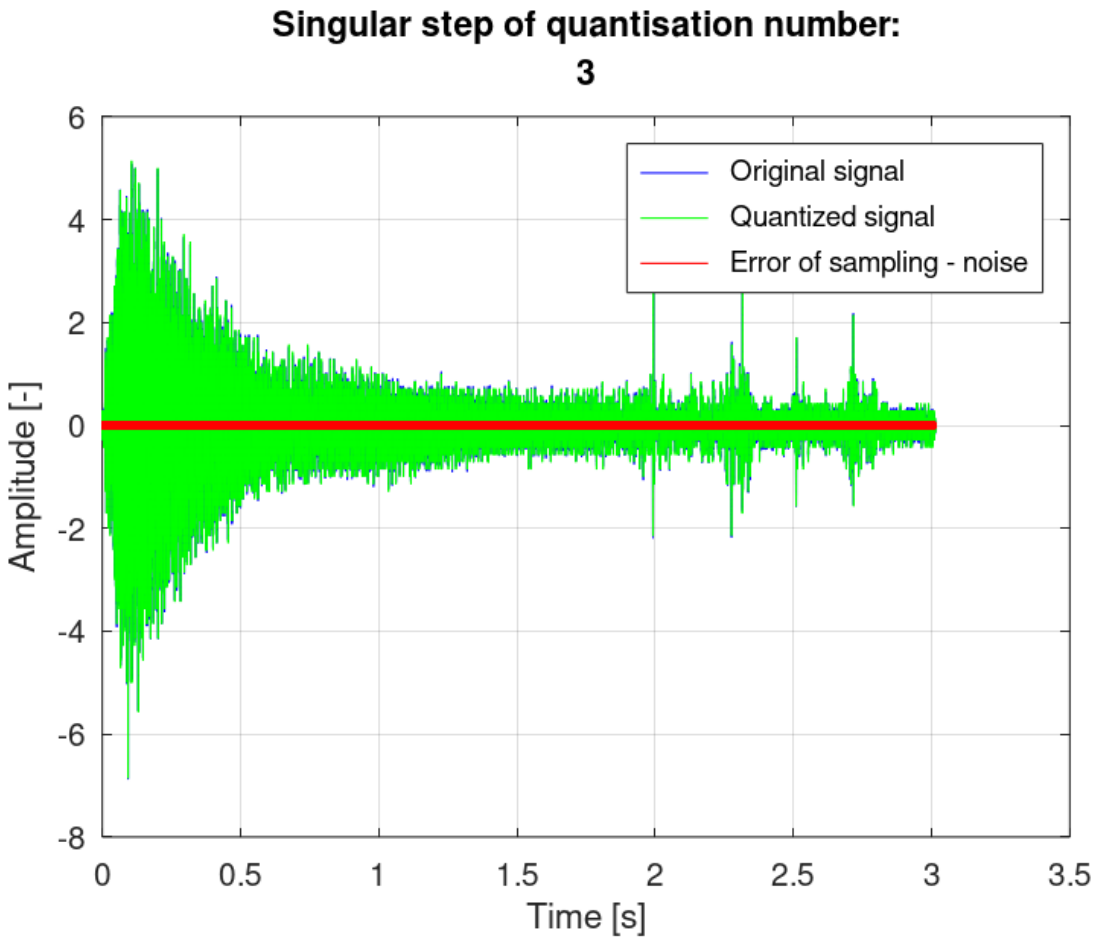

Magnitude (dB)
Frequency (Hz)

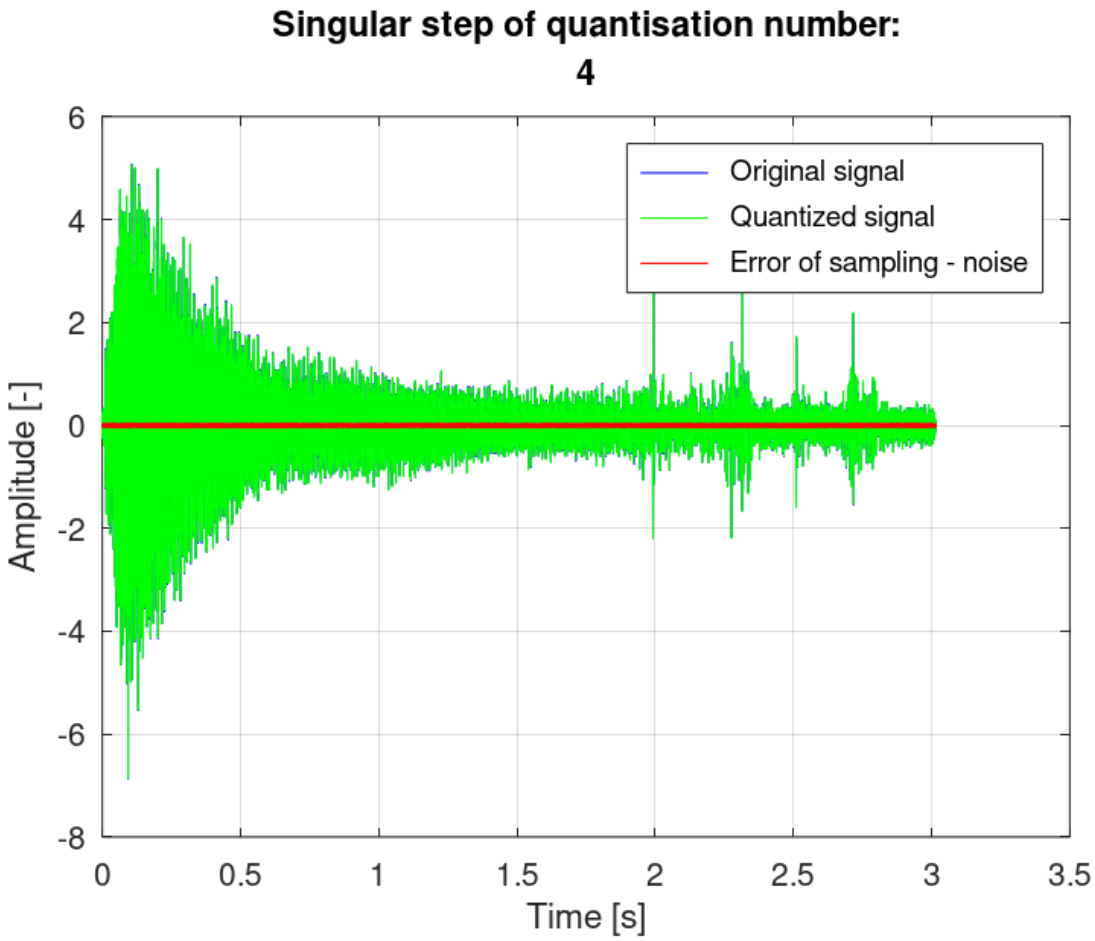

Phase (degrees)
Frequency (Hz)

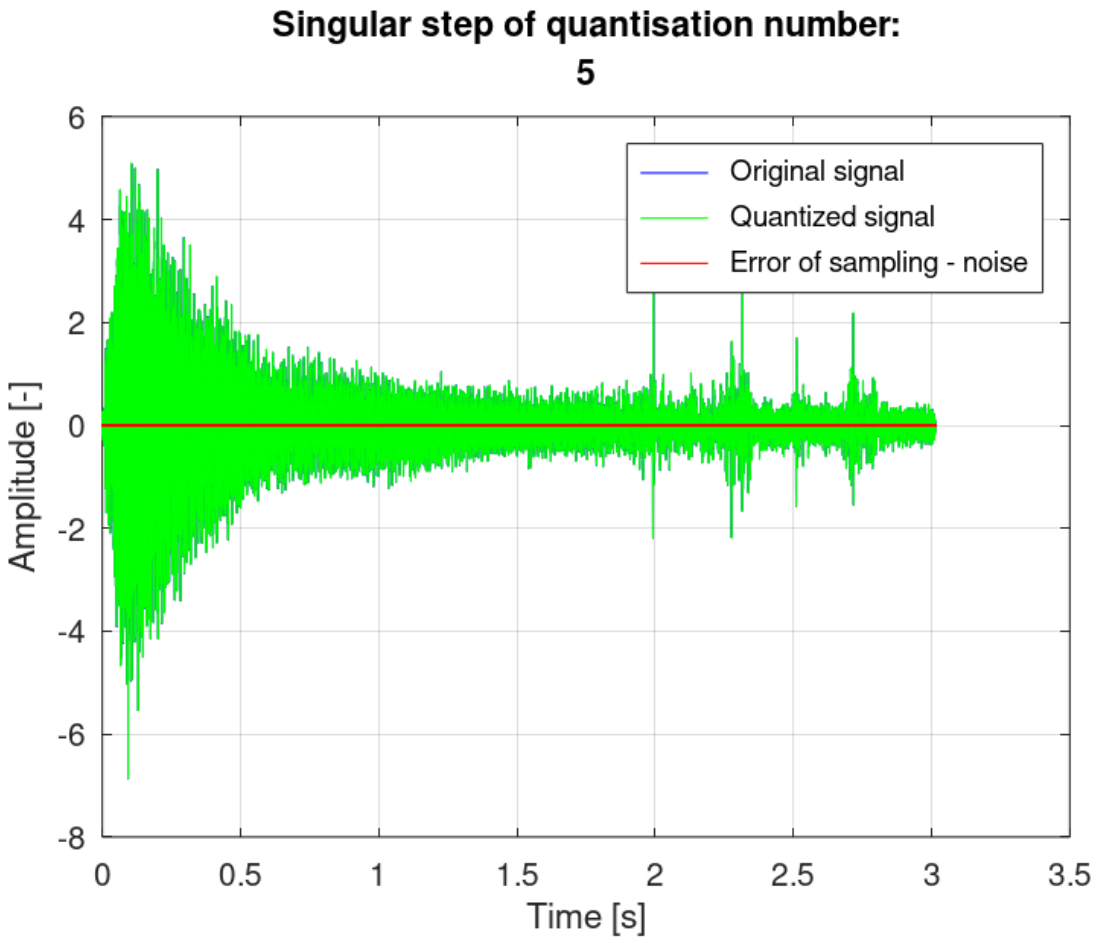
Dm Chord

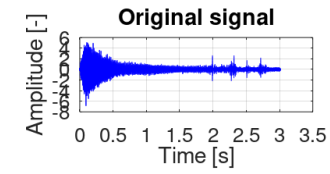

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
6
4
2
0
-2
-4
-6
-8
0
0.5
1
1.5
2
2.5
3
3.5

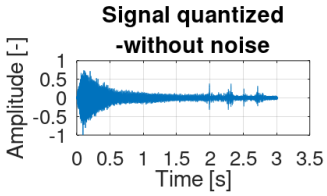

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
6
4
2
0
-2
-4
-6
-8
0
0.5
1
1.5
2
2.5
3
3.5

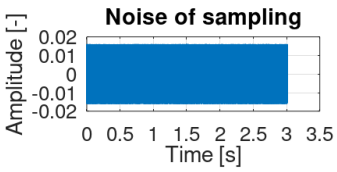

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
6
4
2
0
-2
-4
-6
-8
0
0.5
1
1.5
2
2.5
3
3.5

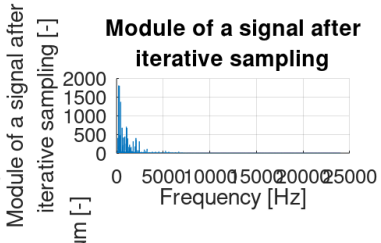

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
6
4
2
0
-2
-4
-6
-8
0
0.5
1
1.5
2
2.5
3
3.5

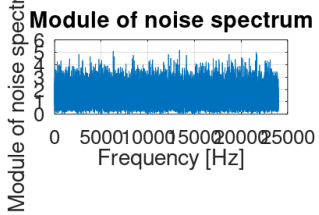

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

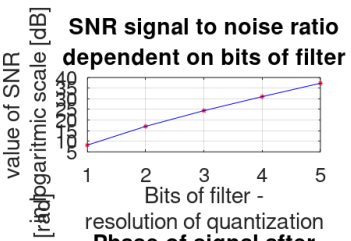

Original signal
Amplitude [-]
Time [s]

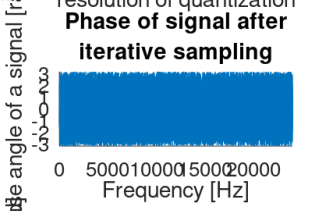

Signal quantized
-without noise
Amplitude [-]
Time [s]

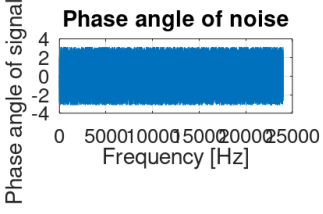

Noise of sampling
Amplitude [-]
Time [s]

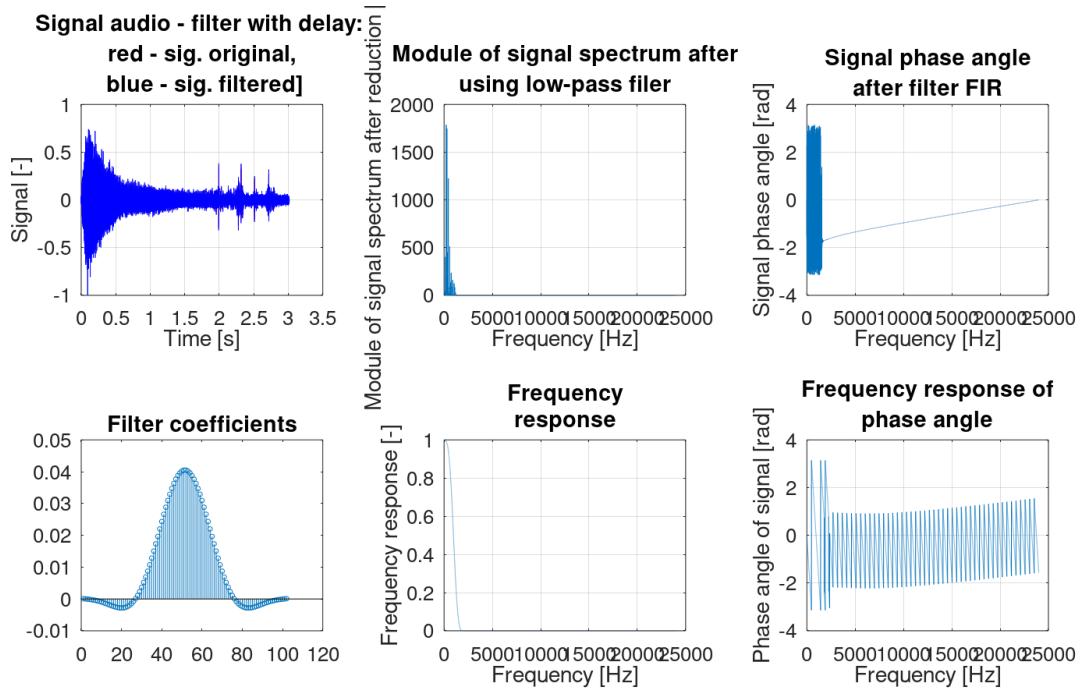

Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

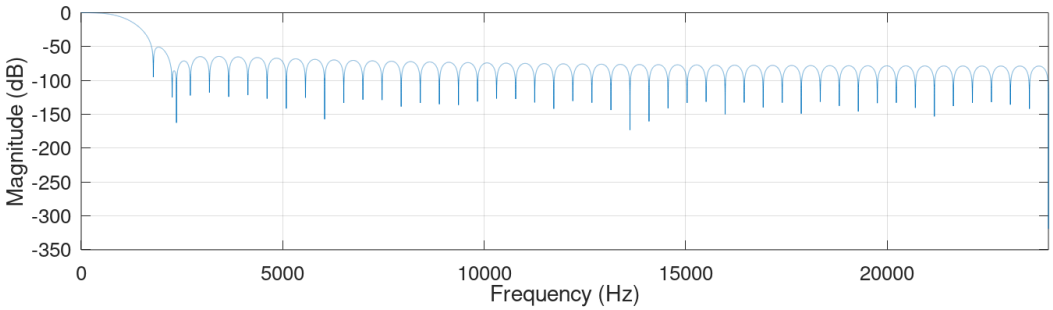

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

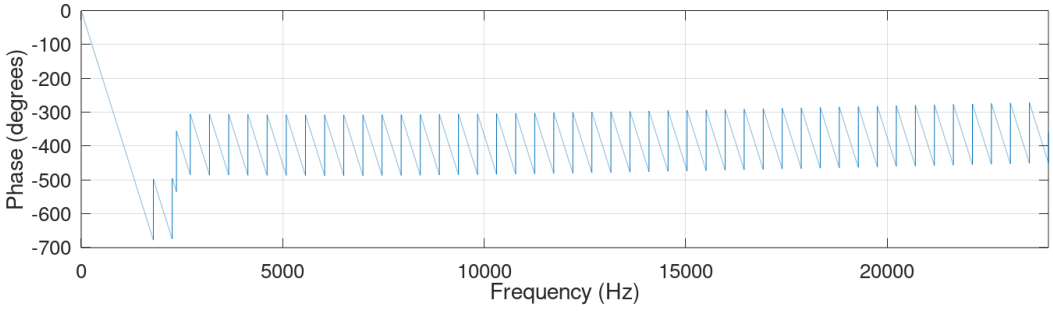

SNR signal to noise ratio
dependent on bits of filter
value of SNR logarithmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]

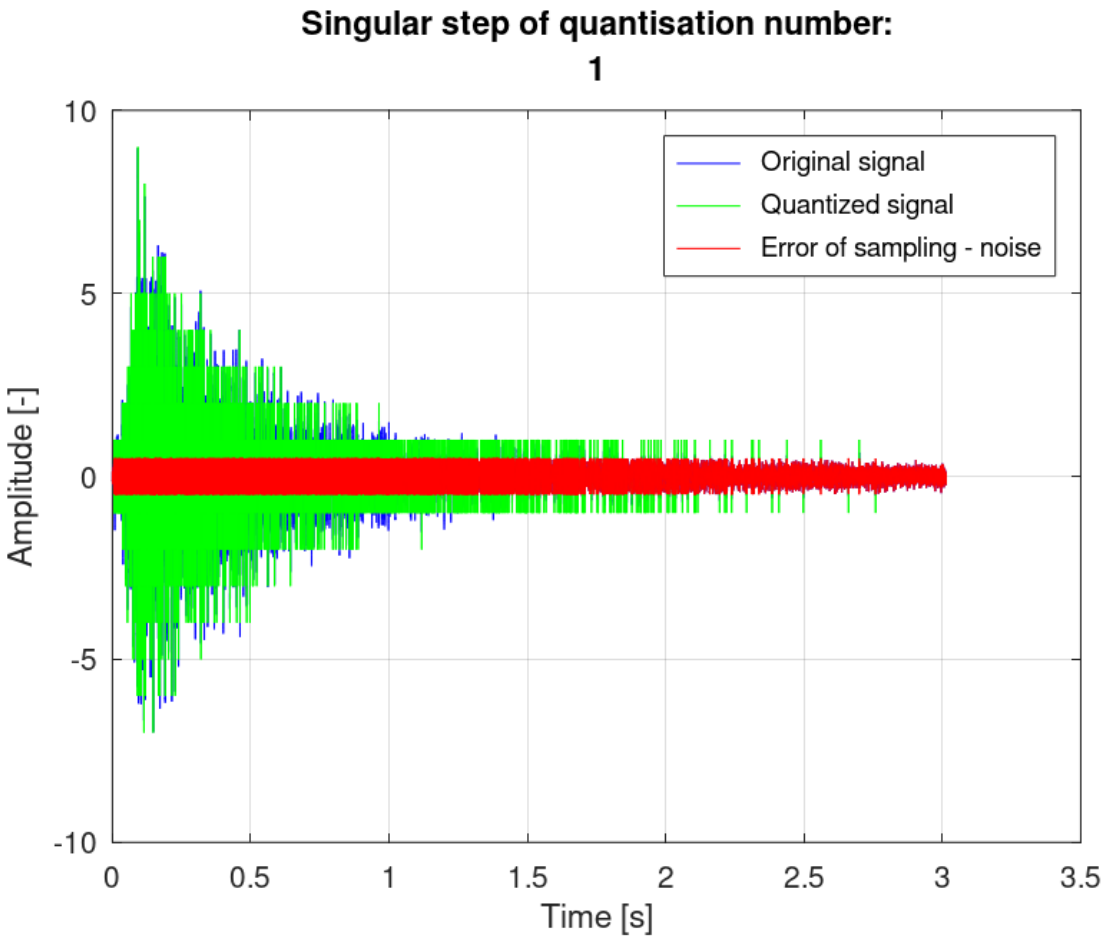

Phase angle of noise
Frequency [Hz]

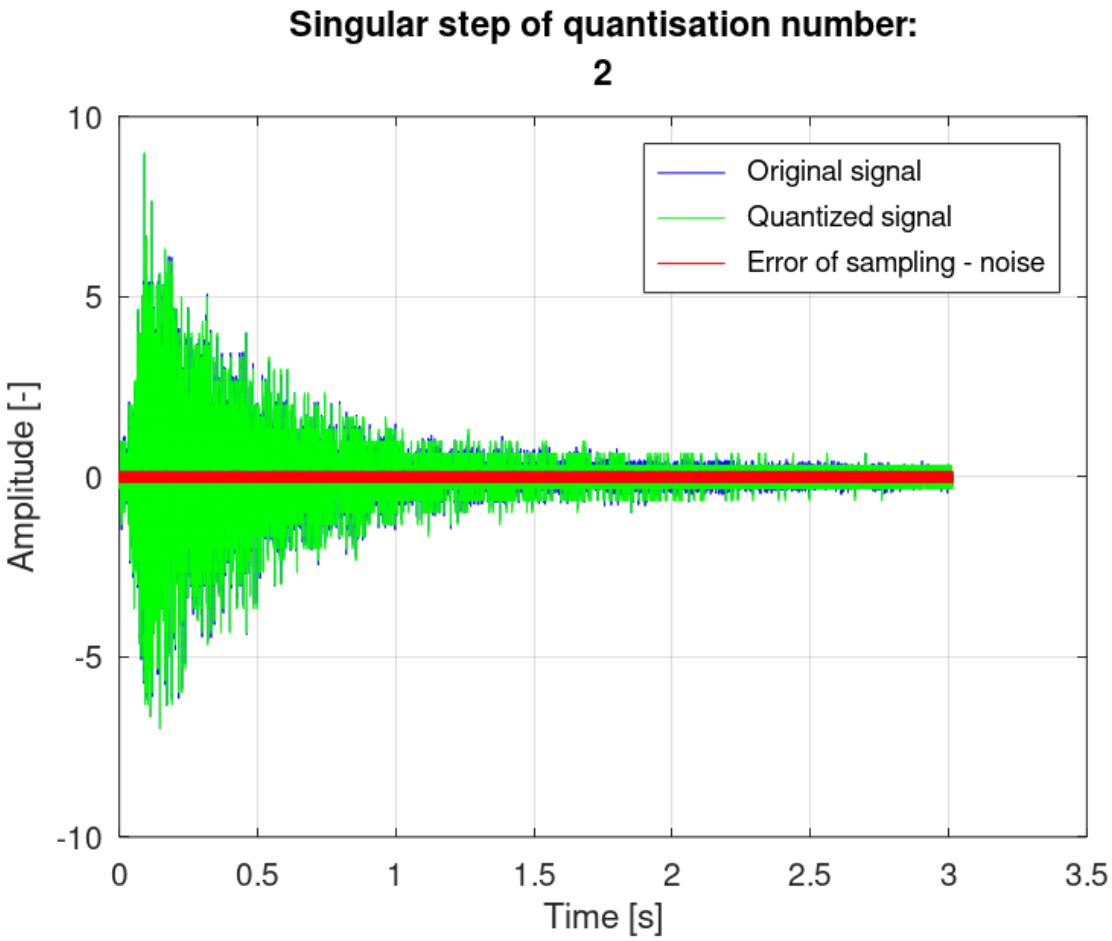

Signal audio - filter with delay:
red - sig. original,
blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after reduction
using low-pass filer
Module of signal spectrum after reduction |
Frequency [Hz]
Signal phase angle
after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency
response
Frequency response [-]
Frequency [Hz]
Frequency response of
phase angle
Phase angle of signal [rad]
Frequency [Hz]

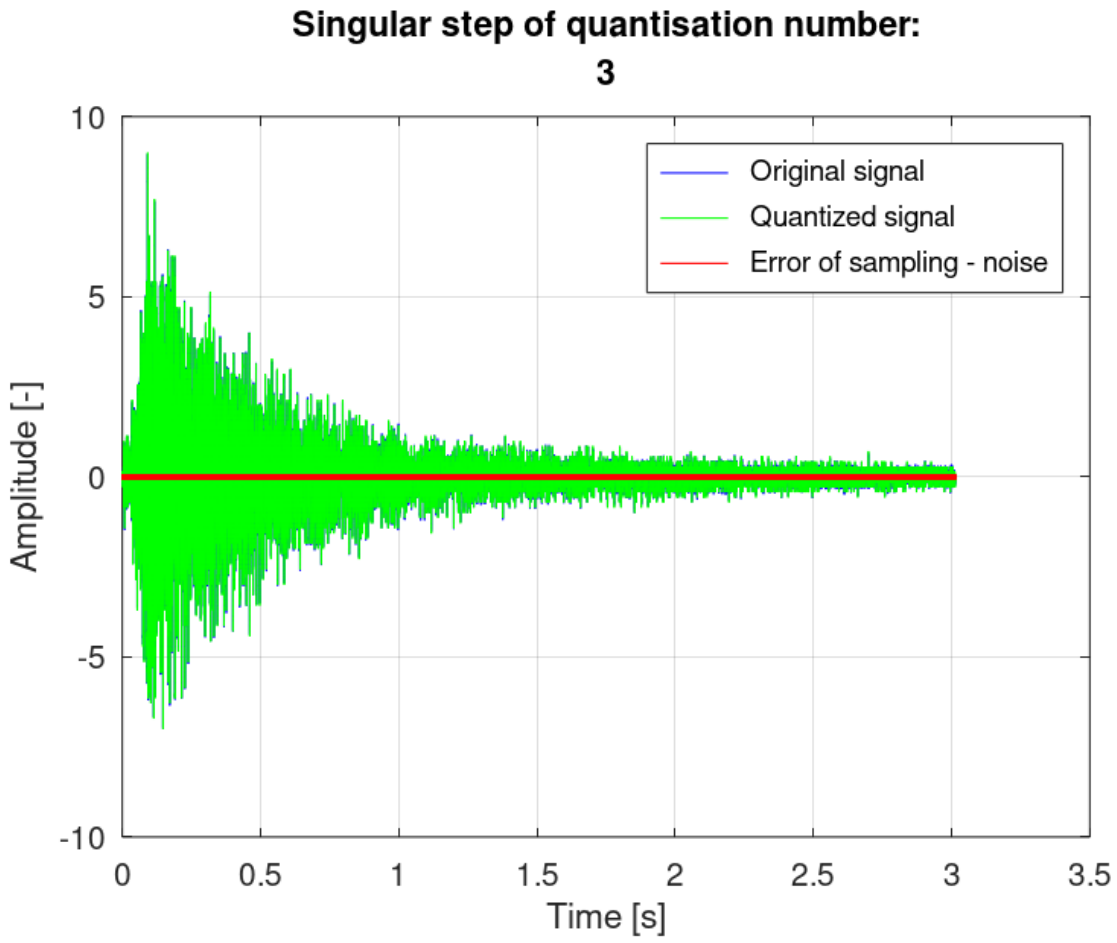

Magnitude (dB)
Frequency (Hz)

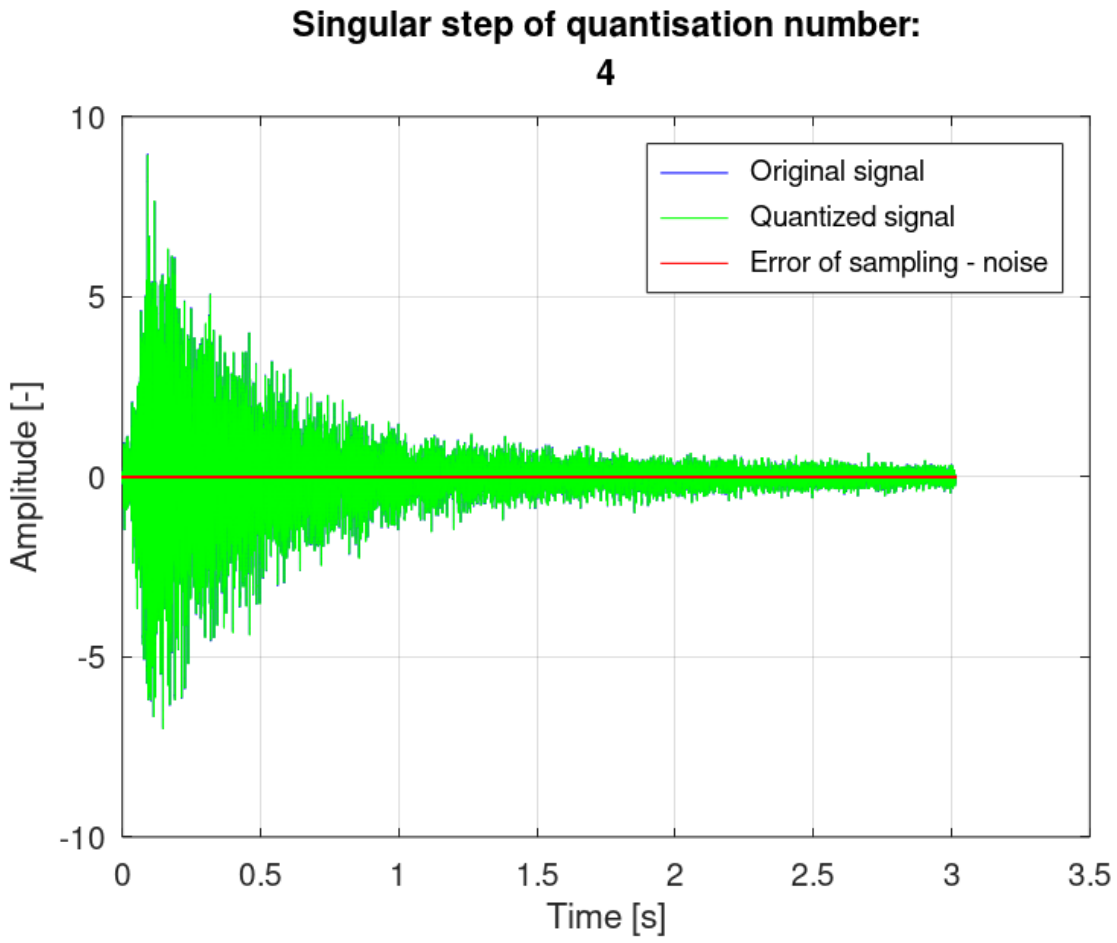

Phase (degrees)
Frequency (Hz)

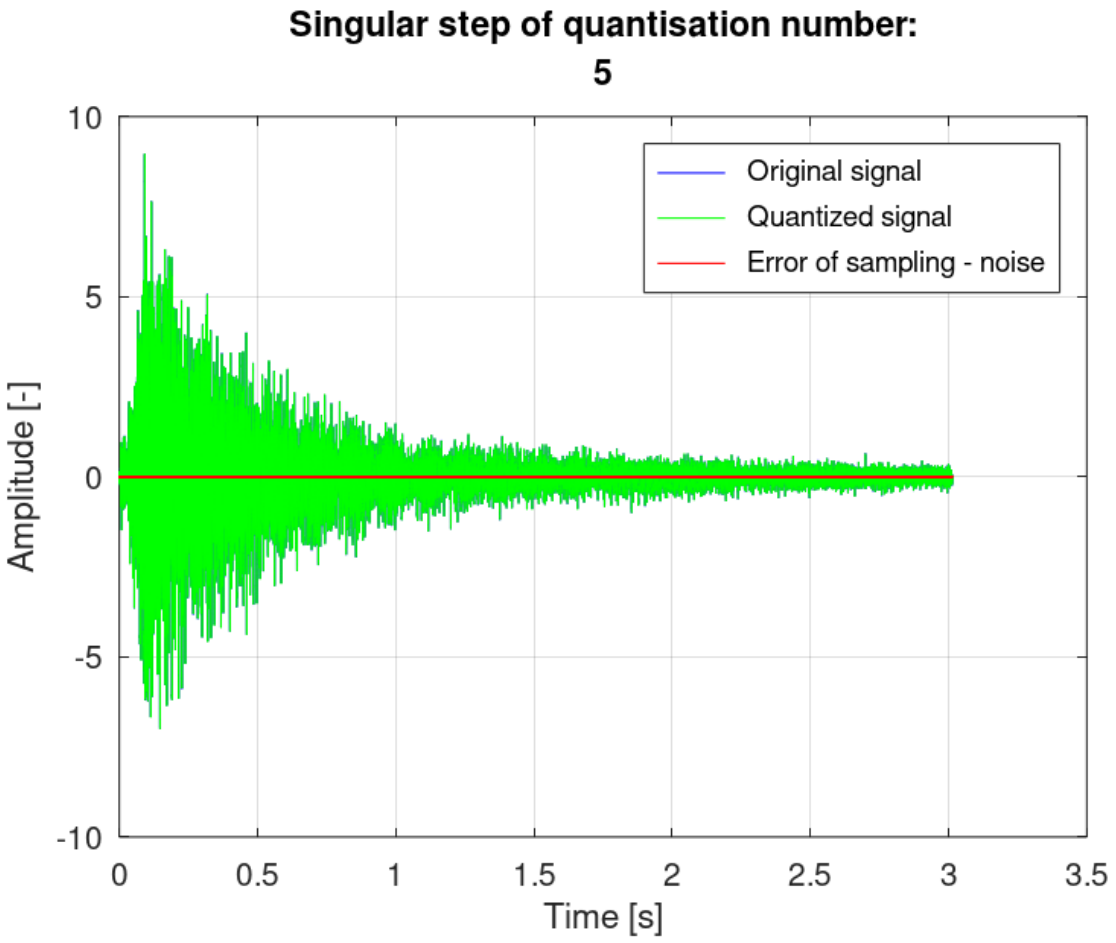
E Chord

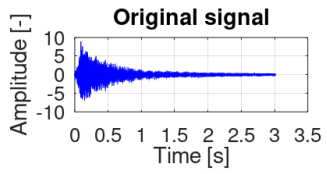

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

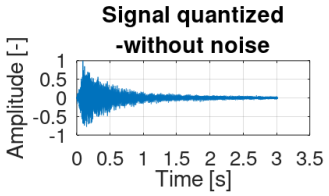

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

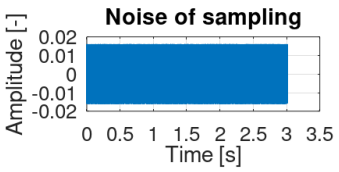

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

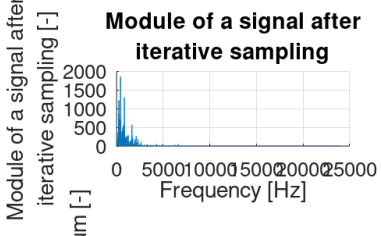

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

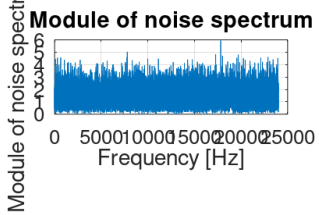

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

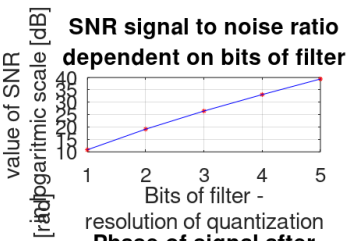

Original signal
Amplitude [-]
Time [s]

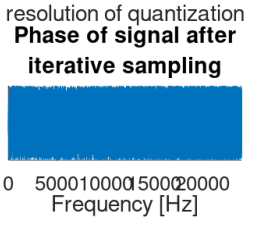

Signal quantized
-without noise
Amplitude [-]
Time [s]

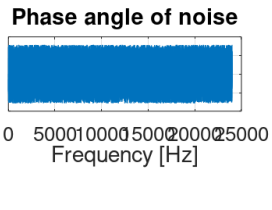

Noise of sampling
Amplitude [-]
Time [s]

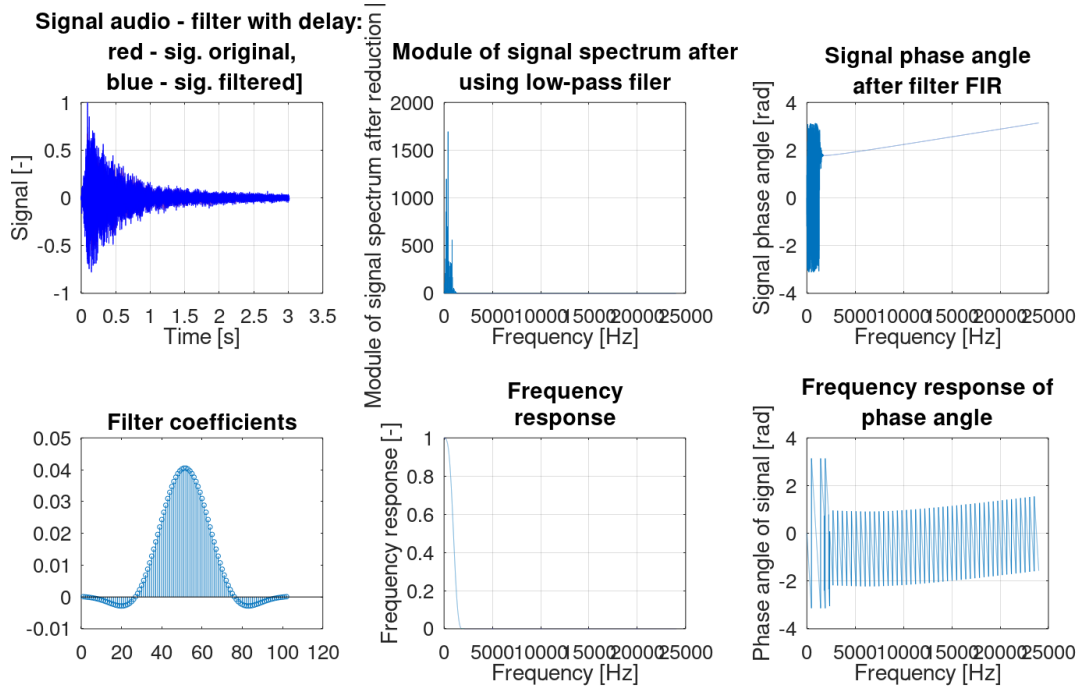

Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

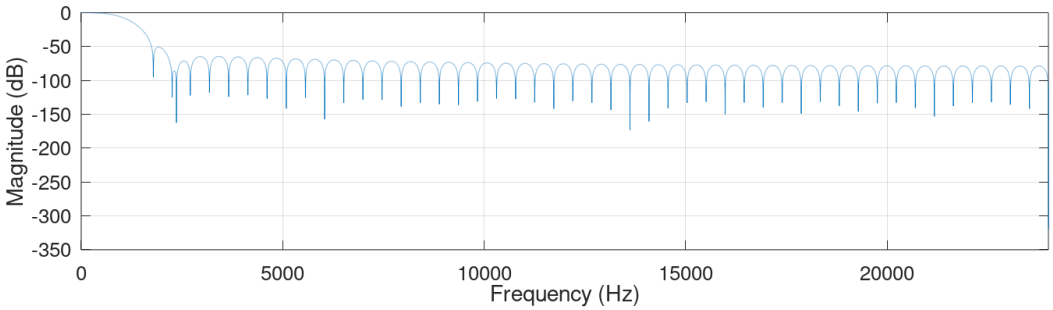

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

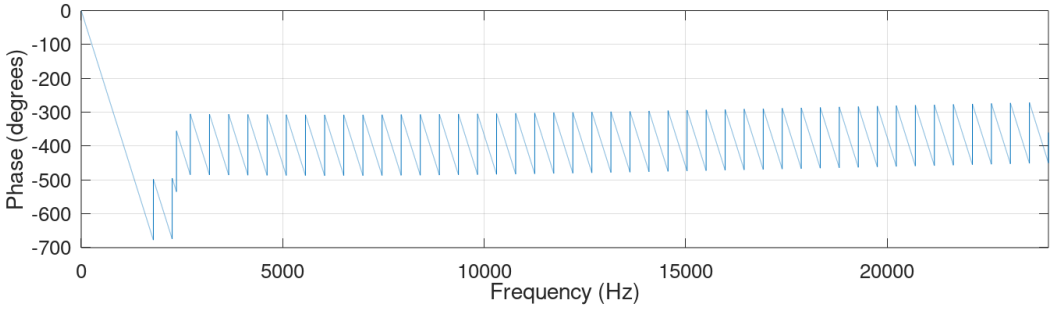

SNR signal to noise ratio
dependent on bits of filter
value of SNR
logaritmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]

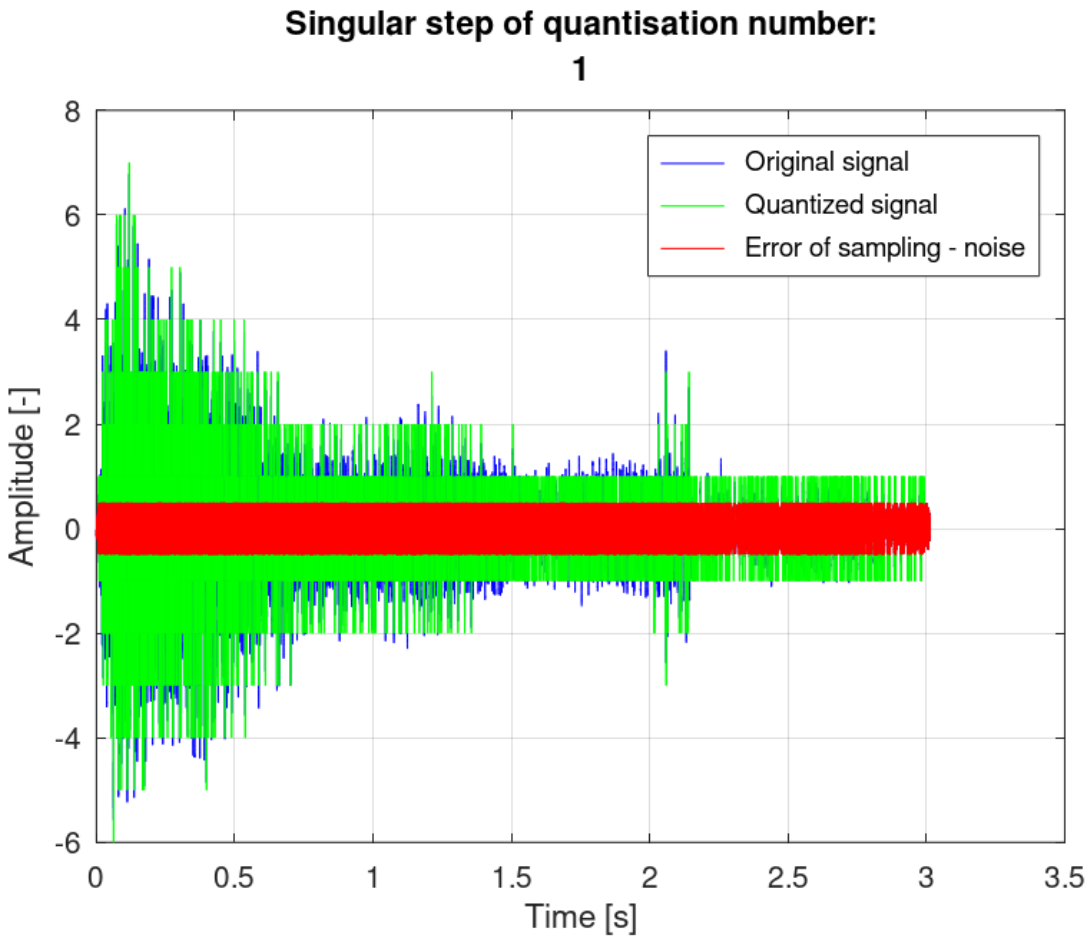

Phase angle of noise
Frequency [Hz]

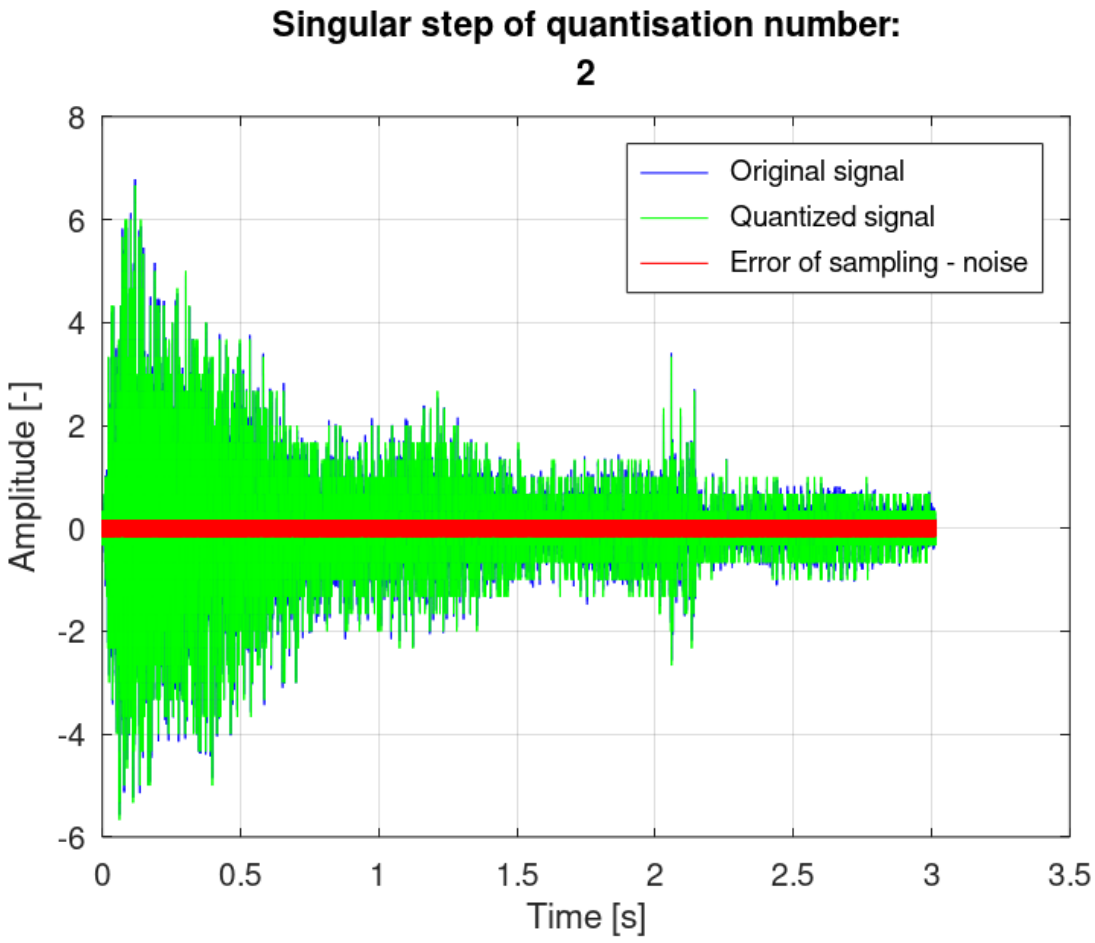

Signal audio - filter with delay:
red - sig. original,
blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after reduction
using low-pass filer
Module of signal spectrum after reduction |
Frequency [Hz]
Signal phase angle
after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency
response
Frequency response [-]
Frequency [Hz]
Frequency response of
phase angle
Phase angle of signal [rad]
Frequency [Hz]

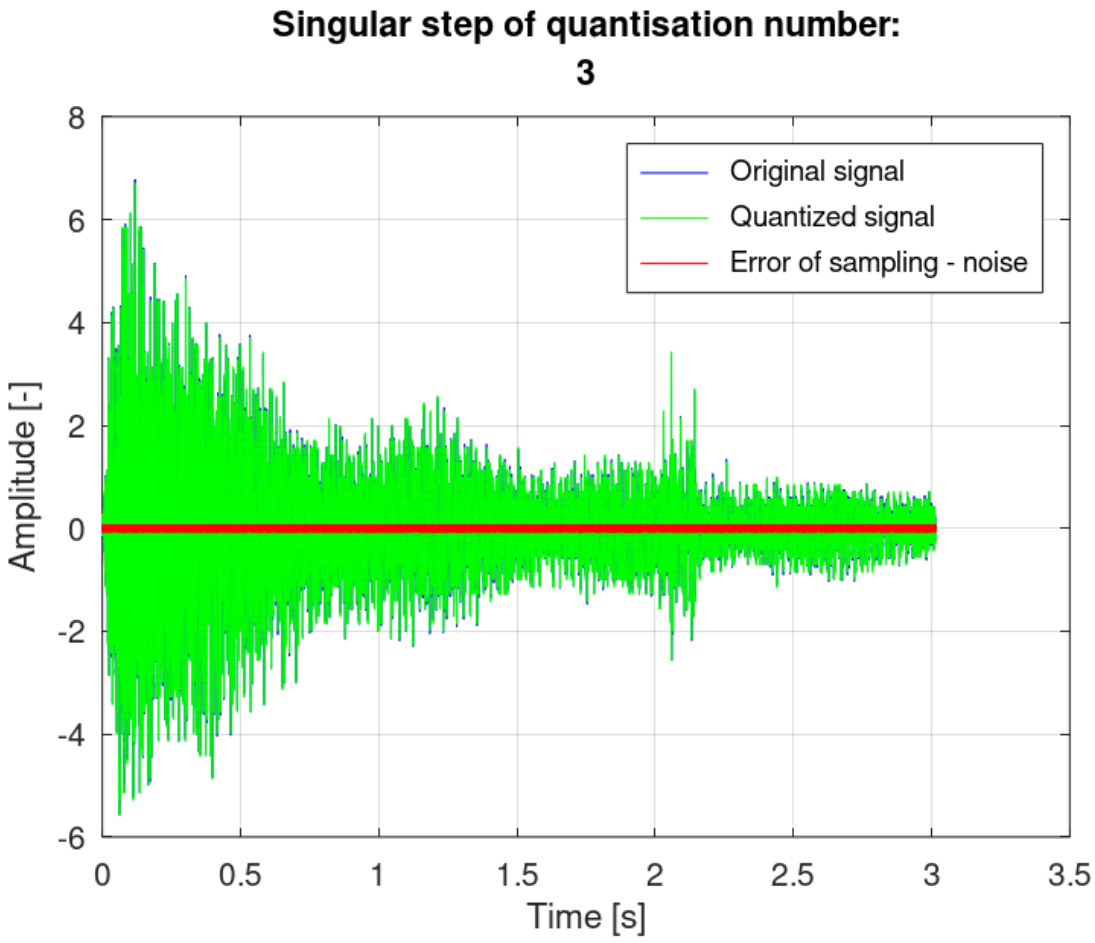

Magnitude (dB)
Frequency (Hz)

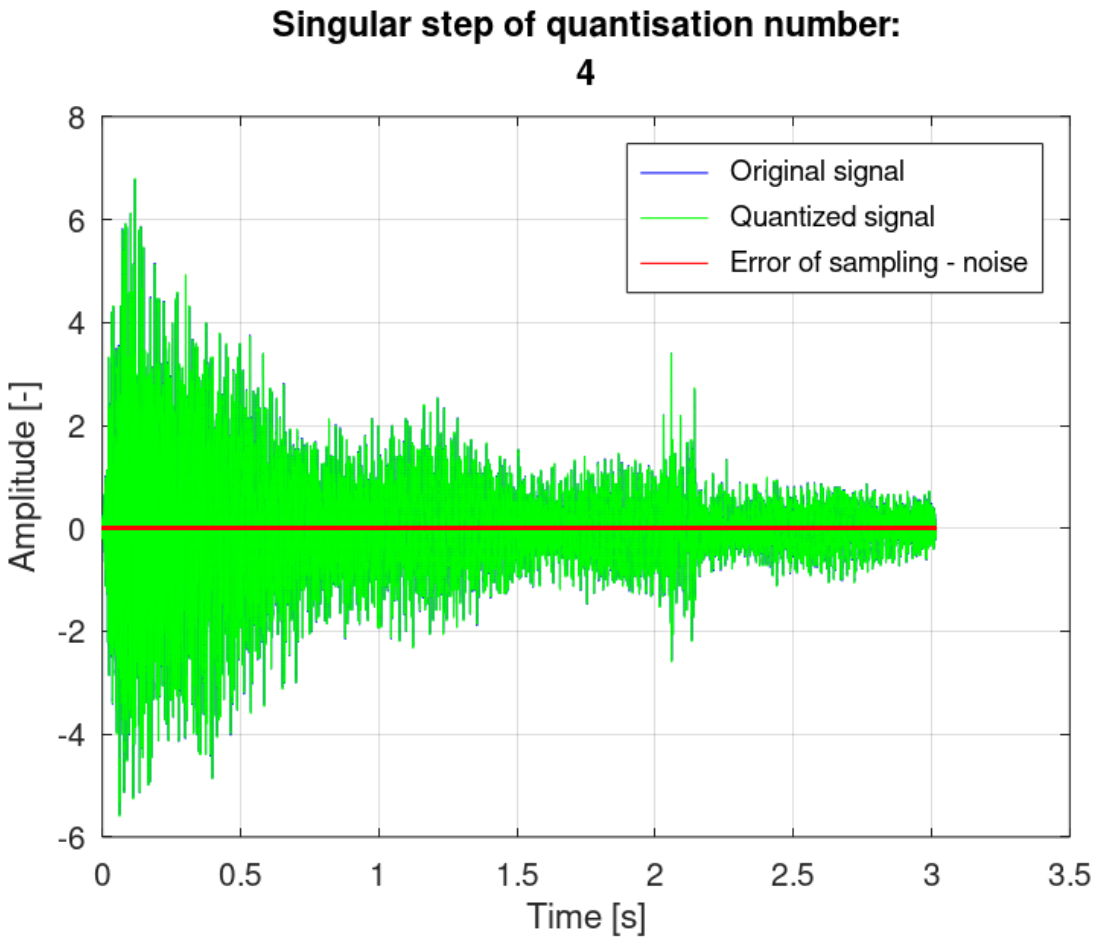

Phase (degrees)
Frequency (Hz)

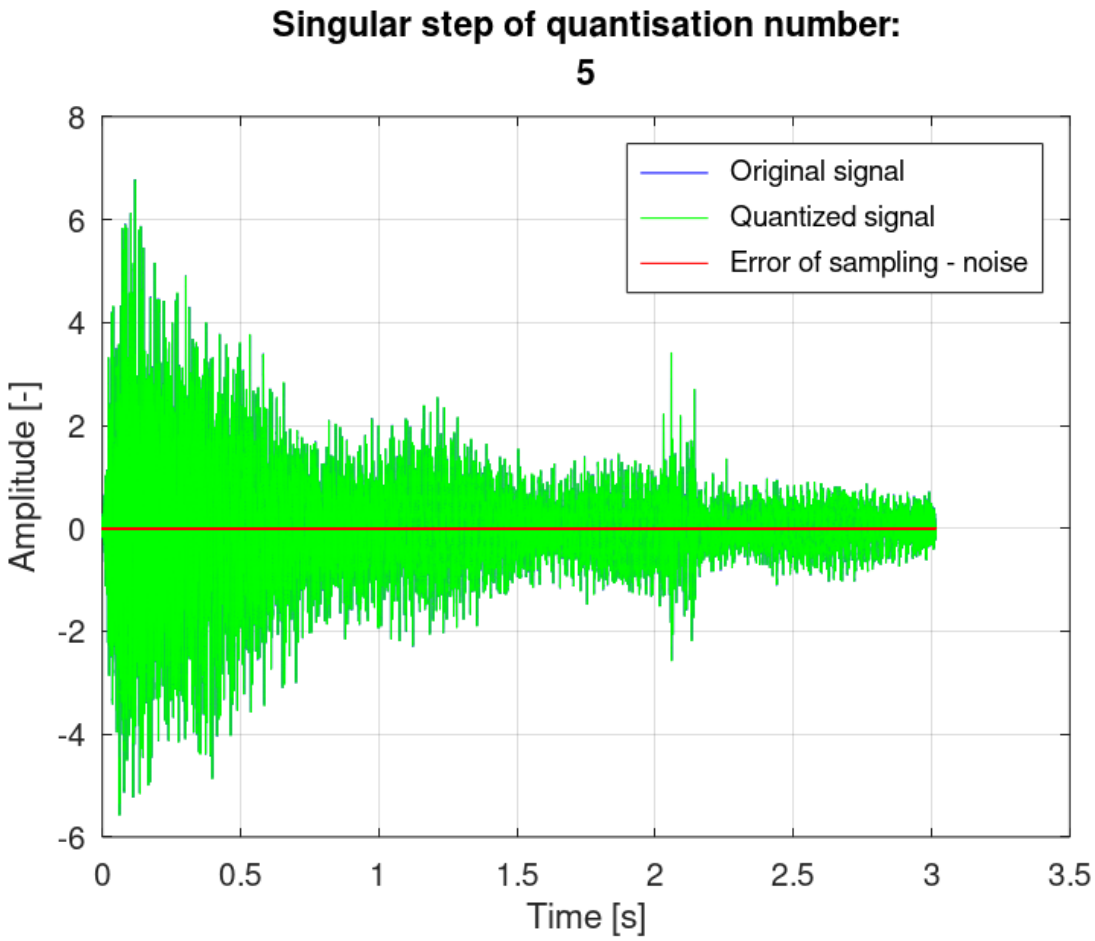
Em Chord

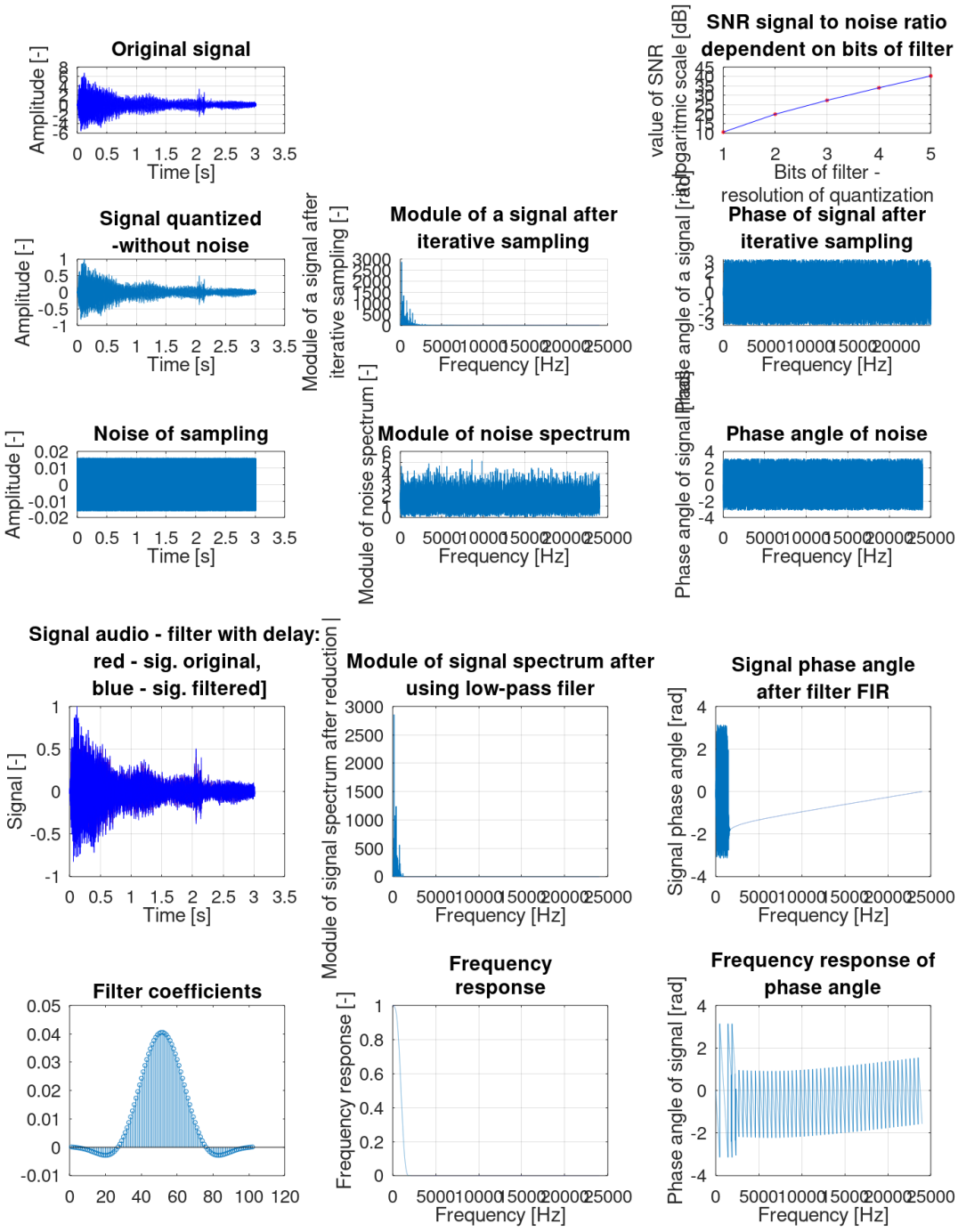

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
8
6
4
2
0
-2
-4
-6
0
0.5
1
1.5
2
2.5
3
3.5

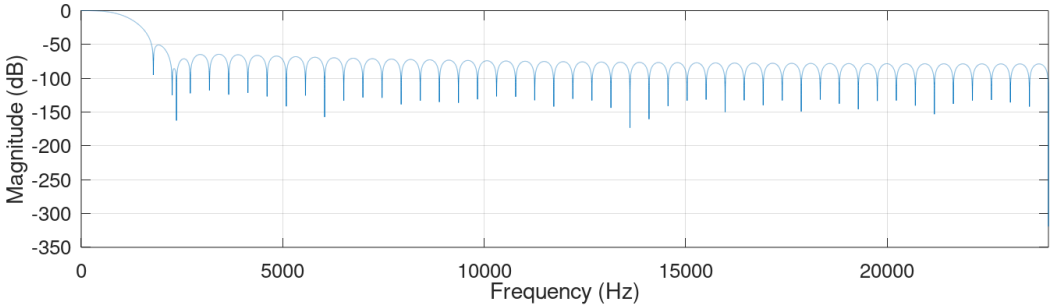

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
8
6
4
2
0
-2
-4
-6
0
0.5
1
1.5
2
2.5
3
3.5

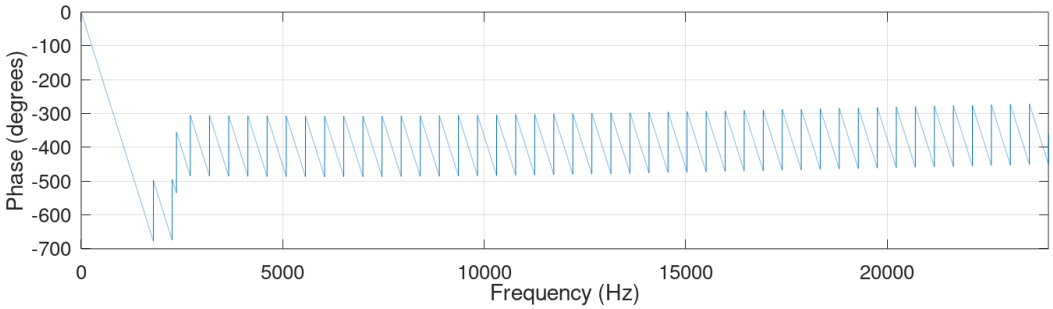

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
8
6
4
2
0
-2
-4
-6
0
0.5
1
1.5
2
2.5
3
3.5


Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
8
6
4
2
0
-2
-4
-6
0
0.5
1
1.5
2
2.5
3
3.5

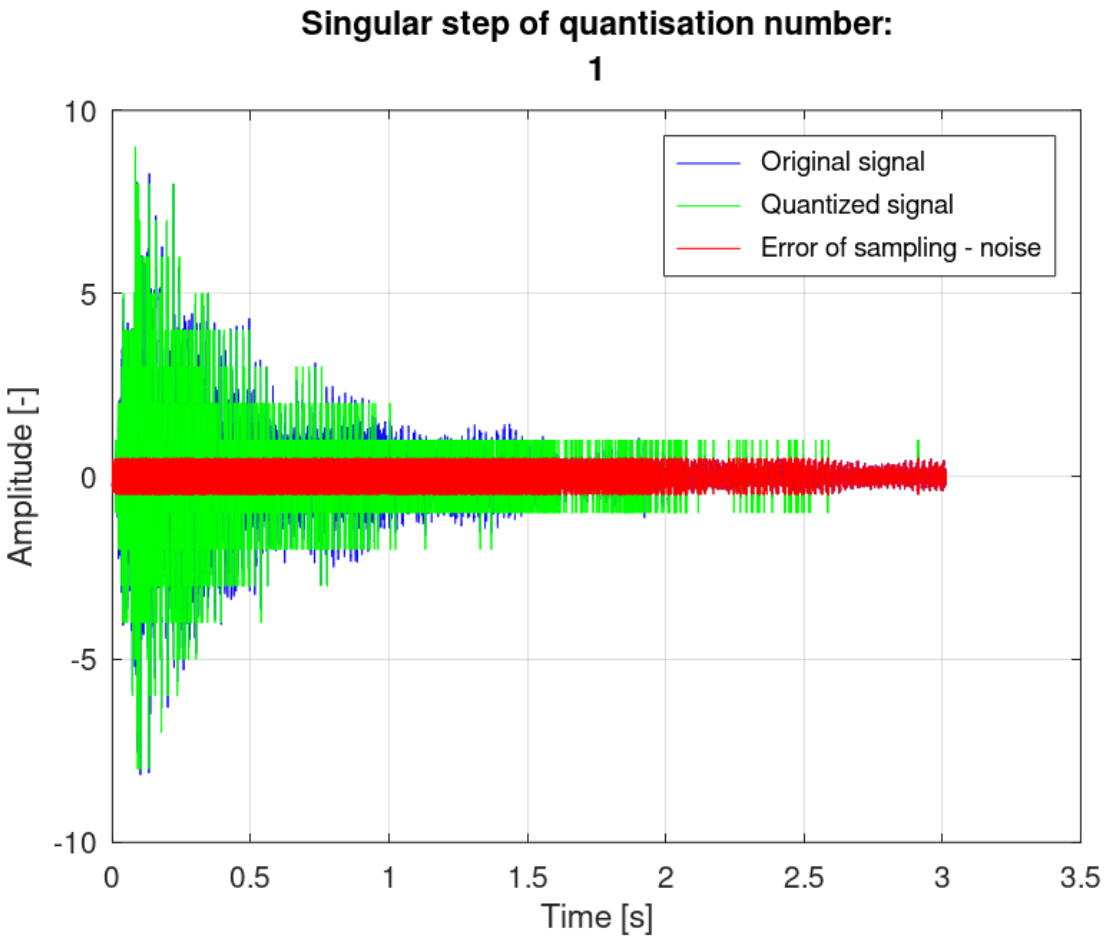

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
8
6
4
2
0
-2
-4
-6
0
0.5
1
1.5
2
2.5
3
3.5

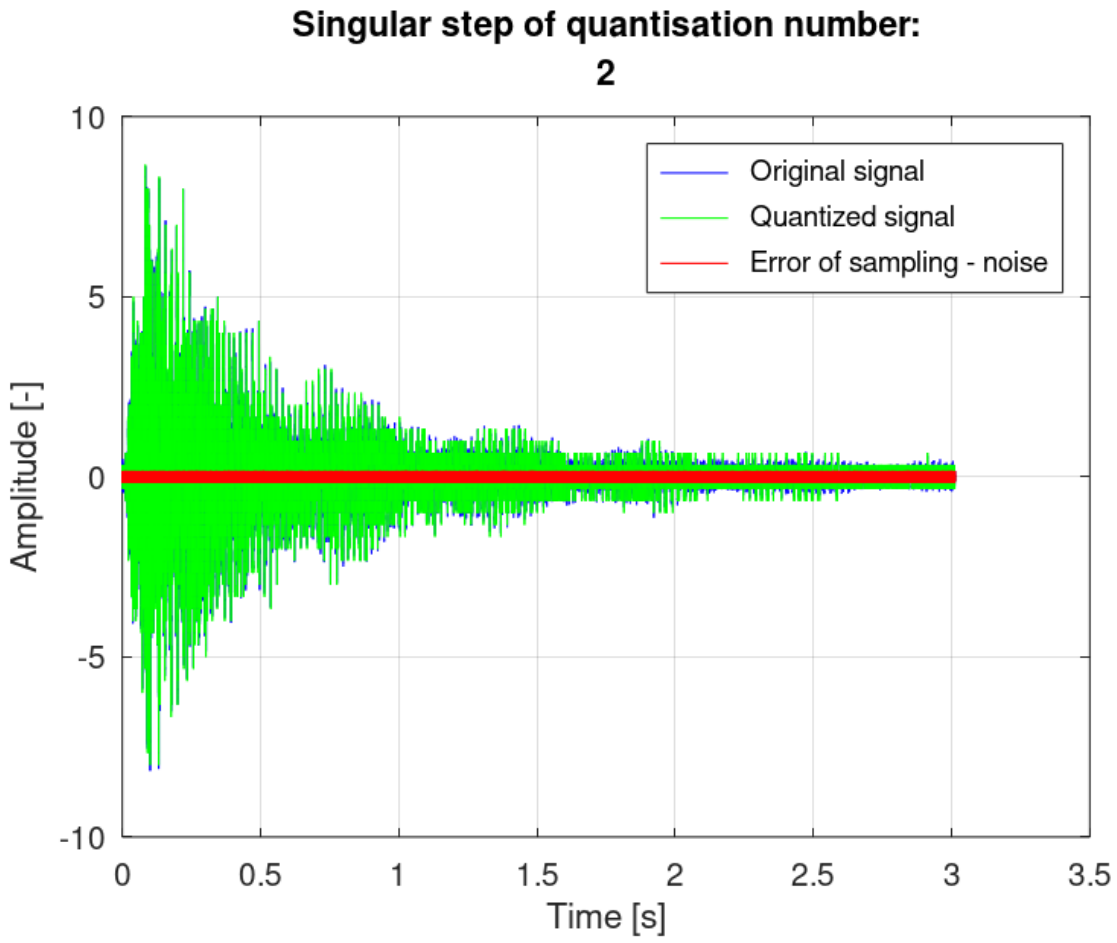

Original signal
Amplitude [-]
Time [s]
SNR signal to noise ratio
dependent on bits of filter
value of SNR
logaritmic scale [dB]
Bits of filter -
resolution of quantization
Signal quantized
-without noise
Amplitude [-]
Time [s]
Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]
Phase of signal after
iterative sampling
Phase angle of a signal [rad]
Frequency [Hz]
Noise of sampling
Amplitude [-]
Time [s]
Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]
Phase angle of noise
Phase angle of signal [rad]
Frequency [Hz]
Signal audio - filter with delay:
red - sig. original,
blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after reduction
Module of signal spectrum after
using low-pass filer
Frequency [Hz]
Signal phase angle
after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency
response
Frequency response [-]
Frequency [Hz]
Frequency response of
phase angle
Phase angle of signal [rad]
Frequency [Hz]

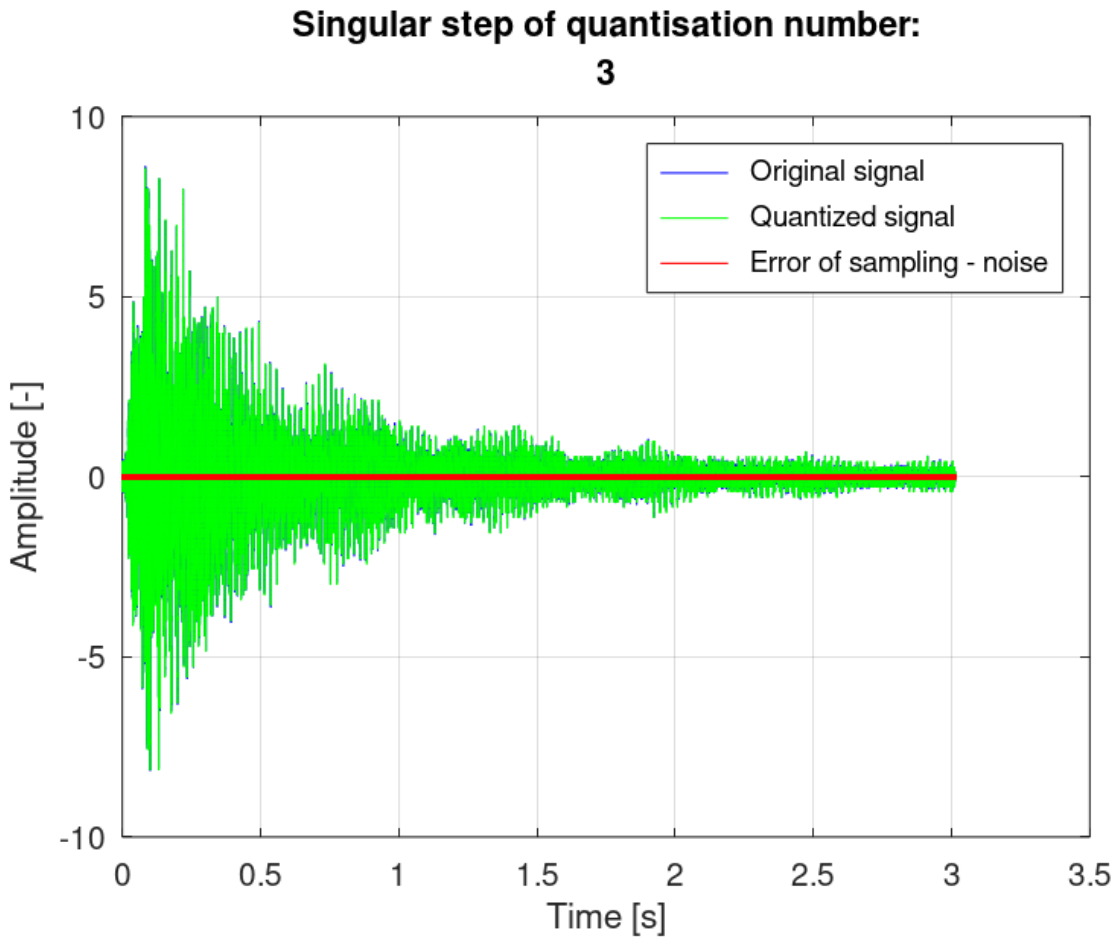

Magnitude (dB)
0
-50
-100
-150
-200
-250
-300
-350
0
5000
10000
15000
20000
Frequency (Hz)

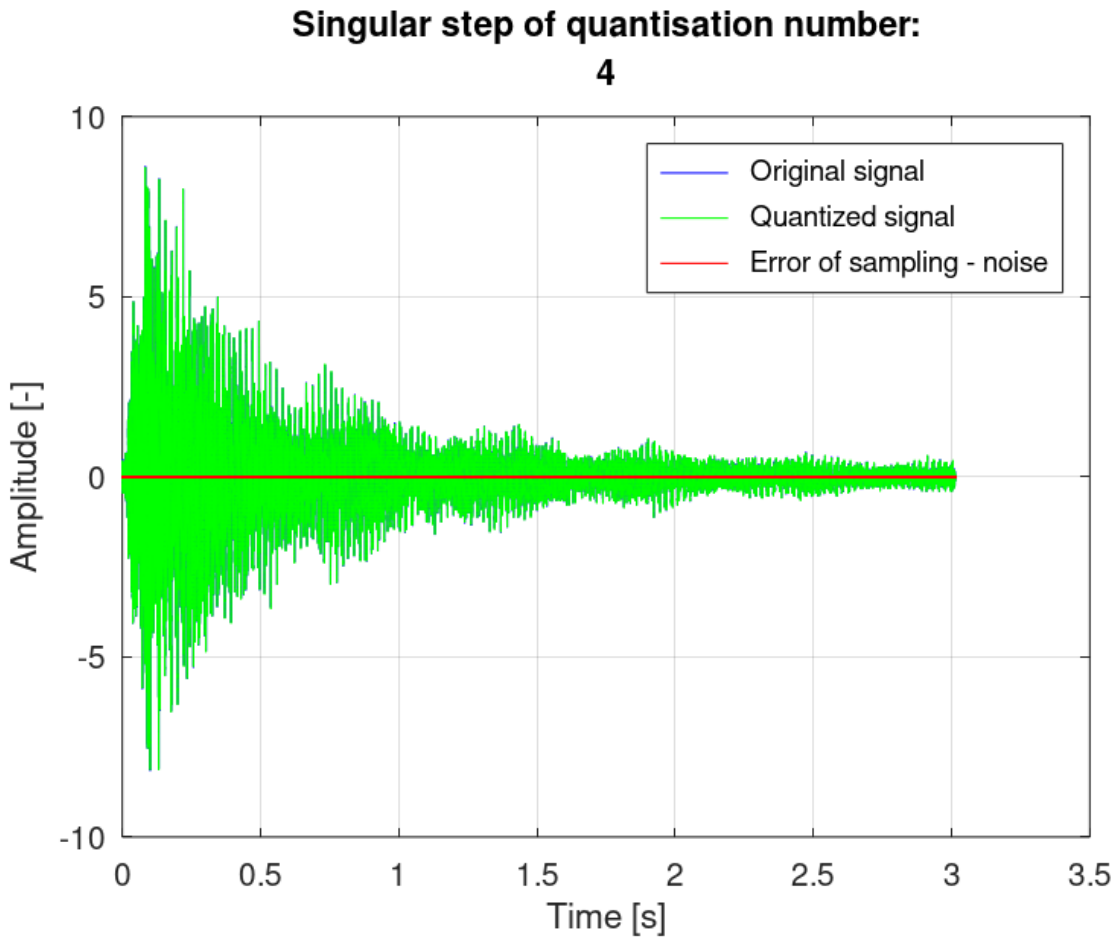

Phase (degrees)
0
-100
-200
-300
-400
-500
-600
-700
0
5000
10000
15000
20000
Frequency (Hz)

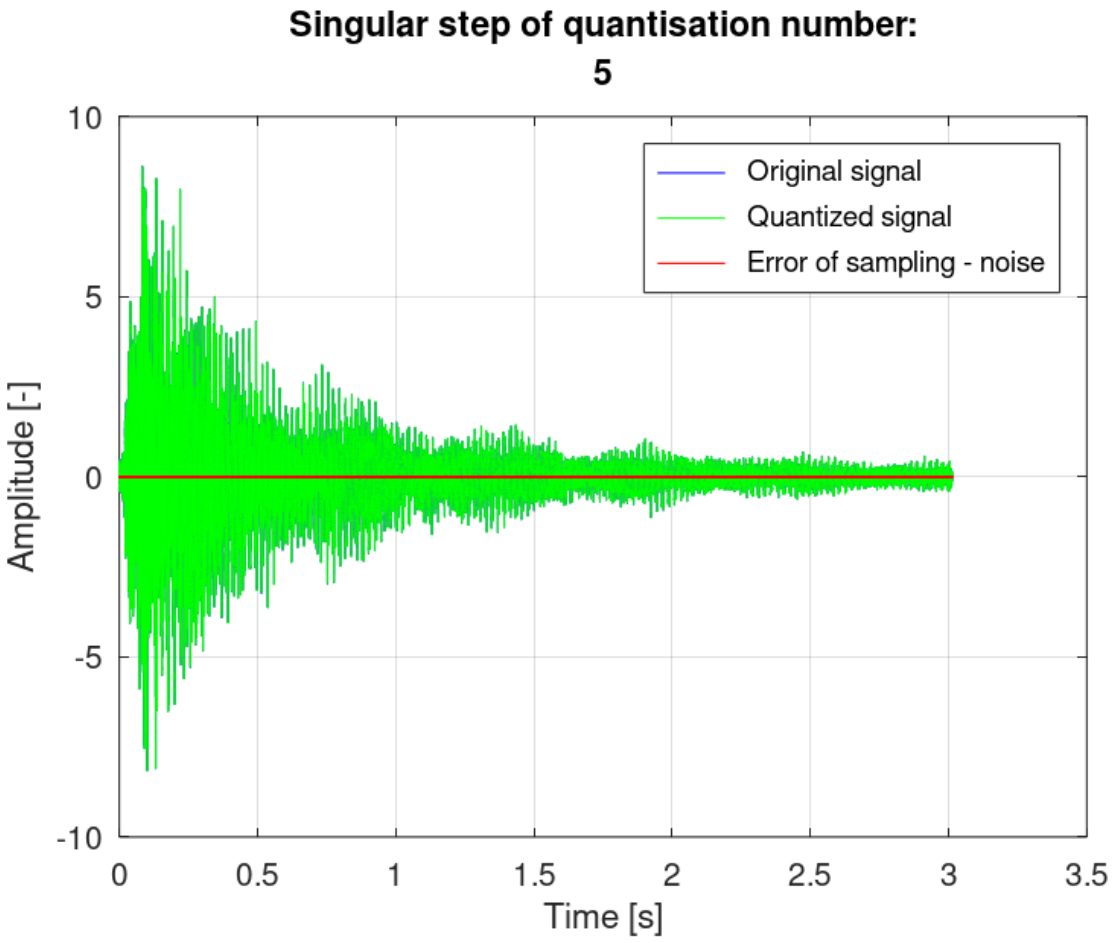
F Chord

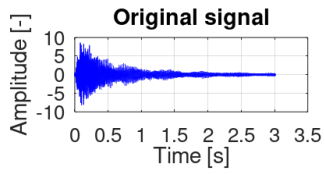

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

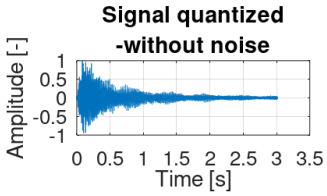

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

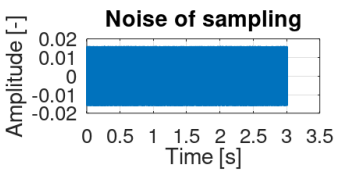

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

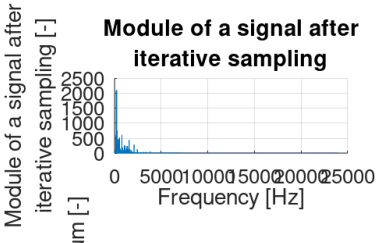

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

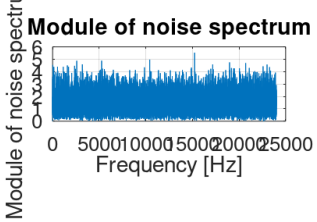

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

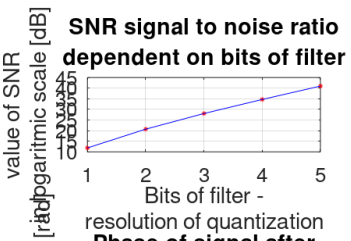

Original signal
Amplitude [-]
Time [s]

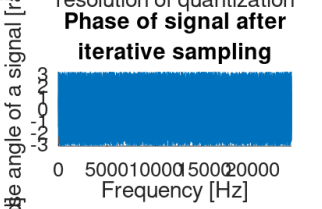

Signal quantized
-without noise
Amplitude [-]
Time [s]

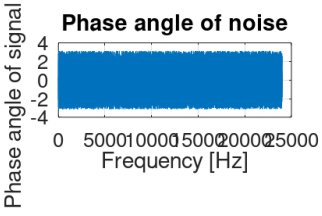

Noise of sampling
Amplitude [-]
Time [s]

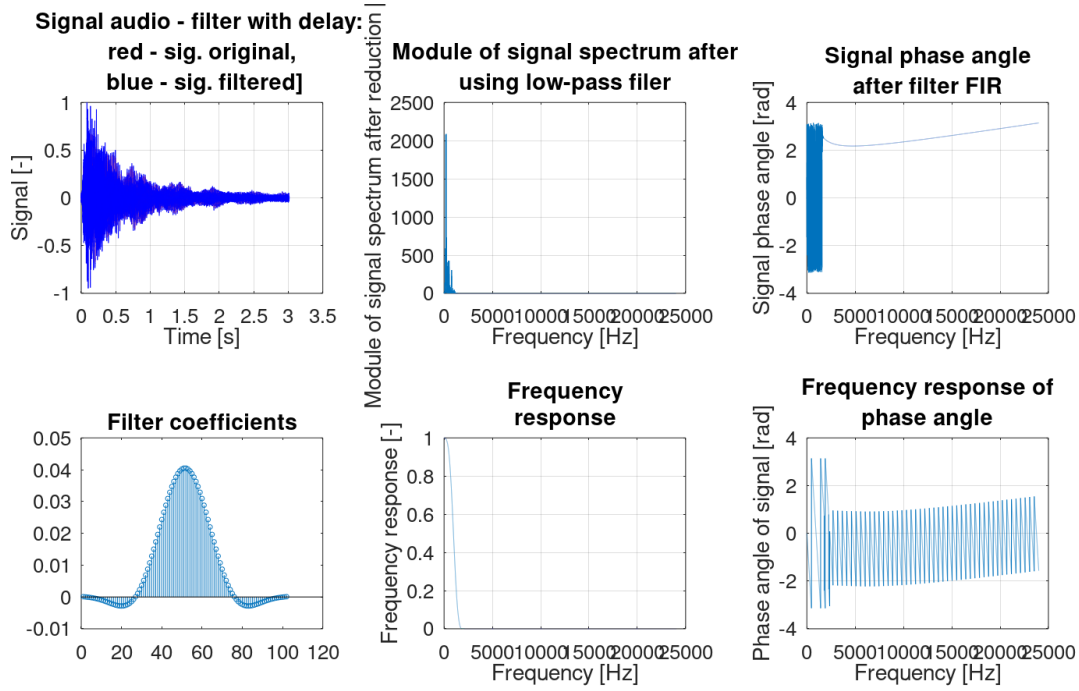

Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

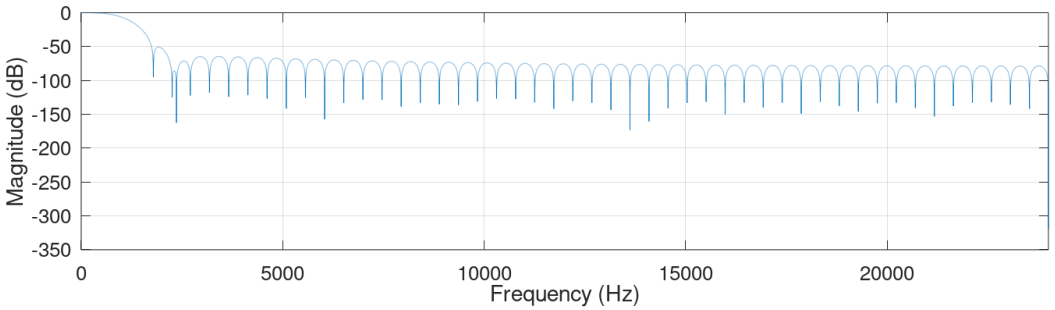

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

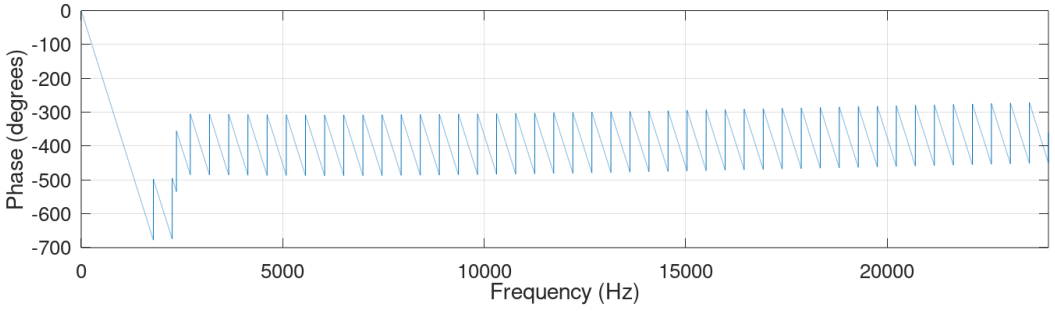

SNR signal to noise ratio
dependent on bits of filter
value of SNR
logaritmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]

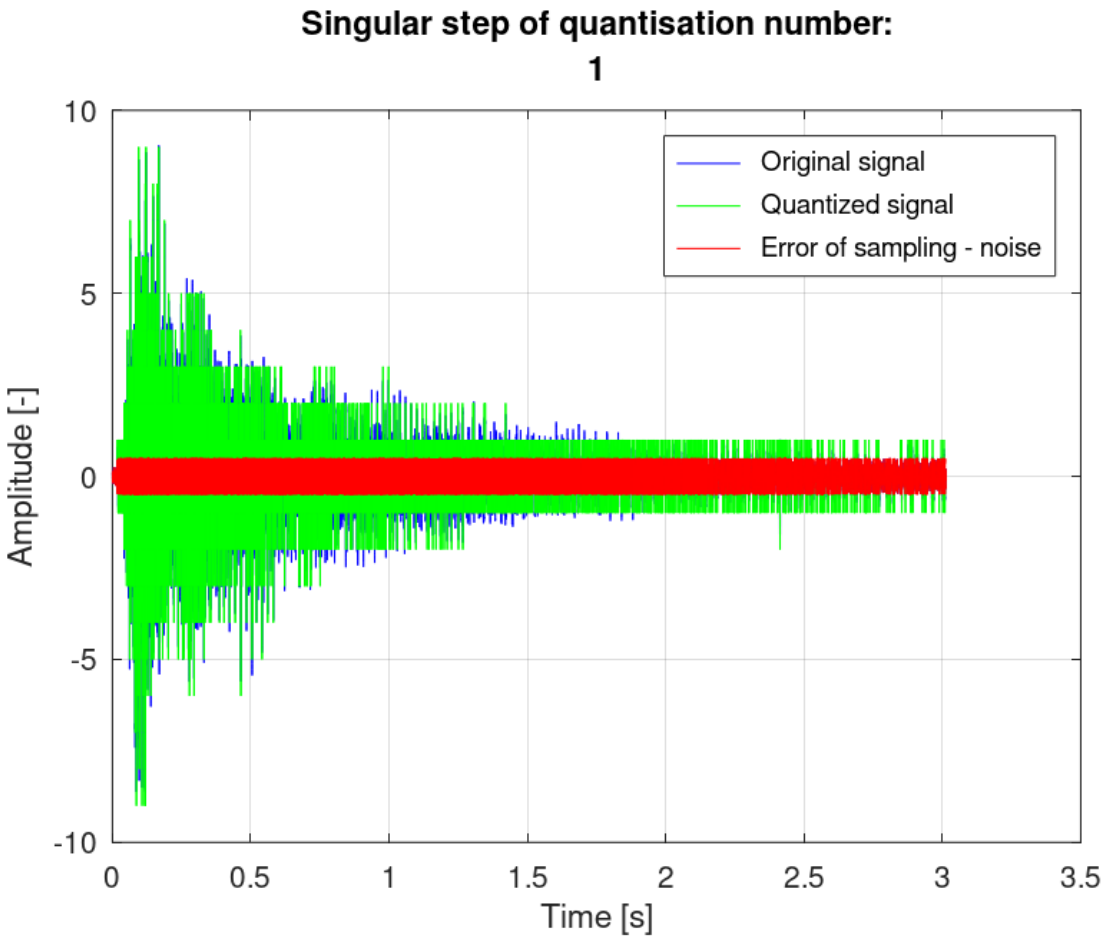

Phase angle of noise
Frequency [Hz]

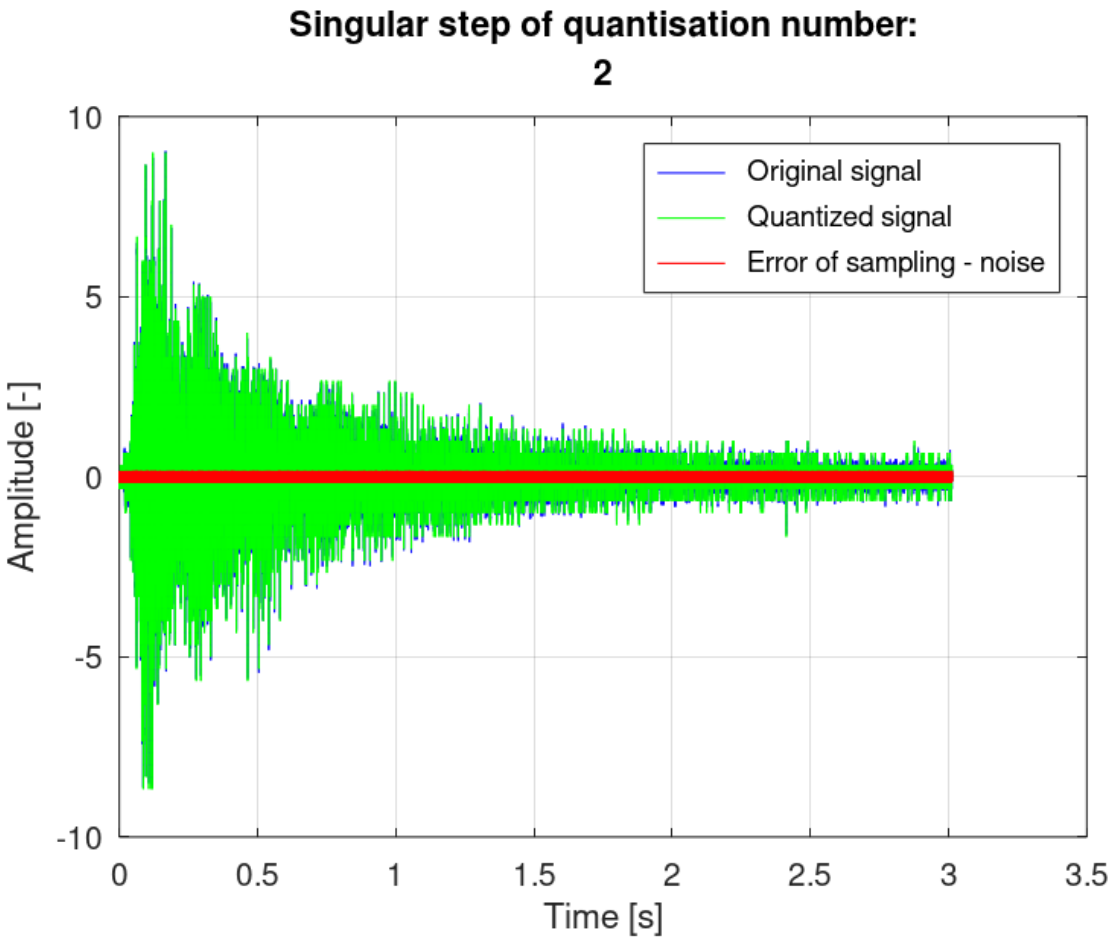

Signal audio - filter with delay:
red - sig. original,
blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after using low-pass filer
Module of signal spectrum after reduction
Frequency [Hz]
Signal phase angle after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency response
Frequency response [-]
Frequency [Hz]
Frequency response of phase angle
Phase angle of signal [rad]
Frequency [Hz]

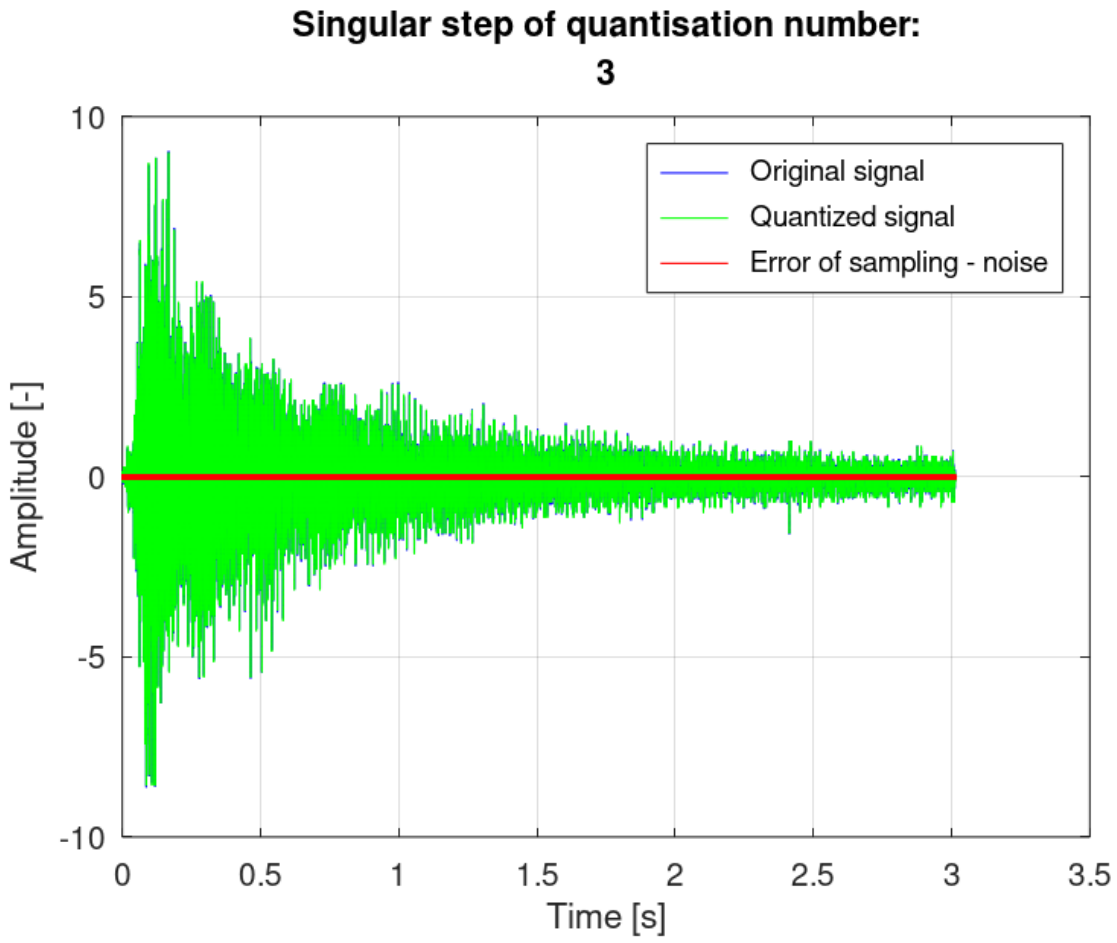

Magnitude (dB)
Frequency (Hz)

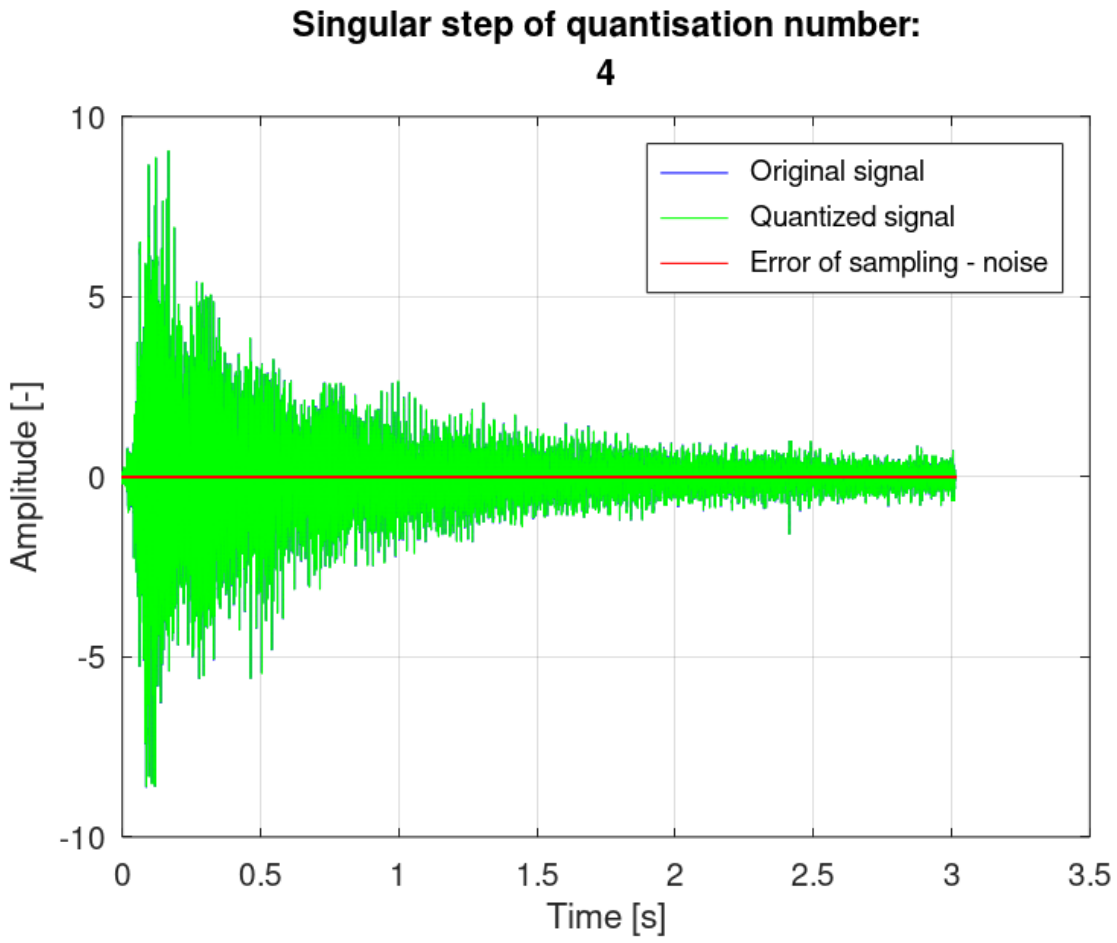

Phase (degrees)
Frequency (Hz)

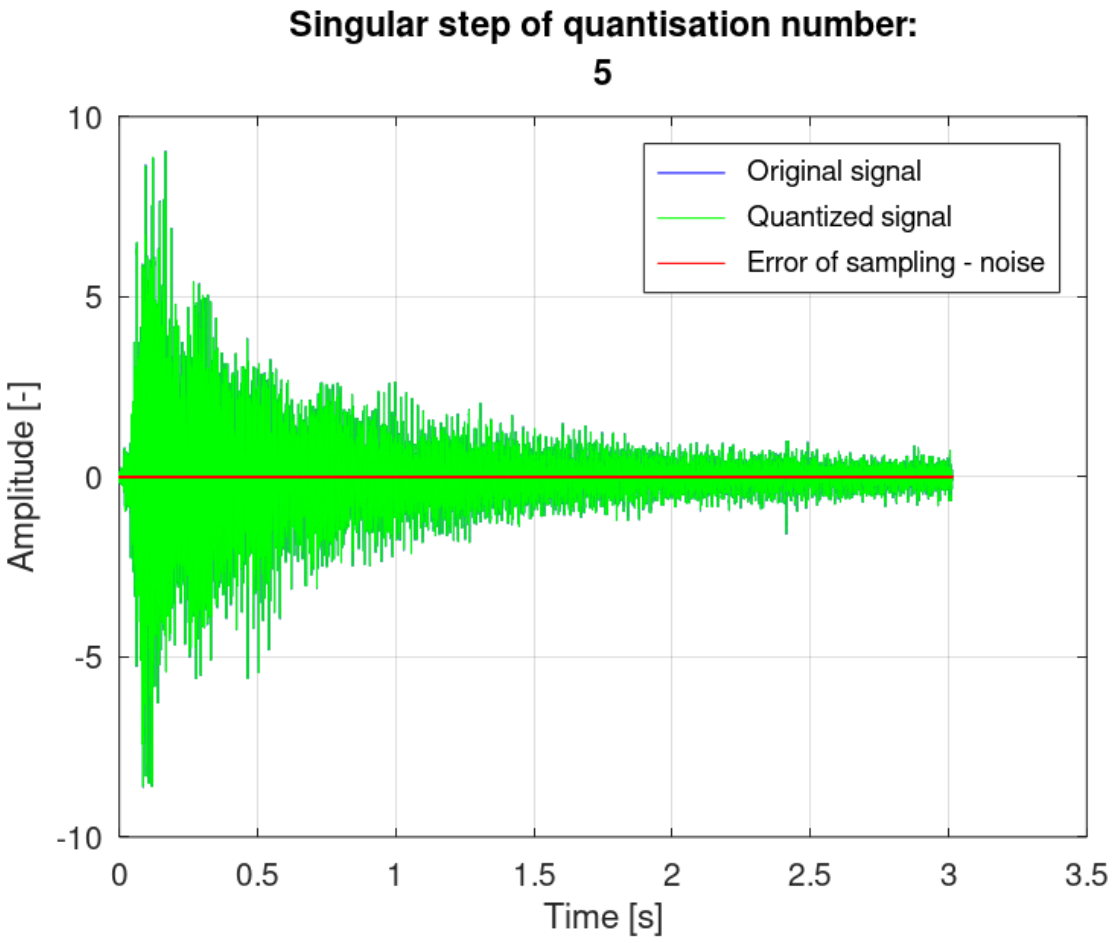
G Chord

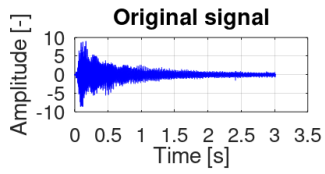

Singular step of quantisation number:
1
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

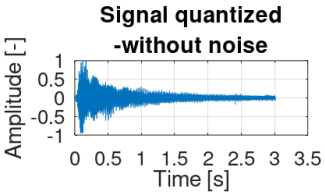

Singular step of quantisation number:
2
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

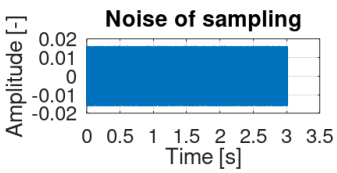

Singular step of quantisation number:
3
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

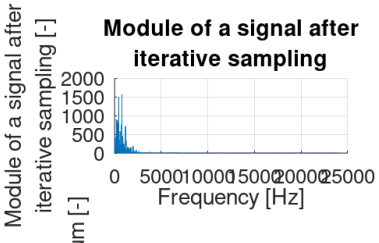

Singular step of quantisation number:
4
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5

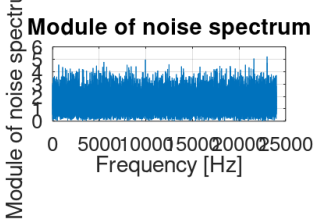

Singular step of quantisation number:
5
Original signal
Quantized signal
Error of sampling - noise
Amplitude [-]
Time [s]

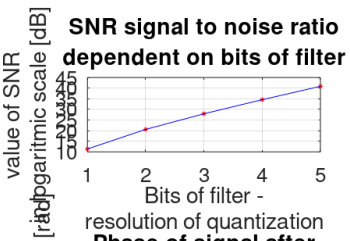

Original signal
Amplitude [-]
Time [s]

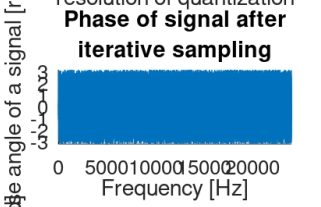

Signal quantized
-without noise
Amplitude [-]
Time [s]

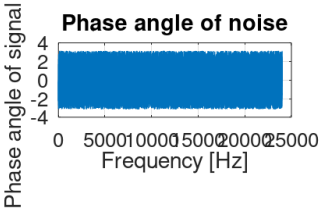

Noise of sampling
Amplitude [-]
Time [s]

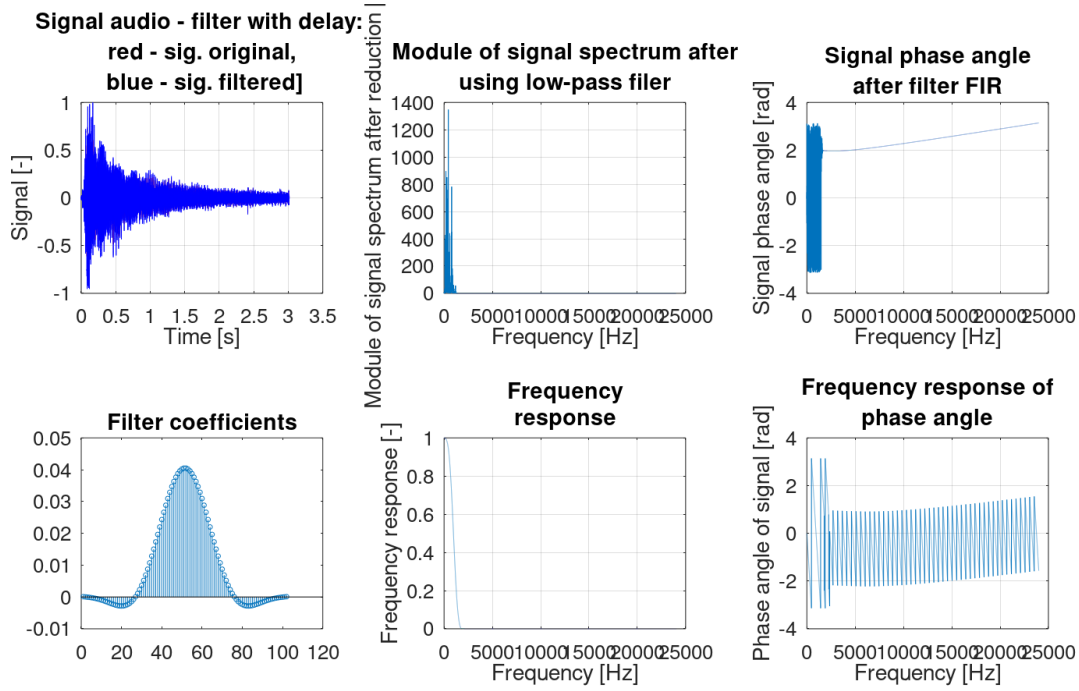

Module of a signal after
iterative sampling
Module of a signal after iterative sampling [-]
Frequency [Hz]

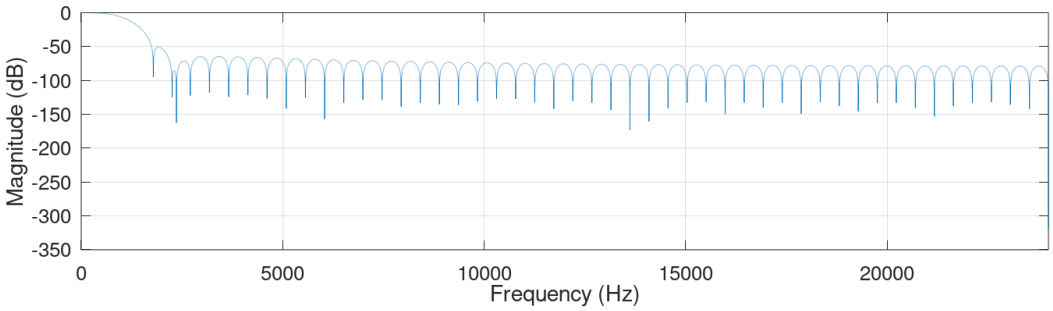

Module of noise spectrum
Module of noise spectrum [-]
Frequency [Hz]

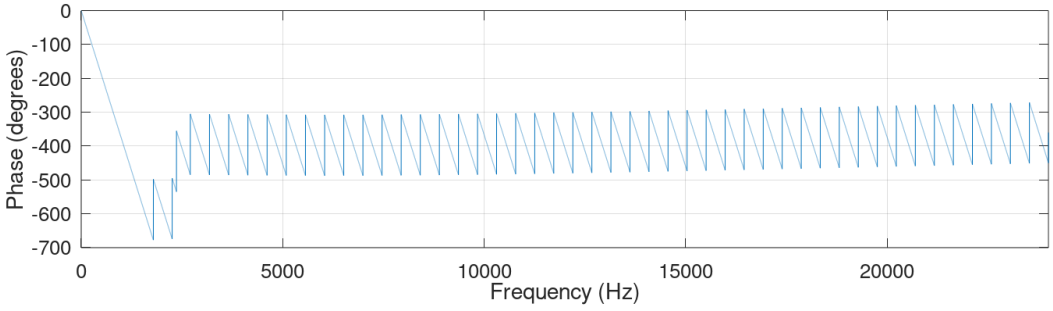

SNR signal to noise ratio
dependent on bits of filter
value of SNR logaritmic scale [dB]
Bits of filter -
resolution of quantization


Phase of signal after
iterative sampling
Frequency [Hz]


Phase angle of noise
Frequency [Hz]


Signal audio - filter with delay:
red - sig. original,
blue - sig. filtered]
Signal [-]
Time [s]
Module of signal spectrum after using low-pass filer
Module of signal spectrum after reduction
Frequency [Hz]
Signal phase angle after filter FIR
Signal phase angle [rad]
Frequency [Hz]
Filter coefficients
Frequency response
Frequency response [-]
Frequency [Hz]
Frequency response of phase angle
Phase angle of signal [rad]
Frequency [Hz]


Magnitude (dB)
Frequency (Hz)


Phase (degrees)
Frequency (Hz)

Appendix C
Researcher Curriculum Vitae

BRIX IAN RELANO

2077-B Smith St. Malate Manila
+639052868450
bidrelano2018@plm.edu.ph
https:/www.linkedin.com/in/brix-relano-910057184
https://www.facebook.com/brixsanityy/

EDUCATION

BS Computer Engineering | Pamantasan ng Lungsod ng Maynila
2018 – 2022
Officially a 4th year Computer Engineering Student at Pamantasan ng Lungsod ng Maynila
Won 3rd place at UltiMATHum by PLM Engineering Society (January 2020)

STEM | Manila Science High School
2016 – 2018
Graduated with Honors

Manila Science High School (Junior High)
2012 – 2016
Graduated with Honors

EXPERIENCE

Auditor | ICpEP.se – PLM Chapter
2019 – 2020
Handled auditing processes inside ICpEP.se – PLM Organization, specifically its inventory and handling of the receipts.

Vice President | PLM Red Cross Youth Council – CET College Board Unit
2019 – 2020
Fulfilled the role of Vice-President in PLM RCYC – CET CBU

Treasurer | ICpEP.se – PLM Chapter
2020 – 2021
Handled main financial processes inside ICpEP.se – PLM Organization, specifically its Financial Statements and Receipts.

President | Haribons E-Sports
2020 – PRESENT
Established a campus esports organization inside PLM to ignite students having passion for esports.

SKILLS

- Backend coding
- Frontend coding
- C++, C, Java, Python Languages
- Research/Thesis Making
- Esports Management

ACTIVITIES

Ateneo Coding Summer Camp | Ateneo De Manila University
2014– 2015
Joined Coding Summer Camp in Ateneo with the topic of learning C Language.

DLSU Coding Summer Camp | De La Salle University
2015 – 2016
Joined Coding Summer Camp in La Salle with the topic of learning C++ Language

Research Symposium | Manila Science High School
MARCH 2017
Presented our Qualitative Research to various panels in Manila Science High School

Research Symposium | Ramon Avanceña Junior High School
MARCH 2017
Participated in a Research Competition held at Ramon Avanceña Junior High School

Global Game Jam | VR Philippines
JANUARY 2019
Participated in a 3-day simultaneous global hackathon with a goal of developing a game given an impromptu topic and a team.

CONTACT INFORMATION

09274961412

rpmreyes2018@plm.edu.ph

facebook.com/ravidphelps

www.linkedin.com/in/ravidphelps

SKILLS

Professional

- Good Communicator
- Fast Learner
- Team Player
- Motivated

Technical

- MS Office
- HTML
- C#
- Python
- OpenCV
- Cisco Networking

RAVID PHELPS M. REYES

PROFILE

I AM A COMPUTER ENGINEERING STUDENT FROM PAMANTASAN NG LUNGSOD NG MAYNILA. I HAVE BEEN SEEKING NEW FIELDS AND EXPERIENCES IN COMPUTER NETWORKS, WORKING MY WAY TO EARNING A CCNA CERTIFICATION. ARTIFICIAL INTELLIGENCE AND CLOUD COMPUTING ARE ALSO FIELDS THAT I AM INTERESTED IN. I BELIEVE IN LEARNING THROUGH COLLABORATION.

EDUCATION

ELEMENTARY
LA CONSOLACION COLLEGE DEPARO
(2006-2012)
- SALUTATORIAN

JUNIOR HIGH SCHOOL
LA CONSOLACION COLLEGE DEPARO
(2012-2016)
- COMPLETED WITH 2ND HONORS

SENIOR HIGH SCHOOL
TECHNOLOGICAL UNIVERSITY OF THE PHILIPPINES
(2016-2018)
- COMPLETED SENIOR HIGH SCHOOL

COLLEGE
PAMANTASAN NG LUNGSOD NG MAYNILA
(2018-PRESENT)
- CURRENTLY TAKING BACHELOR OF SCIENCE IN COMPUTER ENGINEERING (BS CPE)

CONTACT INFORMATION

09260926526

ajasy2018@plm.edu.ph

facebook.com/ aaron.sy.752

www.linkedin.com/in/aaron-jacob-sy

SKILLS

Professional

- Written and Verbal Communication
- Problem Solving
- Openness to Feedback
- Teamwork
- Time management
- Initiative

Technical

- MS Office Applications
- HTML
- CSS
- C#
- Python

AARON JACOB A. SY

PROFILE

I AM A HARD-WORKING STUDENT WITH A GOOD ACADEMIC RECORD. A GOAL DRIVEN INDIVIDUAL WHO IS WILLING TO LEARN DIFFERENT SKILLS IN SOFTWARE DEVELOPMENT AND COMPUTER NETWORKS.

EDUCATION

ELEMENTARY
ST. MARY'S ACADEMY – PASAY CITY
(2006-2012)

- WITH HONORS

JUNIOR HIGH SCHOOL
PASAY CITY NATIONAL SCIENCE HIGH SCHOOL
(2012-2016)

- WITH HONORS

SENIOR HIGH SCHOOL
PASAY CITY NATIONAL SCIENCE HIGH SCHOOL
(2016-2018)

- WITH HONORS

COLLEGE
PAMANTASAN NG LUNGSOD NG MAYNILA
(2018-PRESENT)

- CURRENTLY TAKING BACHELOR OF SCIENCE IN COMPUTER ENGINEERING (BS CPE)