INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY BANGALORE

SUBJECT

SIGNAL PROCESSING

Project: Audio Signal Processing

B.Sashank Reddy ,Ayyappa Koppuravuri
(IMT2020542,IMT2020555)

April 17, 2022

ज्ञानमुत्तमम्

Introduction:

In this project we are going to record two signals and analyse(Reading the info and converting them to .wav files) those signals and we are going generate new signals with those such that both the signals are played at a time in two different methods.Also we separate the combined signal into two signals with two methods.They are ICA(Independent Component Analysis) and Channeling

Data set

We generated the audio using a laptop and recorded the audio signals through voice recorder application of mobile phones.

ICA

Input signals

paths - 'ICA baby elephant 60.mp3' & 'ICA pink_panther.mp3'

.mp3 to .wav converted signals

paths - 'ICA wav_inputs input_Elephant.wav' & 'ICA wav_inputs input_pink_panther.wav'

input1 channels

paths - 'ICA input1_cnls test1.wav' & 'ICA input1_cnls test2.wav'

input2 channels

paths - 'ICA input2_cnls test1_1.wav' & 'ICA input2_cnls test2_1.wav'

Mixed signals

paths - 'ICA ica_mix mix1.wav' & 'ICA ica_mix mix2.wav'

Seperated signals

paths - 'ICA unmix unmix1.wav' & 'ICA unmix unmix2.wav'

Channeling

Input signals

paths - 'Channeling baby elephant 60.mp3' & 'Channeling pink_panther.mp3'

.mp3 to .wav converted signals

paths - 'Channeling wav_inputs input_Elephant.wav' & 'Channeling wav_inputs input_pink_panther.wav'

input1 channels

paths - 'Channeling input1_cnls test1.wav' & 'Channeling input1_cnls test2.wav'

input2 channels

paths - 'Channeling input2_cnls test1_1.wav' & 'Channeling input2_cnls test2_1.wav'

Mixed signals

paths - 'Channeling Output Output_signal.wav'(for 2 channel mixing) 'Channeling Output Output_signal1.wav'(for 4 channel mixing)

Seperated signals

The seperated signals is of 4 channel mixed signal above. paths - 'Channeling splt_audio Output1_x'

The dataset is available in https://drive.google.com/drive/folders/1IB9KS75TFb6BULNo91Hpz3eFWi_JzbH5?usp=sharing

Procedure

The signals recorded using mobile phones in .mp3 format. The information of the recorded signals are collected using the audioinfo function in matlab and the information is described below.

Information of Signal 1:

Filename: 'C:\Users\ayyap\OneDrive\DSP \baby elephant 60.mp3'
CompressionMethod: 'MP3'
NumChannels: 2
SampleRate: 48000
TotalSamples: 2914775
Duration: 60.7245
Title: []
Comment: []
Artist: []
BitRate: 320

Information of Signal 2:

Filename: 'C:\Users\ayyap\OneDrive\Desktop\DSP\pink_panther.mp3'
CompressionMethod: 'MP3'
NumChannels: 2
SampleRate: 48000
TotalSamples: 2936868
Duration: 61.1848
Title: []
Comment: []
Artist: []
BitRate: 320

To convert the given signals into the .wav files we had used audiowrite command. For this command we have to give signal data and sampling frequency which we will get from audioread command. Audioread command will return you signal data in the form of MxN matrix where M is the total no of samples for a channel and N is the no of channels in the given signal, and sampling frequency.
Command:
audiowrite("file name.wav",signal data,Sampling frequency).
Addition of the recorded signals is done in two methods:

Method 1:

Adding both the signals directly we can generate the output signal such that both are played at a time.
$Y_3(t)$=a*$Y_1(t)$+b*$Y_2(t)$.
where a and b belongs to set of natural numbers.

Method 2:

Adding the two signals using channels.
The input signal will have data as channels (for our input signal we have data in two channels) .For adding two

signals we place all the channels of the signals as a single matrix and audiowrite is used to get the output.(In this case we will have 2 channels of each signal and we will get 4 channels as final output).
Seperation of the output signal is done in two methods.

Independent Component Analysis(Blind Source Seperation):

If the signals are added through Method 1. For extracting the inputs signals from the mixed signal we require two mixed signals in the format $Y_3(t)$=a*$Y_1(t)$+b*$Y_2(t)$ Here we considered a = 2 and b = 1 for first signal, and for second signal we have taken a = 1 and b = 2. These signals are prewhitened(centering and whitening) before ICA is applied to generate the original signals.

Channeling:

If the signals are added using Method 2 the output signal will have the channels of the input signals in matrix format we can separate each channel to generate the seperated signals (input signals).

Theory:

Audio signal storage:
Audio signal is compressed to reduce the storage or to reduce transmission bandwidth.There are two methods to compress the audio signals they are lossy and lossless compression.
Lossy audio compression algorithms provide higher compression than lossless compression by removing less-critical data based on psychoacoustic optimizations .In lossless compression the compression will be around 50-60 percent of the original size.In general for .mp3 format they use lossy compression and in DVD-Audio,HD-DVD format lossless compression is used.There will be some formats where both types of compression will be used.
Lossy compression reduces some data by identifying the unnecessary sounds and removing them.The unnecessary sounds might be sounds with higher frequency or the sounds which are repeated at the same time. Lossy compression unsuitable for professional audio engineering applications, such as sound editing etc ,because during file decompression and compression it will undergo digital generation loss. So in this case it is better to use Lossless Commpression.
Independent component analysis(ICA):
This model is used to seperate the combined signals to the individual signals.Let s_1 and s_2 be individual signals and they are statically independent.m_1 and m_2 be linear combination of s_1 and s_2 such that m_1=a_1*s_1+b_1*s_2,m_2=a_2*s_1+b_2* s_2. In this case a_1=2,b_1=1,a_2=1 and b_2=2.

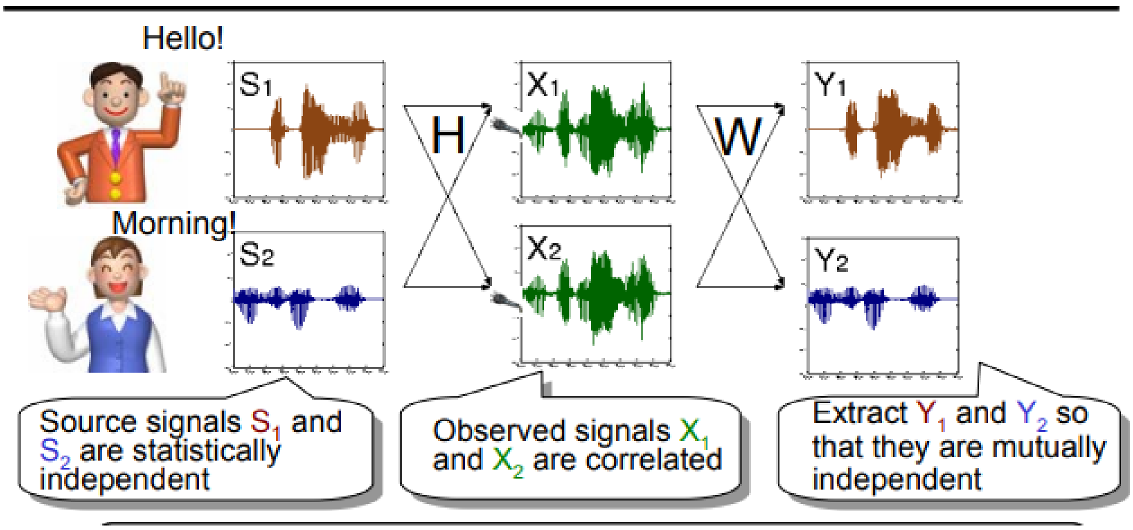


Figure 1: General example of combining of two signals

$$M = A * S \tag{1}$$

Where M is column matrix of combined signals and A is an m by n matrix(in general both m and n should be equal) and S is a column matrix of individual signals.
To separate the signals effectively, "prewhiten" the signals by using the prewhiten function.This function transforms mixdata so that it has zero mean and identity covariance.
Mean has to be zero for algorithms and we made variance to unity in order to decrease complexity.
Next we use rica command to get individual signals.The inverse of the matrix A is found and the below equation is used to get individual signals.

$$S = A^{-}1 * M \tag{2}$$

Results:

Code for Channeling

Listing 1: Code For generating of input signals and output signals

```

% Mixing two recorded signals in such a way that they are played at a
% time.

% This file contains the code for :-
% 1) Converting the recorded signals from mp3 format into .wav
% 2) Seperating the channels of the recorded signals.
% 3) Taking one channel from each signal and add them as channels to
%    generate a signal which looks like they are played at a time.
% 4) Similar to third point all channels of two signals were added
%    together.

% extracting data from first recorded mp3 file.
signal = audioread("baby elephant 60.mp3");
info = audioinfo("baby elephant 60.mp3");
Fs = info.SampleRate;
display(info);

% Converting the extracted data into .wav extension.
audiowrite("wav_inputs/input_Elephant.wav",signal,Fs);


signal = audioread("wav_inputs/input_Elephant.wav");

% Signals recorded in voice recorder will have two channels which vary in
% Power and Amplitude.
% Here, We are seperating the two channels and saving them as .wav files.
column1 = signal(:, 1);
column2 = signal(:, 2);

audiowrite("input1_cnls/test1.wav",column1,Fs);
audiowrite("input1_cnls/test2.wav",column2,Fs);

%Similar process is done for second recorded signal.
signal1 = audioread("pink_panther.mp3");
info1 = audioinfo("pink_panther.mp3");
Fs1 = info1.SampleRate;
display(info1);

audiowrite("wav_inputs/input_pink_panther.wav",signal1,Fs);
```

```

signal1 = audioread("input_pink_panther.wav");

column1_1 = signal1(:, 1);
column2_1 = signal1(:, 2);

audiowrite("input2_cnls/test1_1.wav",column1_1,Fs);
audiowrite("input2_cnls/test2_1.wav",column2_1,Fs);

a = length(column2);
b = length(column2_1);

% we will chose one of the channels from each signal to mix.
%
% Before mixing we ensure that they are of same length(same time duration)
% by padding zeroes at the end.
%
% Here, I have used the signals having higher power(just for convenience.)
if(a-b>0)
    for a = 1 : 1 : a-b
        column2_1(end+1) = 0;
        column1_1(end+1) = 0;

    end
else
    for a = 1 : 1 : b-a
        column2(end+1) = 0;
        column1(end+1) = 0;
    end
end
% We add the Signals in the form of channels taking one as left channel and
% other as right channel.
Output_signal = [column2,column2_1];

audiowrite("Output/Output_signal.wav",Output_signal,Fs);

info = audioinfo("Output/Output_signal.wav");
display(info);


% For the second case, we are adding add the four channels of recorded
% signal.
Output_signal1 = [column2,column2_1,column1,column1_1];

audiowrite("Output/Output_signal1.wav",Output_signal1,Fs);

info1 = audioinfo("Output/Output_signal1.wav");
display(info1);
```

Listing 2: Code For ploting spectrum

```

% Finding the spectrum of input signals and output signals.

% This file contains the code for :-
% 1) Finding the Spectrum of recorded signals.
% 2) Finding the Output signal formed by merging one channel from each of
    the recorded
%    signals.
```

```
% 3) Finding the Output signal formed by merging all the channels of the
    recorded signals.


% Data from first recording is extracted.
signal = audioread("wav_inputs/input_Elephant.wav");
info = audioinfo("wav_inputs/input_Elephant.wav");
Fs = info.SampleRate;
ts = info.Duration * Fs;

f = (-ts/2:ts/2-1)*(Fs/ts);

subplot(4,1,1);
% Plotting the spectrum of first recorded signal.
plot(f,abs(fftshift(fft(signal))/length(signal)));
grid on;
grid minor;
xlim([-10000,10000]);
legend('|X(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|X(f)|');
ylim([0,0.004]);
title("Spectrum of input1 signal");


% Data from the second recording is extracted.
signal = audioread("wav_inputs/input_pink_panther.wav");
info = audioinfo("wav_inputs/input_pink_panther.wav");
Fs = info.SampleRate;
ts = info.Duration * Fs;

f = (-ts/2:ts/2-1)*(Fs/ts);

subplot(4,1,2);
% Plotting the spectrum of second recorded signal.
plot(f,abs(fftshift(fft(signal))/length(signal)));
grid on;
grid minor;
xlim([-10000,10000]);
legend('|X(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|X(f)|');
ylim([0,0.004]);
title("Spectrum of input2 signal");

% Data extracted from first output file.
signal = audioread("Output/Output_signal.wav");
info = audioinfo("Output/Output_signal.wav");
Fs = info.SampleRate;
ts = info.Duration * Fs;

f = (-ts/2:ts/2-1)*(Fs/ts);

subplot(4,1,3);
% plotting the spectrum of the first Output signal.
plot(f,abs(fftshift(fft(signal))/length(signal)));
```

```
grid on;
grid minor;
xlim([-10000,10000]);
legend('|X(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|X(f)|');
ylim([0,0.004]);
title("Spectrum of 2 channel output signal");


% Data extracted from second output file.
signal = audioread("Output/Output_signal1.wav");
info = audioinfo("Output/Output_signal1.wav");
Fs = info.SampleRate;
ts = info.Duration * Fs;

f = (-ts/2:ts/2-1)*(Fs/ts);

subplot(4,1,4);
% plotting the spectrum of the second Output signal.
plot(f,abs(fftshift(fft(signal))/length(signal)));
grid on;
grid minor;
xlim([-10000,10000]);
legend('|X(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|X(f)|');
ylim([0,0.004]);
title("Spectrum of 4 channel output signal");
```

Listing 3: Code For output signal by channel

```

% Extracting the signals from the mixed signals.

% This file contains the code for the seperating the channels of 4 channel
% mixed signal to seperate all the recorded signals.

% Data extraction from the four channel mixed output signal.
[Output_signal1,Out_Fs] = audioread("Output/Output_signal1.wav");
info = audioinfo("Output/Output_signal1.wav");
display(info);

% Seperating the data into channels.
output1_1 = Output_signal1(:,1);
output1_2 = Output_signal1(:,2);
output1_3 = Output_signal1(:,3);
output1_4 = Output_signal1(:,4);

% Writing the extracted channels in .wav extension files.
audiowrite("splt_audio/Output1_1.wav",output1_1,Out_Fs);
audiowrite("splt_audio/Output1_2.wav",output1_2,Out_Fs);
audiowrite("splt_audio/Output1_3.wav",output1_3,Out_Fs);
audiowrite("splt_audio/Output1_4.wav",output1_4,Out_Fs);
```

Code for ICA

Listing 4: Code For ICA

```

% Method 2 : Mixing the recorded signals and extracting the signals using
%  ICA - Independent Component Analysis.(Blind source Seperation).

% This file contains the code for mixing the recorded signals in the form
% of a1+*S1(t)+ a2*S2(t). And seperating using ICA.

signal = audioread("baby elephant 60.mp3");
info = audioinfo("baby elephant 60.mp3");
Fs = info.SampleRate;

display(info);

audiowrite("wav_inputs/input_Elephant.wav",signal,Fs);

signal = audioread("wav_inputs/input_Elephant.wav");
info = audioinfo("wav_inputs/input_Elephant.wav");
Fs = info.SampleRate;
ts = info.Duration * Fs;
ts1 = ts;
f = (-ts/2:ts/2-1)*(Fs/ts);

column1 = signal(:, 1);
column2 = signal(:, 2);

audiowrite("input1_cnls/test1.wav",column1,Fs);
audiowrite("input1_cnls/test2.wav",column2,Fs);


subplot(4,1,1);
plot(f,abs(fftshift(fft(signal))/length(signal)));
grid on;
grid minor;
xlim([-10000,10000]);
legend('|X1(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|X1(f)|');
ylim([0,0.012]);
title("Spectrum of input1 signal");




signal1 = audioread("pink_panther.mp3");
info1 = audioinfo("pink_panther.mp3");
Fs1 = info1.SampleRate;
display(info1);

%display(signal1);

audiowrite("wav_inputs/input_pink_panther.wav",signal1,Fs);

signal1 = audioread("input_pink_panther.wav");
info = audioinfo("wav_inputs/input_pink_panther.wav");
```

```
Fs = info.SampleRate;
ts = info.Duration * Fs;

f = (-ts/2:ts/2-1)*(Fs/ts);

subplot(4,1,2);
plot(f,abs(fftshift(fft(signal1))/length(signal1)));
grid on;
grid minor;
xlim([-10000,10000]);
legend('|X2(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|X2(f)|');
ylim([0,0.012]);
title("Spectrum of input2 signal");

disp("plot")

column1_1 = signal1(:, 1);
column2_1 = signal1(:, 2);

audiowrite("input2_cnls/test1_1.wav",column1_1,Fs);
audiowrite("input2_cnls/test2_1.wav",column2_1,Fs);

a = length(column2);
b = length(column2_1);

%display(length(column2));
%display(length(column2_1));

if(a-b>0)
    max_t = ts1;
    for a = 1 : 1 : a-b
        column2_1(end+1) = 0;
        column1_1(end+1) = 0;

    end
else
    max_t = ts;
    for a = 1 : 1 : b-a
        column2(end+1) = 0;
        column1(end+1) = 0;
    end
end

S = zeros(length(column2),2);
S(:,1)  = column2;
S(:,2)  = column2_1;


rng default % For reproducibility
%mixdata = S*randn(2) + randn(1,2);
mixdata = zeros(length(column2),2);

% Here we are considering the values of a1,b1,a2, and b2(Equations mentioned
    at the starting of the code) as {2,1,1,2}
% respectively.
```

```
mixdata(:,1) = 2*S(:,1) + S(:,2);
mixdata(:,2) = S(:,1) + 2*S(:,2);

f = (-max_t/2:max_t/2-1)*(Fs/max_t);

subplot(4,1,3);
plot(f,abs(fftshift(fft(mixdata(:,1)))/length(mixdata(:,1))));
grid on;
grid minor;
xlim([-10000,10000]);
legend('|M1(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|M1(f)|');
ylim([0,0.012]);
title("Spectrum of mix1 signal");

subplot(4,1,4);
plot(f,abs(fftshift(fft(mixdata(:,2)))/length(mixdata(:,2))));
grid on;
grid minor;
xlim([-10000,10000]);
legend('|M2(f)|');
legend boxoff;
xlabel('f(Hz)');
ylabel('|M2(f)|');
ylim([0,0.012]);
title("Spectrum of mix2 signal");




for i = 1:2
    disp(i);
    %sound(S(:,i));
    %pause;
end

for i = 1:2
    disp(i);
    %sound(mixdata(:,i));
    %pause;
end

audiowrite("ica_mix/mix1.wav",mixdata(:,1),Fs);
audiowrite("ica_mix/mix2.wav",mixdata(:,2),Fs);

% Prewhitening the mixed signals will make the mean of signals zero and
% unity variance. This is very heloful for applying ICA to decrease its
% computation complexity.
mixdata = prewhiten(mixdata);


% For applying ICA we have used a pre-existing function and sperated the
% signals.
Mdl = rica(mixdata,2,'NonGaussianityIndicator',ones(2,1));

% transform fucntion will be helpful for scaling the extracted signals.
```

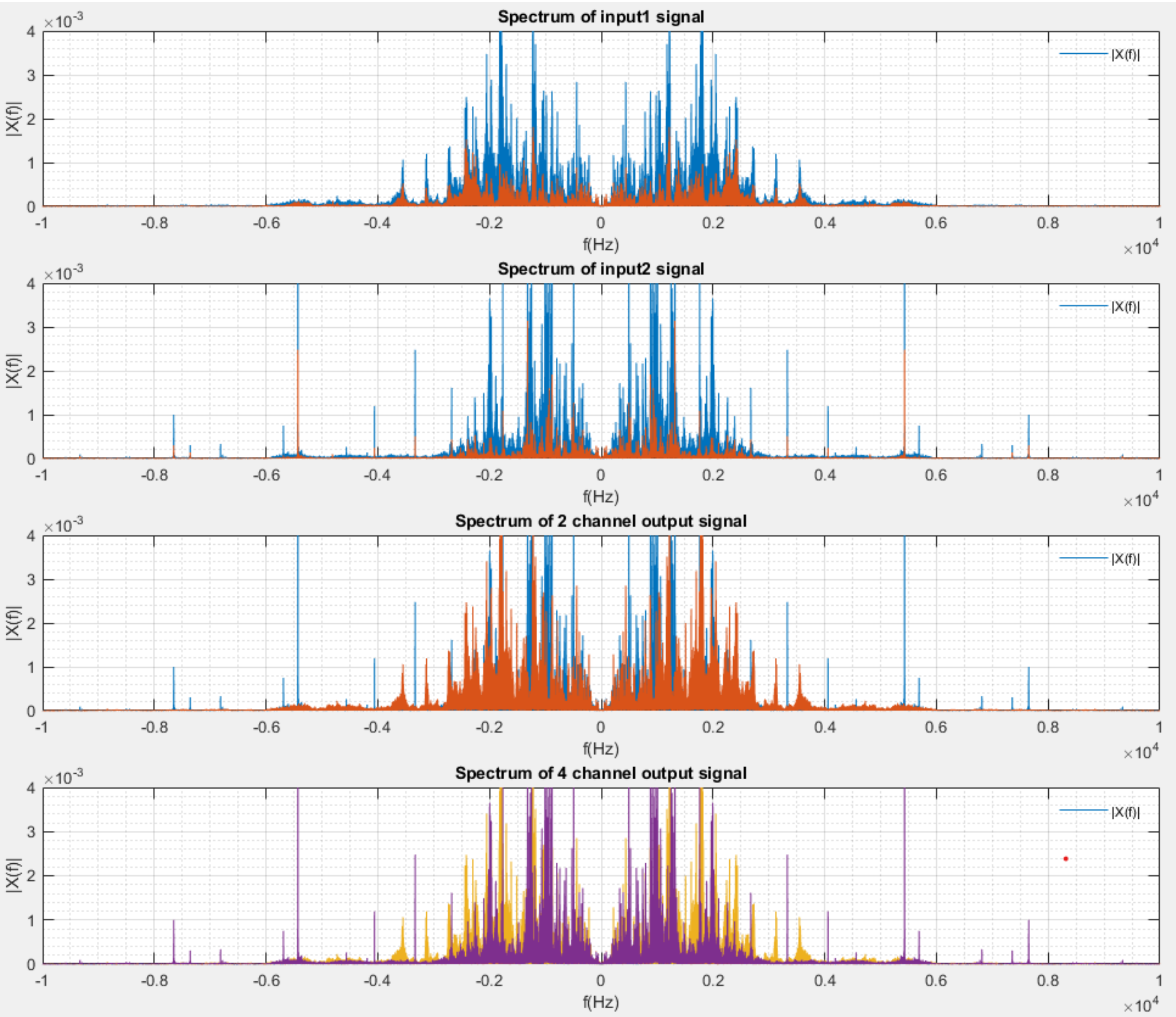
```
unmix = transform(Mdl,mixdata);

audiowrite("unmix/unmix1.wav",unmix(:,1),Fs);
audiowrite("unmix/unmix2.wav",unmix(:,2),Fs);

% This is the prewhiten function used for making mean of the signal as zero and unity variance.
function [X, W] = prewhiten(X)
    X = X - repmat(mean(X, 1), [size(X, 1) 1]);
    W = inv(sqrtm(cov(X)));
    X = X * W;
end
```

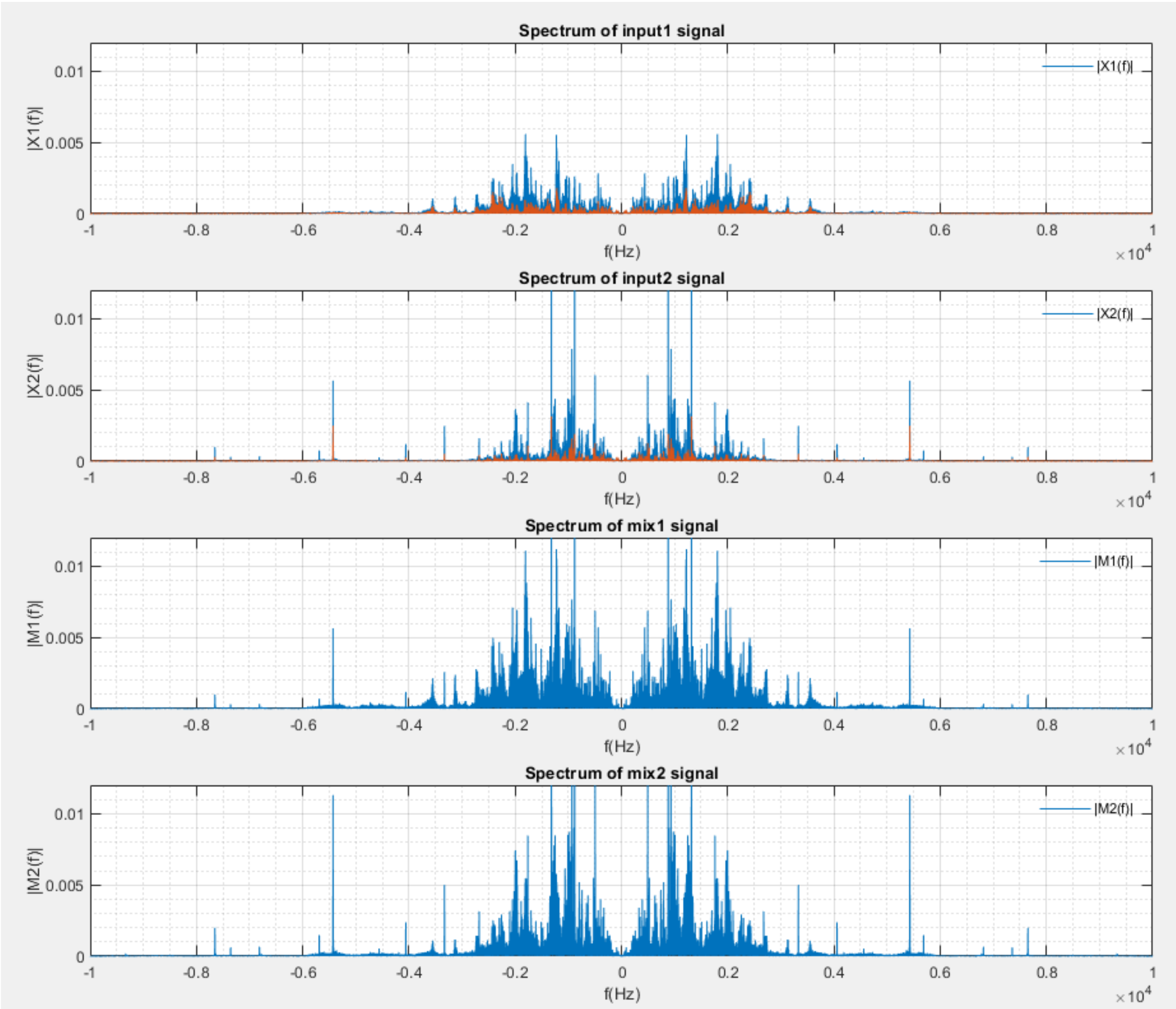
Spectrum

Plots for spectrum in channeling



The above provided spectrum are the spectrum of Input signal 1 ,Input signal 2,Output signal which is generated by taking two channels and Output signal when 4 channels is used to generate.

Plots for spectrum in ICA



Future work:

One more method can be used to add signals as well as seperation. If the sampling frequency is relatively large for a given signal this method can be used. The recorded signals are added in such a way that each bit of first signal is inserted in between two consecutive bits of signal 2 i.e, The first signal is delayed by ts/2 and is added to second signal. Here ts is the time between the two consecutive bits of given signals. The mixed signal is observed to be like a signal which is recorded when the input signals are played at a time.

The output signal is obtained by down sampling the mixed signal by 2. The issue with this process is the mixed signal has a sampling spectrum of 2fs . where fs is the sampling spectrum of input signals/recorded signals.

Listing 5: Code For extra

```
signal = audioread("input1_cnls/test2.wav");
info = audioinfo("input1_cnls/test2.wav");
signal = audioread("input1_cnls/test2.wav");
info = audioinfo("input1_cnls/test2.wav");
Fs = info.SampleRate;

signal1 = audioread("input2_cnls/test2_1.wav");
info1 = audioinfo("input2_cnls/test2_1.wav");
signal1 = audioread("input2_cnls/test2_1.wav");
info1 = audioinfo("input2_cnls/test2_1.wav");
Fs1 = info1.SampleRate;

a = length(signal);
b = length(signal1);

if(a-b>0)
    for a = 1 : 1 : a-b
        signal1(end+1) = 0;

    end
else
    for a = 1 : 1 : b-a
        signal(end+1) = 0;
    end
end
out = [];
for i = 1 : 1 : 2 * max(a,b)
    if(mod(i,2) == 1)
        out(end+1) = signal((i+1)/2);
    else
        out(end+1) = signal1((i/2));
    end
end

audiowrite("sampling_out/test_Out.wav",out,2*Fs);

ts = max(info.Duration,info1.Duration) * 2 * Fs;

f = (-ts/2:ts/2-1)*(2*Fs/ts);

plot(f,abs(fftshift(fft(out))/length(out)));
grid on;
grid minor;
%xlim([-150000,150000]);
legend('|X(f)|');
legend boxoff;
xlabel('f(Hz)');
```

```
ylabel('|X(f)|');
%ylim([0,0.004]);
title("Spectrum of input1 signal");
```

References

https://in.mathworks.com/help/stats/extract-mixed-signals.html

https://signalprocessingsociety.org/sites/default/files/Audio%20Source%20Separation%20based%20on%20Independent%20Component%20Analysis%20-%20Makino%20Sawada%202007.pdf

https://sci.utah.edu/~shireen/pdfs/tutorials/Elhabian_ICA09.pdf