International Institute of Information Technology
Bangalore

Signal Processing
EC304

Audio Signal Processing

Name of the student: Aakash Khot (IMT2020512)
Name of the student: Kedar Deshpande (IMT2020523)

April 17, 2022

Files

Drive link: https://drive.google.com/drive/folders/1RAfNjkd31-Tzc2bWeFiszyoTQkXzVQMB?usp=sharing

INTRODUCTION

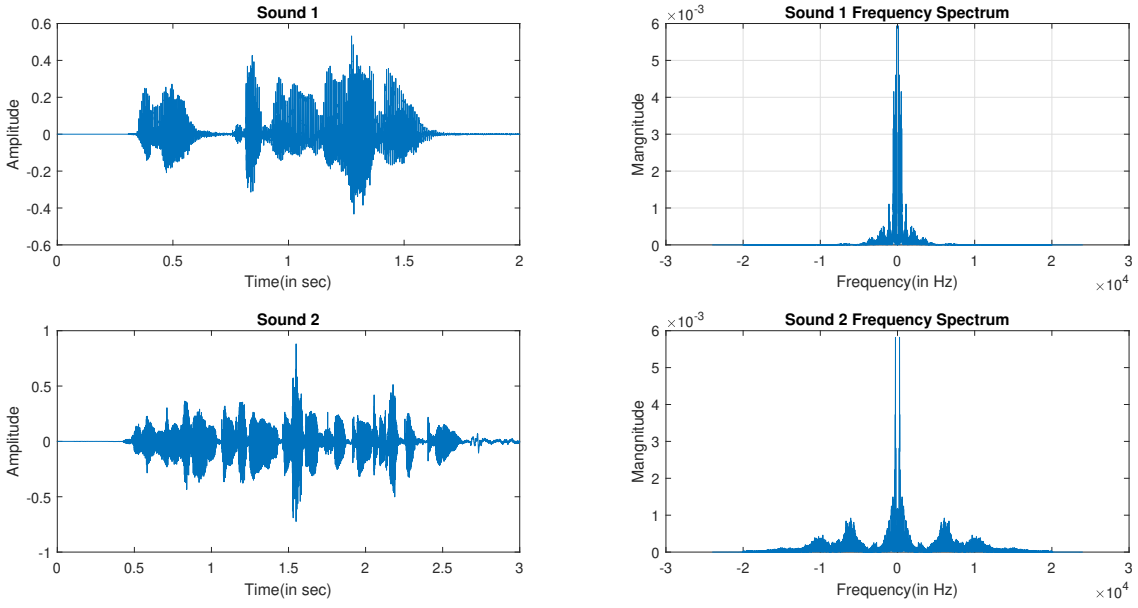
- In the Project, Our main aim is to separate two audio signals from one signal.
- We recorded two audio signals from our phone and analysed the signal by its spectra, sampling frequency, format of the signal, number of channels etc.
- Then with the use of MATLAB, we mixed two audio signals as if they speech signals were spoken simultaneously. We linearly added the signal matrix representation one by one to get the mixed signal in MATLAB. Other way can be adding both the signals as two channels.
- The we analysed the combined signal and tried to separate it to the two recorded signals we took at start.
- We also went through data compression, how signal is compressed and stored in memory with minimal loss.

DATASETS

Audio file format : mp4->wav (converted)
Sampling Frequency : 48 KHz
Number of channels : 1 (Non-stereo mode)
Number of sounds : 2

Graphs :-

Sound 1 is voice of a boy and Sound 2 is voice of a girl taken from mobile using voice recorder. Given below are the plots of both voices with respect to time and its frequency spectrum.



CODE

Ensure that your current working folder is set properly in MATLAB.
Use of rica() and prewhiten() functions were taken from here.[2]
Line number 40 - Here we are creating two different mixed samples of sounds. They both differ in weights of each sound. Each column in s is a sound. So after multiplying by a random 2x2 matrix we get 2 samples of mixed sound(a*s1+b*s2) in 2 columns of out.

```
[s1o,Fs1]=audioread('sound1.wav');
s1=resample(s1o,48000,44100);
s1f=fftshift(abs(fft(s1))/length(s1));
Fs2=48000;
t1=linspace(0,2,length(s1));
figure;
subplot(2,1,1);
plot(t1,s1);
title("Sound 1");
xlabel("Time(in sec)");
ylabel("Amplitude");
f=Fs2/2*linspace(-1,1,length(s1f));
subplot(2,1,2);
plot(f,s1f);
title("Sound 1 Frequency Spectrum");
xlabel("Frequency(in Hz)");
ylabel("Mangnitude");
[s2,Fs2]=audioread('sound2.wav');
s2f=fftshift(abs(fft(s2))/length(s2));
f2=Fs2/2*linspace(-1,1,length(s2f));
t2=linspace(0,3,length(s2));
figure;
subplot(2,1,1);
plot(t2,s2);
title("Sound 2");
xlabel("Time(in sec)");
ylabel("Amplitude");
subplot(2,1,2);
plot(f2,s2f);
title("Sound 2 Frequency Spectrum");
xlabel("Frequency(in Hz)");
ylabel("Mangnitude");

%-------------------
% Taking equal lengths of both sounds
to mix(least one)
s(:,1)=s1;
s(:,2)=s2(1:length(s1));
rng default;
out=s*randn(2);
figure;
subplot(2,1,1);
plot(t1,out(:,1));
title("Mixed sound sample 1");
xlabel("Time(in sec)");
ylabel("Amplitude");
subplot(2,1,2);
plot(t1,out(:,2));
title("Mixed sound sample 2");
xlabel("Time(in sec)");
ylabel("Amplitude");

out1f=fftshift(abs(fft(out(:,1)))/length(out(:,1)));
f3=Fs2/2 * linspace(-1,1,length(out1f));
figure;
subplot(2,1,1);
plot(f3,out1f);
title("Mixed sound sample 1 Frequency Spectrum");
xlabel("Frequency(in Hz)");
ylabel("Mangnitude");
out2f=fftshift(abs(fft(out(:,2)))/length(out(:,2)));
f3=Fs2/2 * linspace(-1,1,length(out2f));
subplot(2,1,2);
plot(f3,out2f);
title("Mixed sound sample 2 Frequency Spectrum");
xlabel("Frequency(in Hz)");
ylabel("Mangnitude");

%-------------------------
%Finding correlation between original signals
%and mixed signal
cr1=corr(s1,out(:,1))
cr2=corr(s2(1:length(s1)),out(:,1))
%-------------------------

audiowrite('Processed/mixed1.wav'
,out(:,1),Fs2);
audiowrite('Processed/mixed2.wav'
,out(:,2),Fs2);
%-----------------------------------------
out_w=prewhiten(out);
sep=rica(out_w,2);
separated=transform(sep,out_w);

%-------------------------
%Finding correlation between separated signals
%and mixed signal
cr1n=corr(separated(:,1),out(:,1))
cr2n=corr(separated(:,2),out(:,1))
%-------------------------

figure;
subplot(2,1,1);
plot(t1,separated(:,1));
title("Separated Sound 1");
xlabel("Time(in sec)");
ylabel("Amplitude");
subplot(2,1,2);
plot(t1,separated(:,2));
title("Separated Sound 2");
xlabel("Time(in sec)");
ylabel("Amplitude");
```

```
audiowrite('Processed/sep1.wav'
,separated(:,1),Fs2);
audiowrite('Processed/sep2.wav'
,separated(:,2),Fs2);

%-----------------------------------------
%-----------------------------------------

function Z = prewhiten(X)
% X = N-by-P matrix for N observations
%and P predictors
% Z = N-by-P prewhitened matrix

    % 1. Size of X.
    [N,P] = size(X);
    assert(N >= P);

    % 2. SVD of covariance of X.
    %We could also use svd(X) to proceed
    %but N
    % can be large and so we sacrifice
    %some accuracy for speed.
    [U,Sig] = svd(cov(X));
    Sig     = diag(Sig);
    Sig     = Sig(:)';

    % 3. Figure out which values of
    %Sig are non-zero.
    tol = eps(class(X));
    idx = (Sig > max(Sig)*tol);
    assert(~all(idx == 0));

    % 4. Get the non-zero elements of
    %Sig and corresponding columns of U.
    Sig = Sig(idx);
    U   = U(:,idx);

    % 5. Compute prewhitened data.
    mu = mean(X,1);
    Z = bsxfun(@minus,X,mu);
    Z = bsxfun(@times,Z*U,1./sqrt(Sig));
end
```

PROCEDURE

1. Recorded two audio signals from mobile phone as in mp4 format.
2. We converted the MP4 format to .wav format using MATLAB.
3. In MATLAB, We read both the audio (S1 and S2) using audioread() function and resampled one signal to the sampling frequency.
4. Now to mix these audio signals, we made two different channels for sound 1 and 2 of same length and multiplied each by a random 2 by 2 matrix to get mixed sample sound 1 and 2. Random selection of matrix so that we get different weights of samples.
5. We used independent component analysis (ICA) method to separate these combined signals.
6. We passed the combined signal to prewhiten() function to separate the signals effectively.This function transforms mixed data so that it has zero mean and identity covariance.
7. Then We performed Reconstruction ICA by inbuilt function "rica" to unmix the signal to get separated signals.
8. Then we checked the plots of the separated signals we got and compared them with the original signals to verify.

THEORY

Independent Component Analysis (ICA)

Independent Component Analysis (ICA) is a technique to separate independent sources from mixed signals. ICA is a special case of blind source separation. [4]

Blind source separation (BSS) is the separation of a set of source signals from a set of mixed signals in which we don't have the information of source and mixing of signals. Some of the applications of Blind source separation are Image processing, Medical imaging, Electroencephalogram (EEG) etc.[6]

ICA attempts to separate the mixed signals into non-Gaussian signal. When some assumptions correct, ICA gives very good results. Let's take an application to understand.

Cocktail Party Problem [7] is an ability to focus on one voice and filtering other voices which are present in the room.

Cocktail Party Problem where speech signals are separated from voice data of people talking simultaneously at a time interval in room. Here we assume that there is no time delays so that the statistical independence assumption is correct.

In simple language, imagine two people talking to you, the waves of both voices will combine and reach your ears but our brain will unmix them and I will hear more of the voice of person which closer to me. So we can say that the signal matrix we get is a product of two matrices, where one is source signals and other is mixing matrix. Now we can retrieve source signal by multiplying inverse of mixed signal to observed signal. So If we get inverse of mixed signal,we can get the source signals. Now let's see some assumptions of ICA.

Assumptions of ICA:-

1. Source Signals should be independent of each other.
2. The values of all source signals should have non-Gaussian distributions.
3. The number of independent components generated should be equal to number of observed mixtures.[5]

Now assuming that these conditions are satisfied by source signals. Now we have to subtract the mean from input so that the mixed signals have zero mean and source signals have also mean zero. Now for the whitening step, we have to make variance unity. In our prewhiten() function, it makes the mean zero. Then we need to optimize the mixed signals so that the observed signal we get is non Gaussian.

For doing this, we need a measure of gaussianity which we can get by **kurtosis**, which is fourth moment of data which measures "tailedness" of a distribution. Here we subtract 3 from the fourth moment of data so that distribution has kurtosis 0.

But in out algorithm, what we do instead of this is take random weights for each component and dot product of this with mixed signal is passed through contrast function of ICA and its inverse and then we subtract the output of it and calculate the mean. Then we will divide the new weights with its norm until we get same result. In these steps, we have to decorrelate the new weights with previous one so that we won't get the same component in further iterations.

Now we can check with the result by checking the kurtosis value if it is less gaussian or not. [1] But there are some drawbacks for this algorithm.

Drawbacks of ICA:-

1. Other than dimensions of input, it is not possible to learn more features.
2. The whitening step (prewhiten() function) is expensive for high dimension inpus data.
3. Constrained optimizers, which are required to solve regularization problem in ICA, are very slow in process and expensive.

These drawbacks are because of normality constraint in ICA formulation. To overcome these drawbacks, Reconstruction ICA (RICA) was proposed where normality constraint is replaced by linear reconstruction penalty.

Benefits of RICA over ICA:-

1. **Overcomplete feature learning**
   In this, the number of components (features) is greater than dimensionality of input. ICA does not work when this holds because orthogonality constraint cannot be held.But this constraint is removed in RICA and because of this overcomplete features can be learned. Also comparing this method with other approaches, this method is easier to implement and have faster convergence rate.
2. **Lower sensitivity to whitening**
   ICA requires input to have zero mean and one variance. This is done by whitening step (prewhiten() function) but sometimes ICA produces noisy filters with data not whitened. This can be overcome by using Principal component analysis (PCA) but its very expensive and not feasible for high dimension input.Compared to this RICA is very less sensitive to whitening and works well with approximate whitening or even without whitening.
3. **Unconstrained optimizer**
   From the third drawback, optimizers are required to solve regularization problem in ICA but are very slow in process and expensive. As this constraint is removed in RICA, unconstrained optimizers can be used which have faster convergence rate.

Data compression

Data compression in signal compression can also be called as bit-rate reduction which is the process of encoding information using fewer bits that the original representation. There are two types of compression, lossy or lossless.

Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossless compression is possible because of statistical redundancy of real world data.

The Lempel–Ziv (LZ) compression methods are the most famous algorithms for lossless storage. In mid 1980s, the work by Lempel–Ziv–Welch (LZW) algorithm became the most used method for general-purpose compression systems.The strongest modern lossless compressors use probabilistic models, such as prediction by partial matching and statistical modelling.

Lossy compression reduces bits by removing unnecessary or less important information.

In early 1990s, Lossy compression methods started being widely used. As storage space is saved and we are dropping non-essential information which is acceptable. We can also say that there is a trade-off between preserving information and reducing size.

One of the example where this is used is JPEG image compression. Most of the lossy compression is used in transform coding that is in Discrete Cosine Transform (DCT). DCT is most widely used method of lossy compression and is used in audio, video, images etc.

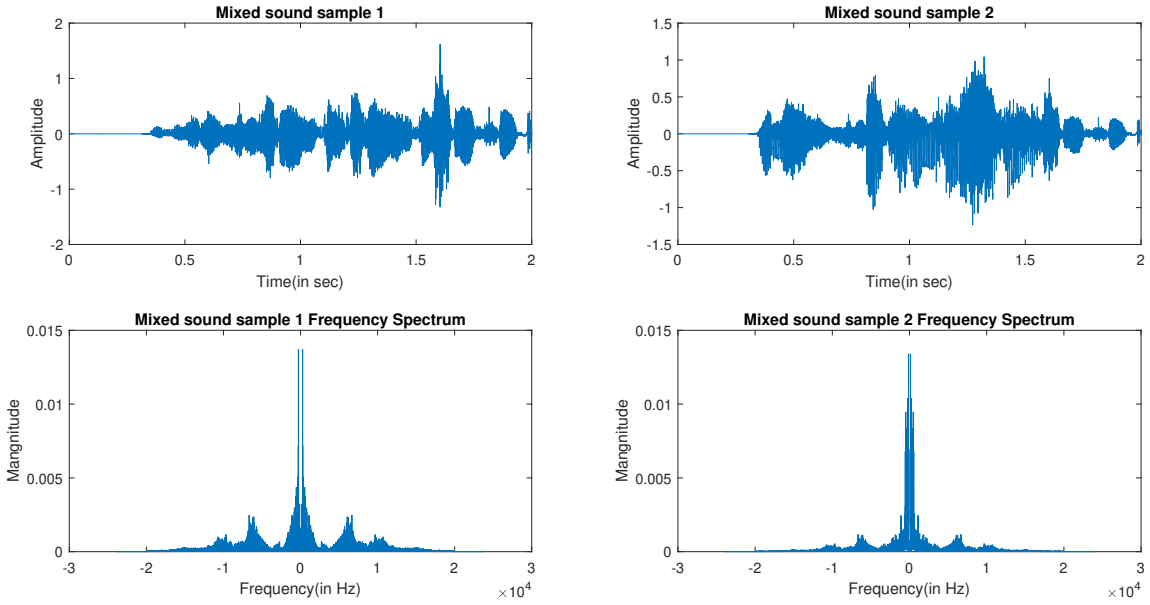
Lossy image compression methods are also used in digital cameras to increase storage capacity, DVDs, Blu-ray, streming video etc.

Lossy audio compression methods are used to remove non-essential or non-audible or we can say less audible components from audio signal. Human speech compression is done even with more specialised techniques.[3]

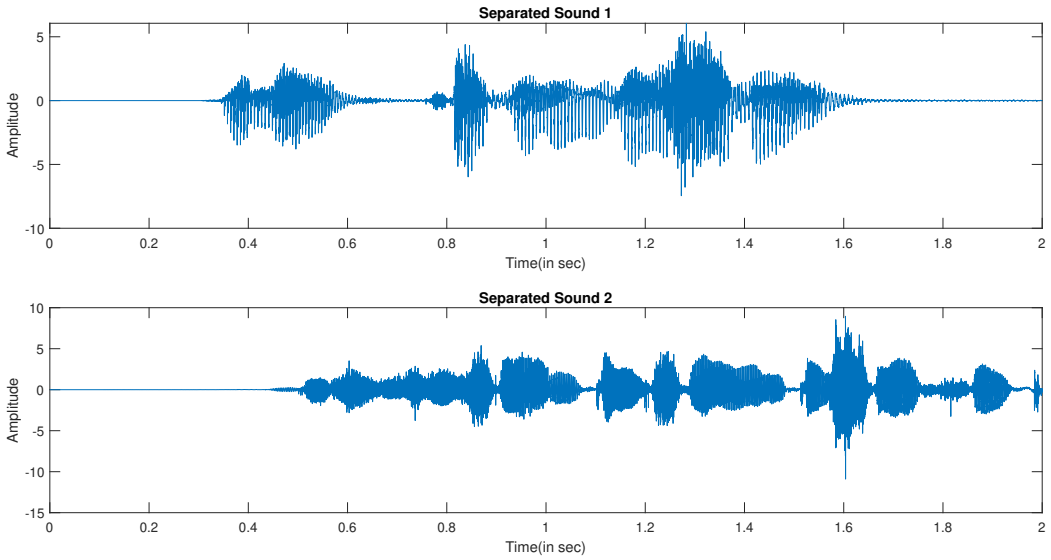
RESULTS

Correlation between original sound and mixed sound(1):
For original sound 1: 0.2572
For original sound 2: 0.9681
Correlation between separated sound and mixed sound(1):
For separated sound 1: -0.2524
For separated sound 2: -0.9677

Graphs :-



Graphs above are the plots of mixed signal 1 and mixed signal 2 with its Frequency spectrum which is the combined signal we got from two original signals.



Graphs above are plots of separated signals we got from performing ICA method on mixed signal. It can be seen that these almost match to the original sound signals.

FUTURE WORK

1. We separated two audios here. This can be extended to separate many more audios. This will help us analyse different audios(that were spoken at the same time) separately. This separate analysis can help identify the nature of those sounds and the differences when they are mixed.
   Audio separation can also help in identifying important sounds from a sample like, in a crime case to get only the suspects voice.
2. ICA algorithm can be used in the field of image processing. It can be widely used to distinguish between many things and can be very helpful for research purposes. This can also be used in investigation department to catch culprits and also many other used for clues by observing differnt types of images in investigation department.

References

https://towardsdatascience.com/separating-mixed-signals-with-independent-component-analysis-38205188f2f4

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

https://en.wikipedia.org/wiki/Data_compression

https://en.wikipedia.org/wiki/Independent_component_analysis#:~:text=In%20signal%20processing%2C%20independent%20component,statistically%20independent%20from%20each%20other.

https://www.geeksforgeeks.org/ml-independent-component-analysis/

https://en.wikipedia.org/wiki/Signal_separation

https://en.wikipedia.org/wiki/Cocktail_party_effect