אוניברסיטת בר-אילן

Faculty of Engineering
Signal Processing Laboratory

Blind Source Separation in Noisy Environments Using Model-based EM Algorithm

Ofek Ophir

Afek Steinberg

Final Project of BSc in Electrical Engineering

Advisor: Mr. Aviad Eisenberg

Academic Advisor: Prof. Sharon Ganot

2020

Table of Contents

Abstract

Speech separation is a core problem is signal processing, real world speech signals often involve distortions such as reverberation, interfering speakers, and noise.
In this context, source separation refers to the problem of extracting the target speakers within the mixture while cancelling interfering speakers and/or noise.
This problem may arise in various real scenarios, for instance, spoken communication over mobile phones, conference call systems, hearing aids and even video game consoles.

In this project we present the problem of blind source separation of speech signals in noisy environments using multiple microphones.
Blind estimation of the acoustic parameters for each one of the source signals is initialized using the Degenerate Unmixing Estimation Technique (DUET), and carried out using the Expectation Maximization (EM) algorithm.
The latent data are estimated in the E step of the algorithm, whereas in the M step, the algorithm estimates the acoustic transfer function of each source and the noise covariance matrix.

In the model presented in this project, the clean signal is assumed deterministic unknown, meaning that only the a posteriori probabilities of the presence of each source are estimated in the E step, while the time-frequency coefficients of the parameters are estimated in the M step using the minimum variance distortionless response beamformer (MVDR beamformer).

The algorithm was then tested using a noisy mixtures of two speech sources in different reverberation and noise settings.
The recording device used in this project is a MATRIX Voice (with 8 MEMs microphone array) connected to a Raspberry Pi 3, giving us an 8-channel audio signal of the mixture containing the two speakers in the presence of noise and reverberation.

This project is based on the paper:
B. Schwartz, S. Gannot, E. Habets, "Two Model-Based EM Algorithms for Blind Source Separation in Noisy Environment", "IEEE/ACM Transactions on Audio, Speech, and Language Processing", 2017/01/01

A video demonstration of this project can be found in the following link:
https://www.youtube.com/watch?v=3byVhd68VZQ

Gratitude

In this project we had faced many difficult tasks and without the help of the staff at Bar Ilan University we wouldn't have been able to surpass our expectations from ourselves.

We would like to give special thanks to Mr. Aviad Eisenberg who advised us during the project and have always been patient with our many, many questions.

Another big thank you for our laboratory engineer Mr. Pinchas Tandeitnik and academic advisor Prof. Sharon Ganot.

We would also like to thank our family and friends who had to listen to us complain and were kind enough to advise us.

"As we express our gratitude, we must never forget that the highest appreciation is not the utter words, but to live by them" – John F. Kennedy.

Presenting the Problem

As we have mentioned before, our project deals with blind source separation in a noisy environment.

Speech signals recorded in a room are commonly degraded by reverberation. In most of the cases, both the speech signal and the acoustic system of the room are unknown and time-varying,

Blind source separation is the study of separating a set of source signals from a set of mixed signals with very little information (and sometimes none at all) about the source signals or the mixing process.

Most commonly, blind source separation is applied in digital signal processing.

In our project, we have dealt with BSS of speech signals in an environment which includes an unknown noise that is statistically independent from the clean source signals and reverberation effects.

An acoustic sound or speech that propagates in an enclosure/room is repeatedly reflected from the walls and other objects, the phenomenon, usually referred to as reverberation, degrades the speech quality and in severe cases, the intelligibility.

In recent years, due to advances in understanding the phenomenon, and the availability of stronger computational resources, the interest in dereverberation increased and numerous methods were proposed.

However, some BSS algorithms still struggle with a very reverberant environment, one example of such algorithm is the DUET, which fails at the presence of reverb. If we look at the histogram of symmetric attenuation and delay estimate pairs for two sources which DUET yields (figure 1), we could see that the peaks are far less distinguishable when the Room Impulse Response (RIR) includes a lot of reverb/echo.

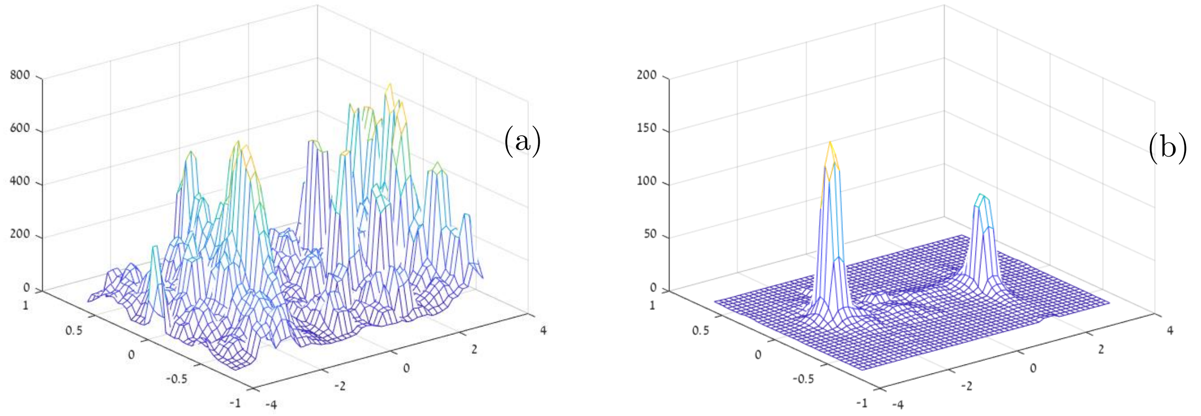


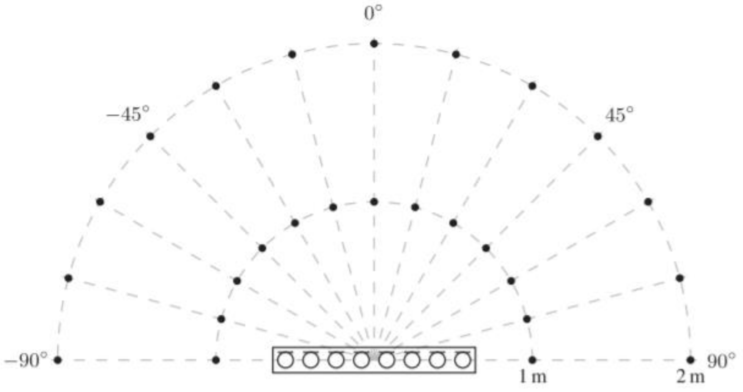
Figure 1. (a) histogram of symmetric attenuation and delay estimate pairs for RIR with t_{60}=610 ms. (b) histogram of symmetric attenuation and delay estimate pairs for RIR which is a delta function.

Blind source separation is in general a highly underdetermined problem, though useful solution can be derived, one of which is the EM algorithm used in our project. BSS is used today in many different fields such as image processing and audio processing, each field has a set a conditions in which said field confronts.

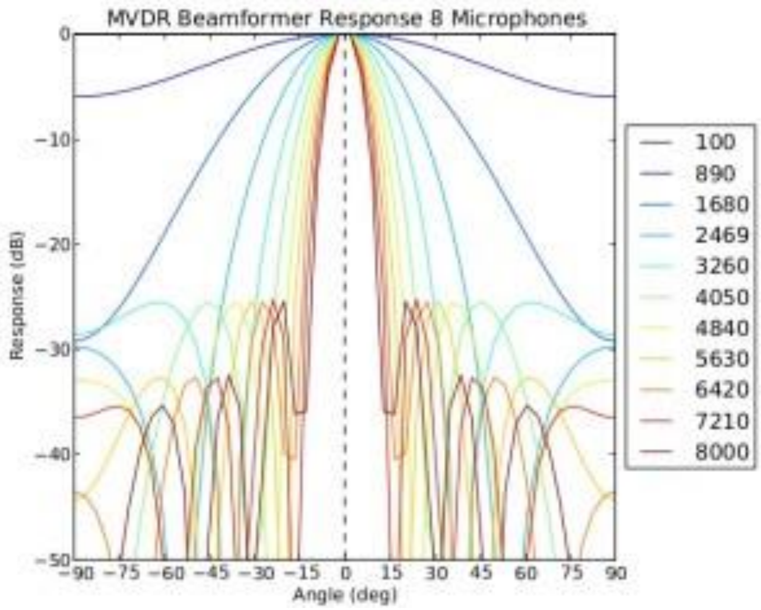
In this project, we use multichannel microphone array analysis and various spatial signal processing methods in order the achieve a qualitative source separation.
As of recent years, the number of microphones per device has steadily increased, today most smartphones, tablets and other systems are equipped with two or three microphones.
Conference call system with eight microphones are commercially available and research prototypes with forty to hundreds of microphones have been demonstrated. The analysis capabilities offered by these multichannel interfaces are usually greater than those of single-channel interfaces.
These devices make it possible to design multichannel spatial filters that can selectively enhance or suppress sounds in certain directions.
Blind source separation schemes can be used in such devices and achieve a superior result to that of the single channel analysis.

One such spatial analysis method is the one used in our project, the Minimum Variance Distortionless Response beamformer (MVDR) is used to minimize the variance of the recorded signal.
If the noise and the underlying desired signals are uncorrelated, as is typically the case, then the variance of the recorded signal is the sum of the variances of the desired signal and the noise. Therefore, the MVDR solution seeks to minimize this sum, and thereby mitigating the effects of the noise.

The MVDR beampattern is highly dependent on the number of microphones.
In our case, the number of microphones in our array is 8 as follows:



For this type of structure, the MVDR response is robust, and can be visualized as follows:



As the figure shows, the mainlobe is about 15 degrees as the Nyquist frequency is approached, and the sidelobes offer about 25dB of suppression.
As the frequency lowers, the mainlobe gets wider, and the sidelobes get higher, but not to a great extent.

For us, the most difficult issue we have dealt with is the stabilization of the EM algorithm.
In theory the implementation of such algorithm is supposed to be straightforward, but in practice, to implement the algorithm correctly takes a lot of effort.
The initialization of the EM algorithm, the gain ambiguity and correct way to implement each step were our most difficult tasks, and we will expand more on that later.
Overall we are happy with what we had accomplished, and we are glad to have taken this task as our final BSc degree project.

Theoretical background

Short Time Fourier Transform

The entire algorithm works in the Time-Frequency domain, so first, we must explain what the STFT transform is.

As we know, analyzing the signal in the time domain does not give us any info about its frequency characteristics.

If we want to analyze a signal's frequency characteristics we must apply the Fourier Transform as such:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(t)e^{-2\pi jt\xi}dt$$

And for a discrete signal, such as any signal that has been sampled, we can use the DFT (Discrete time Fourier Transform):

$$X_{2\pi}(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

The problem with using the Fourier transforms is that we lose any information about the time characteristics of our signal.

And what if the signal varies both in time and frequency? Meaning that as time goes on, the frequency characteristics change as well, the signal in this case is not WSS (wide sense stationary).

For this we can use that Short Time Fourier Transform (STFT).

The Short Time Fourier Transform is a Fourier related transform which is used to determine the frequency contents of location sections of a signal as it changes over time. Meaning that we can analyze how the frequency characteristics of the signal change over time.

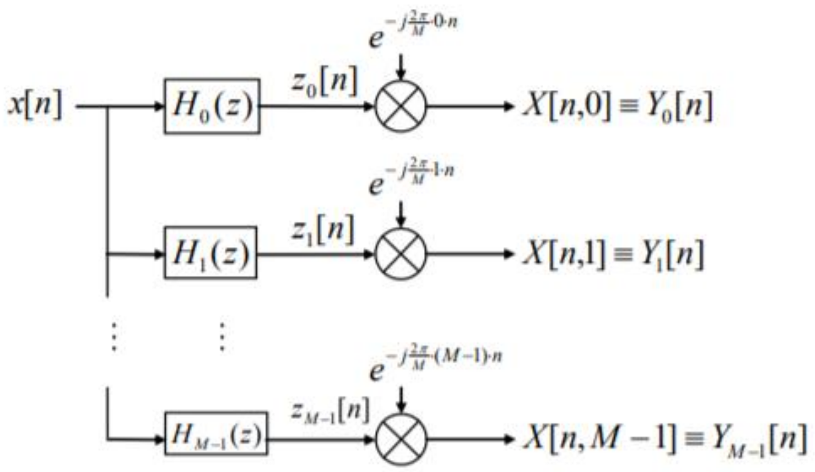
In practice, the procedure for computing the STFT of a signal is to divide the longer signal into shorter time segments of equal length and compute the Fourier Transform separately on each shorter segment.

This method reveals the Fourier Spectrum on each shorter segment, so we can plot the changing spectra as a function of time, known as a spectrogram.

We can define the Discrete Short Time Fourier Transform of a discrete signal as:

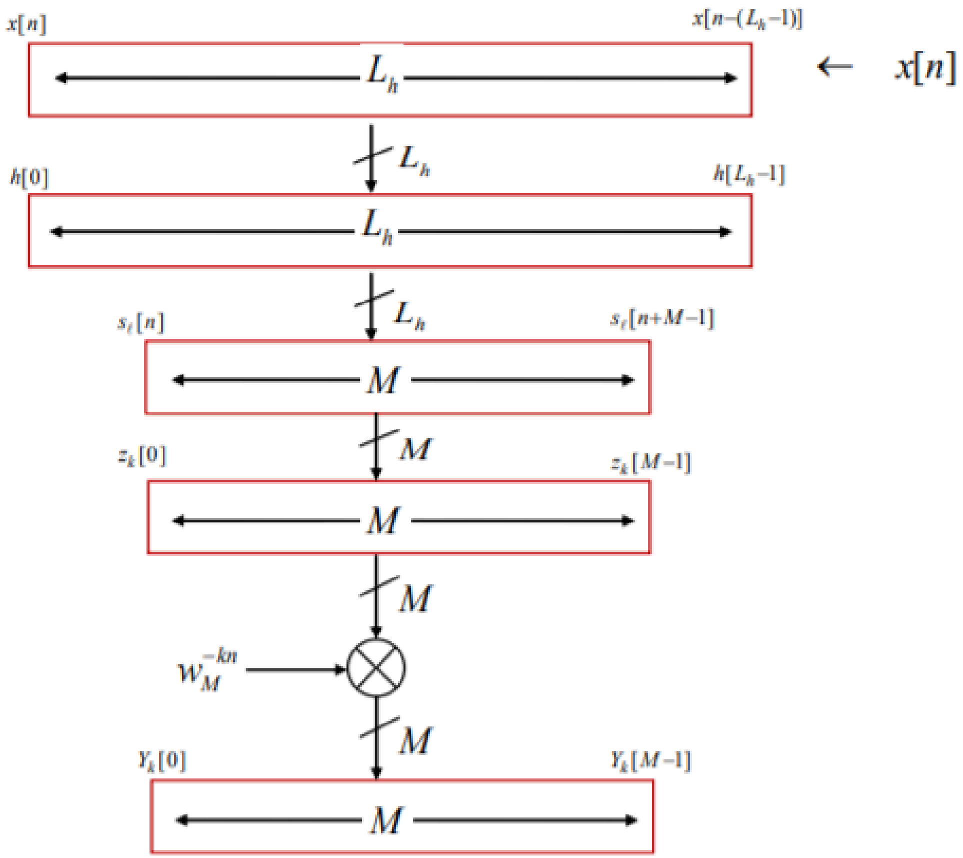
$$X_{STFT}[n,k] = X_{STFT}\left(e^{j\omega}, n\right)\Big|_{\omega=\frac{2\pi}{M}k} = \sum_{m=-\infty}^{\infty} x[m]w[n-k]e^{-j\frac{2\pi}{M}km} \qquad k = 0,1,\dots,M-1$$

$x[n]$ is a discrete infinite signal in time.
$w[n]$ is the analysis window with length of L_h.
We can describe the STFT of a signal using a filter bank as so:


$x[n]$
$H_0(z)$
$z_0[n]$
$e^{-j\frac{2\pi}{M}0\cdot n}$
$X[n,0]\equiv Y_0[n]$
$H_1(z)$
$z_1[n]$
$e^{-j\frac{2\pi}{M}1\cdot n}$
$X[n,1]\equiv Y_1[n]$
$H_{M-1}(z)$
$z_{M-1}[n]$
$e^{-j\frac{2\pi}{M}(M-1)\cdot n}$
$X[n,M-1]\equiv Y_{M-1}[n]$

When $h_k[n] = w[n]e^{j\frac{2\pi}{M}kn} \leftrightarrow H_k\left(e^{j\omega}\right) = W\left(e^{j\left(\omega-\frac{2\pi}{M}k\right)}\right)$.

We can also illustrate the entirety of the analysis process as:



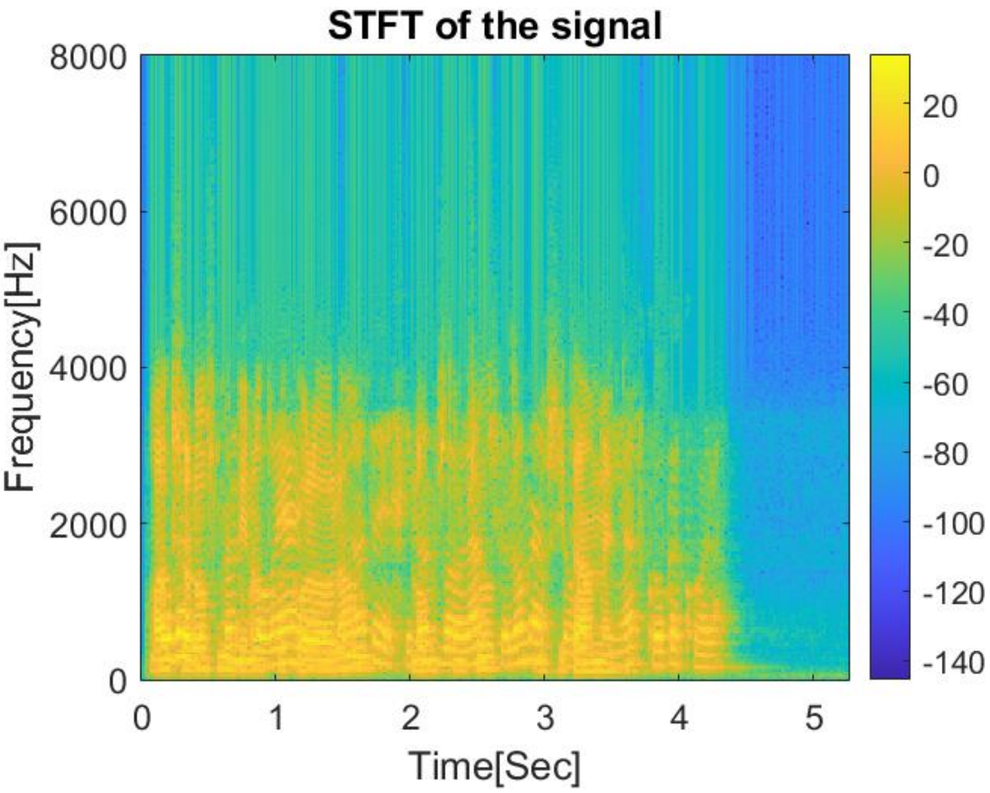
1. For each index n we multiply the signal $x[n+m]$ with $h[-m]$

2. For L_h>M and a multiple of M (otherwise we add zeros), we fold the buffer into itself $\frac{L_h}{M}$ times.
3. We perform the DFT in length of M for the folds.
4. We multiply by the exponent $e^{-j\frac{2\pi}{M}kn}$

It is worth mentioning that the illustrations above refer to a STFT with $R = 1$, meaning that we only insert 1 new sample of the signal in each iteration, usually we decimate the signal and then $R > 1$.

After the STFT analysis of the signal we are left the spectrogram in the Time-Frequency domain.

An example of such spectrogram for a speech signal with noise and $T_{60} = 160ms$ reverberation can be visualized as:

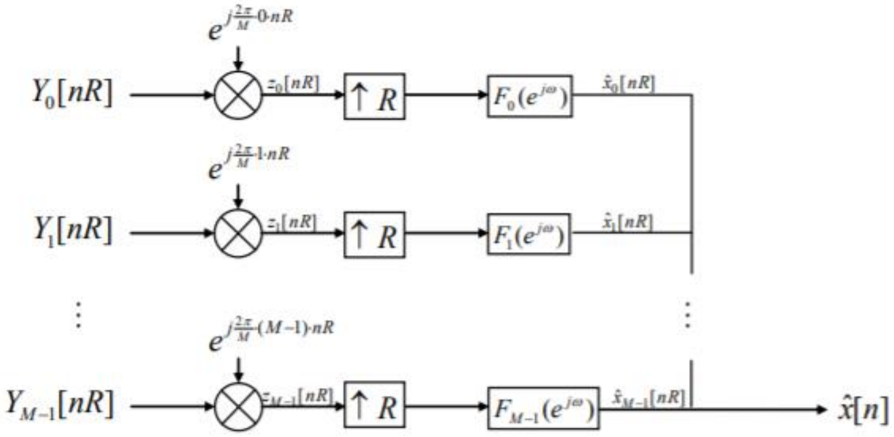


As we can see, the y-axis represents the frequency and the x-axis represents the time. The color (yellow or blue) represents the power of the signal in the TF-bin, in this case, yellow represents a stronger signal.

The ladder of value to the right correlates the color of the TF bin to the power of the signal in dB, making this plot a sort of 3-dimensional graph.

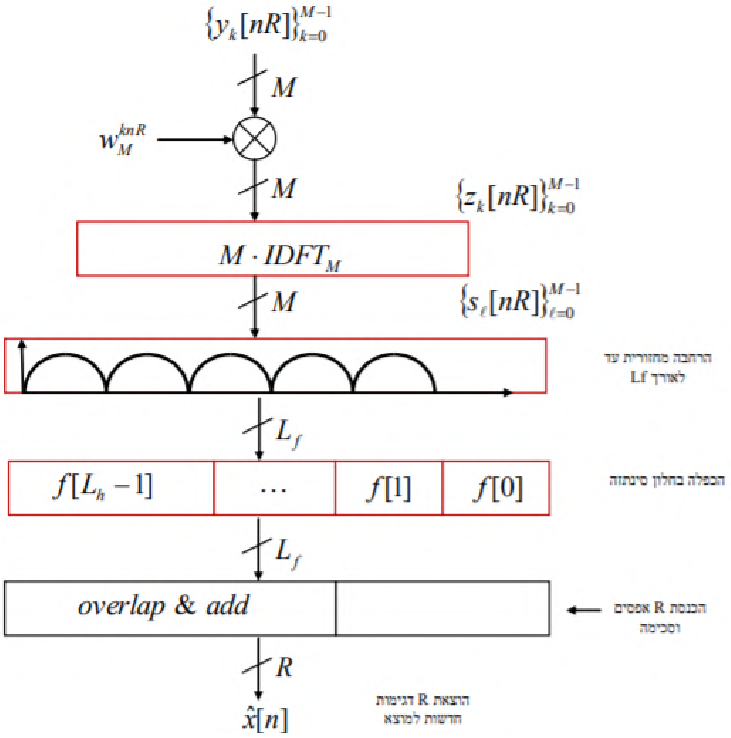
The inverse STFT transform, reverses the STFT analysis and results in the time representation of the signal.

The iSTFT can be illustrated using a filter bank as:



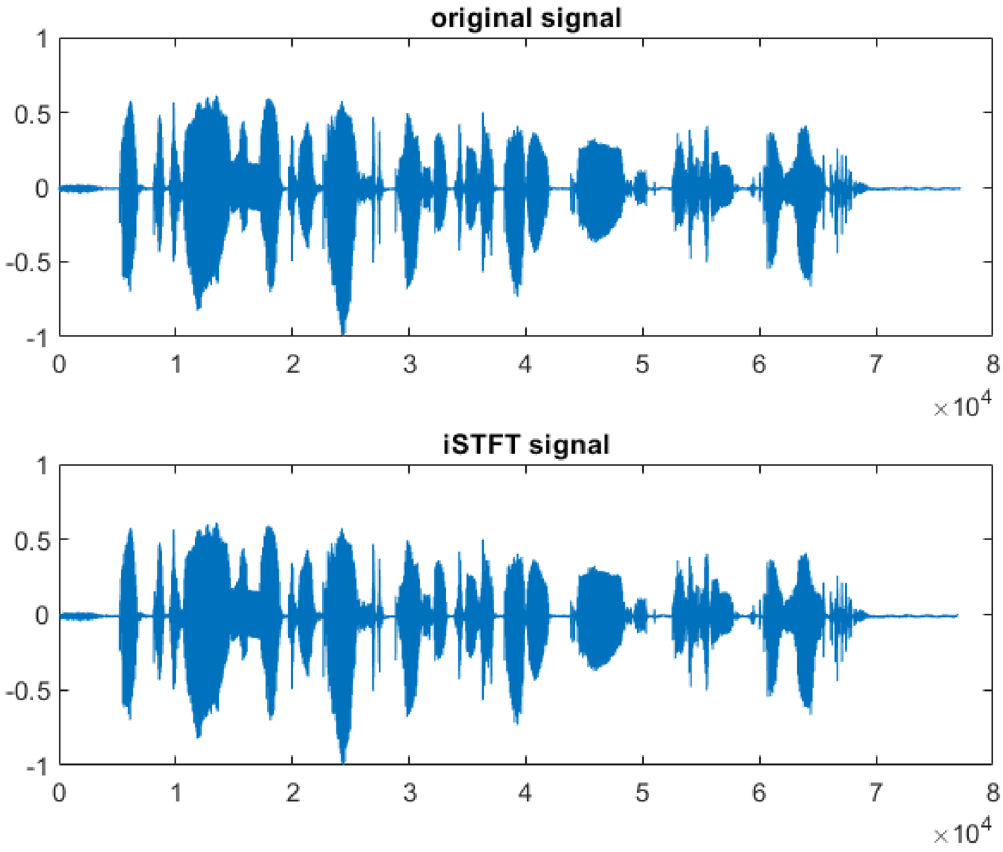
1. Multiplying by the exponent in order to offset the analysis exponent and get $z_k[nR]$ (in the STFT illustration we only showed $R = 1$ so we will assume this for the iSTFT as well).
2. Interpolate by R (again, if $R = 1$ then this is of course unnecessary).
3. Multiplying by the synthesis window $F_k(e^{j\omega})$.
4. Summing up all the branches.

This synthesis can also be written as:



1. Multiplying by the exponent.
2. M·iDFT in length of M
3. Cyclic expansion to length of L_f.
4. Multiplying by the synthesis window of length L_f.
5. Overlap and add, Overlap and save, and other methods.

If all of the perfect reconstruction conditions apply then we should get the reconstructed signal in the time-domain, as can be seen in this wave plot for a speech signal:



In this case, there is no Time Frequency processing, this is just a signal which was transformed using the STFT and iSTFT.
As we can see, we get perfect reconstruction, proving that in this case the iSTFT is indeed the inverse of the STFT transform.

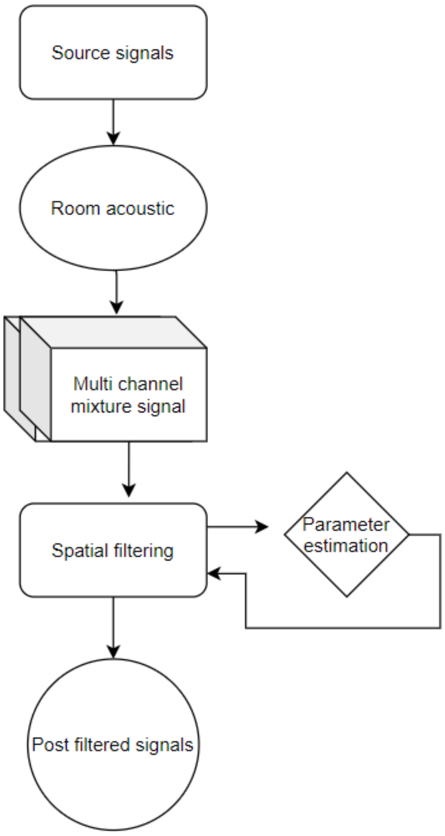
Blind Source Separation

In order to understand the method of blind source separation implemented in this project, we first must have some theoretical background on what sound is.
Sound can be thought of as the vibration that propagates as an acoustic wave, though a transmission medium such as solid, gas or liquid.

We can think of speech as a type of sound with a variation of air pressure on the order of $10^{-2}Pa$ at a distance of 1 meter.
For comparison, the average atmospheric pressure is $10^{5}Pa$, meaning that speech is a very low-pressure sort of sound.
For this low-pressure value, the wave equation which governs the propagation of sound can be estimated as linear, and this has two implications:

1. The pressure field for a given source propagates in space and time linearly.
2. The pressure field at any given point in time is the overall sum of pressure fields from each source at the same given time.

Microphones operate linearly to record the pressure value at a given point in space, therefore, the overall phenomenon is linear.
The general schema for the acoustical propagation and the BSS can be shown as:



The source signals convolve in the RIR and recorded by the multi-microphones device; the mixture goes through a certain filtering method which is the BSS which results in the post filtered signals.

A class of unsupervised methods called BSS are a class of algorithms based on the assumption that the sources are mutually independent.
BSS algorithms pursue a goal which is similar to that of beamforming: to reduce interferences.
Although, one major distinction between BSS and beamforming is that BSS does not require any information about the source's location.

Another difference between BSS and beamforming is that BSS recovers all individual sources simultaneously from the mixture, beamforming on the other hand, only concentrates on extracting one target source each time.

The BSS problem is still far from being solved.
Realistic audio mixtures are better separated by methods such as DUET, EM or hybrid models. However, the quality remains insufficient for demanding applications such as hearing aids, driving safety, etc.
There are some issues that account for this:

1. Time-Frequency overlap of the sources (meaning that the sparse assumption is not always correct).
2. Reverberation.
3. Source movement (in our project we assumed stationary sources for example, this is not always the case).
4. Similarities between source characteristics.
5. Too close of a proximity between the source directions.

Because of all these difficulties, many BSS methods have been proposed independently by researchers from different subfields for the last 30 years.
Each approach with its own limitations and emphasizes on what the method is trying to clear up between the issues and its own approach on how to make the estimated sources the clearest.

In our example, we had used the EM algorithm to estimate the sources from the mixture, with the DUET method used to initialize the EM algorithm (a bit of a hybrid model), we assumed that the sources are stationary, are not too close to each other, and most importantly, we assumed the sparse assumption.

The sparse assumption refers to the domination of TF bins by only a single source, meaning that for each TF bin in the Time-Frequency domain, only one of the sources is present and that we can indicate the source dominating the TF bin (which can be called masking).
This assumption is usually reasonable with speech because it is highly unlikely for the two sources to speak the same note (frequency wise) at the exact same time.

Sometimes, the sparse assumptions doesn't "hold water", meaning that there are still cases where more than one of the sources is dominating the same TF bin.
In this case, the resulting separation will lose the data over the TF bin by masking it as just one speaker or the algorithm will decide that both speakers dominate the said TF bin and therefore the separation will be weaker, either way, this degrades the separation's quality.
For this reason, our separation (that assumes the sparse assumption) can never truly be perfect, but it can always be improved.

The EM algorithm

The Expectation-Maximization (EM) algorithm was first explained in a classic 1977 paper by Arthur Dempster, Nan Laird and Donald Rubin. Though they pointed out that the method had been proposed many times by earlier authors.
One of the earliest is the gene counting method for estimating allele frequencies by Cedric Smith.

In our project, we use the expectation-maximization algorithm as an iterative method to find the maximum likelihood (ML) or the maximum a posteriori (MAP) estimates of the parameters in a statistical model.
The model depends on the unobserved latent data variables which we would like the estimate.

The EM iteration alternates between two steps, the E step and the M step.
In the E step, we create a function for the expectation of the log-likelihood evaluated using the current estimate for the parameters.
In the M step we compute the parameters which maximize the expected log-likelihood we created in the E step.
These parameter estimates are then used to determine the latent variables in the next E step and so on goes the cycle between the two steps.

The ultimate goal of the algorithm implemented in our project is to globally maximize the auxiliary function that will be described in the following pages.
Therefore, the initialization of the algorithm is of much importance, since not initializing the parameters correctly may result in the auxiliary function converging into a local maximum and ultimately, the failure of the algorithm.

Mathematical analysis

In the following mathematical analysis, we analyze the signal in the STFT domain, where K denotes the number of frequency bands, and T denotes the number of time frames. We assume D speech sources, received by J microphones. In this analysis we assume that the sparsity property applies – that is when each TF bin is dominated by only one source.

We denote the possible values of t, k, d as such:

$$\mathbb{m}_T \triangleq \{1, \dots, T\}$$
$$\mathbb{m}_K \triangleq \{1, \dots, K\}$$
$$\mathbb{m}_D \triangleq \{0, \dots, D\}$$

We make use of the discrete random variable $d(t,k) \in \mathbb{m}_D$ that indicates which speaker is active at the (t, k)-th bin. If $d(t,k) = 0$, then no speaker is active.
We define the set of indicators $\mathcal{D} = \{d(t,k) : t \in \mathbb{m}_T, k \in \mathbb{m}_K\}$.

We define the set of clean speech signals $\mathcal{X} \triangleq \{x_d(t,k) : t \in \mathbb{m}_T, k \in \mathbb{m}_K, d \in \mathbb{m}_D\}$, and by definition $x_0(t,k) = 0$.

We model the noise $\boldsymbol{V}(t,k)$ as a stationary additive noise with the following distribution:

$$\boldsymbol{V}(t,k) \sim \mathcal{N}_c\{0, \boldsymbol{R}_v(k)\}$$

Where $\boldsymbol{R}_v(k)$ is the time-invariant covariance matrix.

We define $\boldsymbol{h}_d(k)$ as the relative transfer function between the d-th speaker and the microphones.

With all that in mind, we can write the observation vector as

$$\boldsymbol{z}(t,k) = \begin{cases} \boldsymbol{V}(t,k) & ; d(t,k) = 0 \\ \boldsymbol{V}(t,k) + \boldsymbol{h}_1(k) \cdot x_1(t,k) & ; d(t,k) = 1 \\ \vdots & ; \quad \vdots \\ \boldsymbol{V}(t,k) + \boldsymbol{h}_D(k) \cdot x_D(t,k) & ; d(t,k) = D \end{cases}$$

Where $\boldsymbol{z}(t,k)$, $\boldsymbol{V}(t,k)$ and $\boldsymbol{h}_D(k)$ are all $J \times 1$ vectors.
We denote the set of observations as $\mathcal{Z} = \{\boldsymbol{z}(t,k) : t \in \mathbb{m}_T, k \in \mathbb{m}_K\}$, and the time invariant priori probability of the d-th hypothesis as $p_d(k) \equiv \Pr\{d(t,k) = d\}$.

In our project, we model $\mathcal{X}$ as a deterministic unknown signal, therefore it is part of the parameter set $\overline{\boldsymbol{\theta}} = \{p_d(k), \boldsymbol{h}_d(k), \boldsymbol{R}_v(k), \mathcal{X} : k \in \mathbb{m}_K,\ d \in \mathbb{m}_D\}$. And the hidden data is the association of each TF bin to one of $D+1$ classes $\overline{\mathcal{H}} = \mathcal{D}$.

Next, we would like to develop the log-likelihood function of the complete data. For that we make use of the indicator random variable $\mathbb{I}_{t,k,d}$, which equals one when $\boldsymbol{d(t,k)=d}$ and zero otherwise.

The probability density function of the complete data checks whether each one of the speakers is active at a specific (t, k)-bin

$$f(\mathcal{Z},\bar{\mathcal{H}};\bar{\boldsymbol{\theta}}) = \prod_{t,k}\sum_{d} \mathbb{I}_{t,k,d}\cdot p_d(k)\cdot f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}})$$

where

$$f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}}) = f(\boldsymbol{V}(t,k)+\boldsymbol{h}_d(k)\cdot x_d(t,k)\,;\bar{\boldsymbol{\theta}}) = \mathcal{N}_c\{\boldsymbol{h}_d(k)\cdot x_d(t,k),\boldsymbol{R}_v(k)\}$$

and the log likelihood of the complete data is

$$\log f(\mathcal{Z},\bar{\mathcal{H}};\ \bar{\boldsymbol{\theta}}) = \log\prod_{t,k}\sum_{d} \mathbb{I}_{t,k,d}\cdot p_d(k)\cdot f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}})$$

$$\log f(\mathcal{Z},\bar{\mathcal{H}};\ \bar{\boldsymbol{\theta}}) = \sum_{t,k}\log\sum_{d} \mathbb{I}_{t,k,d}\cdot p_d(k)\cdot f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}})$$

Since the indicator $\mathbb{I}_{t,k,d}$ equals one for only one value of d each time:

$$\log f(\mathcal{Z},\bar{\mathcal{H}};\ \bar{\boldsymbol{\theta}}) = \sum_{t,k,d}\mathbb{I}_{t,k,d}\log[p_d(k)\cdot f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}})]$$

$$\log f(\mathcal{Z},\bar{\mathcal{H}};\ \bar{\boldsymbol{\theta}}) = \sum_{t,k,d}\mathbb{I}_{t,k,d}[\log p_d(k)+\log f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}})]$$

Expectation step

In this part we define and develop the auxiliary function $Q\left(\bar{\boldsymbol{\theta}}\middle|\bar{\boldsymbol{\theta}}^{(l)}\right)$ which we will try to maximize in the M step

$$Q\left(\bar{\boldsymbol{\theta}}\middle|\bar{\boldsymbol{\theta}}^{(l)}\right) \triangleq E\{\log f(\mathcal{Z},\bar{\mathcal{H}};\ \bar{\boldsymbol{\theta}}) \mid \mathcal{Z};\ \bar{\boldsymbol{\theta}}^{(l)}\}$$

$$Q\left(\bar{\boldsymbol{\theta}}\middle|\bar{\boldsymbol{\theta}}^{(l)}\right) = E\left\{\sum_{t,k,d} \mathbb{I}_{t,k,d}[\log p_d(k) + \log f(\boldsymbol{z}(t,k) \mid d;\bar{\boldsymbol{\theta}})]\right\}$$

according to linearity of the expectation:

$$Q\left(\bar{\boldsymbol{\theta}}\middle|\bar{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,k,d} E\left\{\mathbb{I}_{t,k,d}|\ \mathcal{Z};\ \bar{\boldsymbol{\theta}}^{(l)}\right\}\cdot[\log p_d(k) + \log f(\boldsymbol{z}(t,k) \mid d;\bar{\boldsymbol{\theta}})]$$

$$Q\left(\bar{\boldsymbol{\theta}}\middle|\bar{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,k,d} \Pr\{d(t,k) = d|\ \mathcal{Z};\bar{\boldsymbol{\theta}}^{(l)}\}\cdot[\log p_d(k) + \log f(\boldsymbol{z}(t,k) \mid d;\bar{\boldsymbol{\theta}})]$$

We denote $\bar{w}_d^{(l)}(t) = \Pr\{d(t,k) = d|\ \mathcal{Z};\bar{\boldsymbol{\theta}}^{(l)}\}$, and according to the law of total probability

$$\bar{w}_d^{(l)}(t) = \frac{p_d^{(l)}(k)\cdot f(\boldsymbol{z}(t,k) \mid d;\bar{\boldsymbol{\theta}}^{(l)})}{\sum_{d'\in\mathbb{m}_D} p_{d'}^{(l)}(k)\cdot f(\boldsymbol{z}(t,k) \mid d';\bar{\boldsymbol{\theta}}^{(l)})}$$

$$Q\left(\bar{\boldsymbol{\theta}}\middle|\bar{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,k,d} \bar{w}_d^{(l)}(t)\cdot[\log p_d + \log f(\boldsymbol{z}(t,k) \mid d;\bar{\boldsymbol{\theta}})]$$

Maximization step

In this section we try to update the parameter set which will maximize the auxiliary function

$$\overline{\boldsymbol{\theta}}^{(l+1)} = \underset{\overline{\boldsymbol{\theta}}}{\operatorname{argmax}}\, Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right)$$

The probability density function of a complex gaussian random variable x with expectation vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{C}_x$ is

$$X\sim\mathcal{N}_c\{\boldsymbol{\mu},\boldsymbol{C}_x\} \Rightarrow f(x) = \frac{1}{|\boldsymbol{C}_x|} e^{-(x-\boldsymbol{\mu})^H \boldsymbol{C}_x^{-1}(x-\boldsymbol{\mu})}$$

Therefore, we can write

$$\log f(\boldsymbol{z}(t,k) \mid d;\overline{\boldsymbol{\theta}}) = \log\left[\frac{1}{|\boldsymbol{R}_v(k)|} e^{-\left(\boldsymbol{z}(t,k)-\boldsymbol{h}_d(k)\cdot x_d(t,k)\right)^H \boldsymbol{R}_v^{-1}(k)\left(\boldsymbol{z}(t,k)-\boldsymbol{h}_d(k)\cdot x_d(t,k)\right)}\right]$$

And by defining

$$\boldsymbol{e}_d(t,k) = \begin{cases}\boldsymbol{z}(t,k) & ;d=0\\ \boldsymbol{z}(t,k)-\boldsymbol{h}_d(k)\cdot x_d(t,k) & ;d>0\end{cases}$$

$$\log f(\boldsymbol{z}(t,k) \mid d;\overline{\boldsymbol{\theta}}) = \log\left[\frac{1}{|\boldsymbol{R}_v(k)|} e^{-\boldsymbol{e}_d(t)^H \boldsymbol{R}_v^{-1}(k)\boldsymbol{e}_d(t)}\right]$$

$$\log f(\boldsymbol{z}(t,k) \mid d;\overline{\boldsymbol{\theta}}) = -\log|\boldsymbol{R}_v| - \boldsymbol{e}_d(t)^H \boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)$$

Substituting this into the auxiliary function will result in

$$Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,d} \overline{w}_d^{(l)}(t)\cdot\left[\log p_d - \log|\boldsymbol{R}_v| - \boldsymbol{e}_d(t)^H \boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)\right]$$

We begin by maximizing the auxiliary function with respect to $x_d^*(t)$

$$\nabla_{x_d^*(t)} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \nabla_{x_d^*(t)} \sum_{t,d} \overline{w}_d^{(l)}(t)\cdot\left[\log p_d - \log|\boldsymbol{R}_v| - \boldsymbol{e}_d(t)^H \boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)\right]$$

According to the linearity of the derivative

$$\nabla_{x_d^*(t)} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \overline{w}_d^{(l)}(t)\cdot\nabla_{x_d^*(t)}\left[\log p_d - \log|\boldsymbol{R}_v|\right] - \overline{w}_d^{(l)}(t)\cdot\nabla_{x_d^*(t)}\left[\boldsymbol{e}_d(t)^H \boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)\right]$$

Since $\nabla(const) = 0$

$$\nabla_{x_d^*(t)} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = -\overline{w}_d^{(l)}(t)\cdot\ \nabla_{x_d^*(t)}\left[\left(\boldsymbol{z}(t)-\boldsymbol{h}_d\cdot x_d(t)\right)^H \boldsymbol{R}_v^{-1}\left(\boldsymbol{z}(t)-\boldsymbol{h}_d\cdot x_d(t)\right)\right]$$

$$\nabla_{x_d^*(t)} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \overline{w}_d^{(l)}(t)\cdot\ \nabla_{x_d^*(t)}\left[\boldsymbol{h}_d^H\cdot x_d^*(t) - \boldsymbol{z}^H(t)\right]\cdot\boldsymbol{R}_v^{-1}\left(\boldsymbol{z}(t)-\boldsymbol{h}_d\cdot x_d(t)\right)$$

$$\nabla_{x_d^*(t)} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \overline{w}_d^{(l)}(t)\cdot\ \boldsymbol{h}_d^H\cdot\boldsymbol{R}_v^{-1}\left(\boldsymbol{z}(t)-\boldsymbol{h}_d\cdot x_d(t)\right)$$

Equating to zero

$$\bar{w}_d^{(l)}(t)\cdot\ \boldsymbol{h}_d^H\cdot\boldsymbol{R}_v^{-1}\big(\boldsymbol{z}(t)-\boldsymbol{h}_d\cdot x_d(t)\big)=0$$

$$\boldsymbol{h}_d^H\cdot\boldsymbol{R}_v^{-1}\cdot\boldsymbol{z}(t)=\boldsymbol{h}_d^H\cdot\boldsymbol{R}_v^{-1}\cdot\boldsymbol{h}_d\cdot x_d(t)$$

$$x_d(t)=\frac{\boldsymbol{h}_d^H\boldsymbol{R}_v^{-1}}{\boldsymbol{h}_d^H\boldsymbol{R}_v^{-1}\boldsymbol{h}_d}\cdot\boldsymbol{z}(t)$$

We denote

$$\bar{\boldsymbol{u}}_d(\bar{\boldsymbol{\theta}})=\frac{\boldsymbol{h}_d^H\boldsymbol{R}_v^{-1}}{\boldsymbol{h}_d^H\boldsymbol{R}_v^{-1}\boldsymbol{h}_d}\ \Rightarrow\ \bar{x}_d^{(l+1)}(t)=\bar{\boldsymbol{u}}_d(\bar{\boldsymbol{\theta}})\cdot\boldsymbol{z}(t)$$

Next, we maximize the auxiliary function with respect to p_d under the constraint $\sum_d p_d=1$, using the Lagrangian, and since the addition to the auxiliary function equals zero, the value that maximizes the Lagrangian, also maximizes the auxiliary function.

$$\mathcal{L}(\bar{\boldsymbol{\theta}})=Q\big(\bar{\boldsymbol{\theta}}|\bar{\boldsymbol{\theta}}^{(l)}\big)+\lambda\left(1-\sum_d p_d\right)$$

$$\nabla_{p_d}\mathcal{L}(\bar{\boldsymbol{\theta}})=\nabla_{p_d}\left\{\sum_{t,d}\left\{\bar{w}_d^{(l)}(t)\cdot[\log p_d-\log|\boldsymbol{R}_v|-\boldsymbol{e}_d(t)^H\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)]\right\}+\lambda\left(1-\sum_d p_d\right)\right\}$$

$$\nabla_{p_d}\mathcal{L}(\bar{\boldsymbol{\theta}})=\sum_t\left\{\nabla_{p_d}\left[\bar{w}_d^{(l)}(t)\log p_d\right]-\nabla_{p_d}[\log|\boldsymbol{R}_v|+\boldsymbol{e}_d(t)^H\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)]\right\}$$
$$+\nabla_{p_d}\lambda\left(1-\sum_d p_d\right)$$

$$\nabla_{p_d}\mathcal{L}(\bar{\boldsymbol{\theta}})=\sum_t\left\{\nabla_{p_d}\bar{w}_d^{(l)}(t)\cdot\log p_d+\bar{w}_d^{(l)}(t)\cdot\nabla_{p_d}\log p_d\right\}+\lambda\nabla_{p_d}(1-p_d)$$

$$\nabla_{p_d}\mathcal{L}(\bar{\boldsymbol{\theta}})=\sum_t\left\{\nabla_{p_d}\bar{w}_d^{(l)}(t)\cdot\log p_d+\bar{w}_d^{(l)}(t)\cdot\nabla_{p_d}\log p_d\right\}-\lambda$$

The derivative of posteriori probability of the d-th hypothesis with respect to p_d is

$$\nabla_{p_d}\bar{w}_d^{(l)}(t)=\nabla_{p_d}\frac{p_d^{(l)}(k)\cdot f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}}^{(l)})}{\sum_{d'\in\mathbb{m}_D}p_{d'}^{(l)}(k)\cdot f(\boldsymbol{z}(t,k)\mid d';\bar{\boldsymbol{\theta}}^{(l)})}$$

$$\nabla_{p_d}\bar{w}_d^{(l)}(t)$$
$$=\frac{f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}}^{(l)})\cdot p_d^{(l)}(k)\cdot f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}}^{(l)})-p_d^{(l)}(k)\cdot f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}}^{(l)})f(\boldsymbol{z}(t,k)\mid d;\bar{\boldsymbol{\theta}}^{(l)})}{\left(\sum_{d'\in\mathbb{m}_D}p_{d'}^{(l)}(k)\cdot f(\boldsymbol{z}(t,k)\mid d';\bar{\boldsymbol{\theta}}^{(l)})\right)^2}$$
$$=0$$

Going back to the derivative of the Lagrangian function, we get

$$\nabla_{p_d}\mathcal{L}(\overline{\boldsymbol{\theta}}) = \sum_t \frac{\overline{w}_d^{(l)}(t)}{p_d} - \lambda = \frac{1}{p_d}\sum_t \overline{w}_d^{(l)}(t) - \lambda$$

Equating to zero

$$\frac{1}{p_d}\sum_t \overline{w}_d^{(l)}(t) - \lambda = 0 \;\Rightarrow\; p_d^{(l+1)} = \frac{1}{\lambda}\sum_t \overline{w}_d^{(l)}(t)$$

Since $\overline{w}_d^{(l)}(t) \in [0,1]$, we chose $\lambda = T$ in order to normalize p_d and by so making $p_d \in [0,1]$.

We move on for the maximization of the auxiliary function with respect to $\boldsymbol{h}_d^H$

$$\nabla_{\boldsymbol{h}_d^H} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \nabla_{\boldsymbol{h}_d^H} \sum_{t,d} \overline{w}_d^{(l)}(t) \cdot \left[\log p_d - \log|\boldsymbol{R}_v| - \boldsymbol{e}_d(t)^H \boldsymbol{R}_v^{-1} \boldsymbol{e}_d(t)\right]$$

$$\nabla_{\boldsymbol{h}_d^H} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \sum_t \overline{w}_d^{(l)}(t) \cdot \nabla_{\boldsymbol{h}_d^H}\left[\log p_d - \log|\boldsymbol{R}_v|\right] - \overline{w}_d^{(l)}(t) \cdot \nabla_{\boldsymbol{h}_d^H}\left[\boldsymbol{e}_d(t)^H \boldsymbol{R}_v^{-1} \boldsymbol{e}_d(t)\right]$$

$$\nabla_{\boldsymbol{h}_d^H} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = -\overline{w}_d^{(l)}(t) \cdot\ \nabla_{\boldsymbol{h}_d^H}\left[\left(\boldsymbol{z}(t) - \boldsymbol{h}_d \cdot x_d(t)\right)^H \boldsymbol{R}_v^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{h}_d \cdot x_d(t)\right)\right]$$

$$\nabla_{\boldsymbol{h}_d^H} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \overline{w}_d^{(l)}(t) \cdot\ \nabla_{\boldsymbol{h}_d^H}\left[\boldsymbol{h}_d^H \cdot x_d^*(t) - \boldsymbol{z}^H(t)\right] \cdot \boldsymbol{R}_v^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{h}_d \cdot x_d(t)\right)$$

$$\nabla_{\boldsymbol{h}_d^H} Q\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \sum_t \overline{w}_d^{(l)}(t) x_d^*(t) \boldsymbol{R}_v^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{h}_d \cdot x_d(t)\right)$$

Equating to zero results in

$$\sum_t \overline{w}_d^{(l)}(t) x_d^*(t) \boldsymbol{R}_v^{-1}\left(\boldsymbol{z}(t) - \boldsymbol{h}_d \cdot x_d(t)\right) = 0$$

$$\sum_t \overline{w}_d^{(l)}(t) x_d^*(t) \boldsymbol{R}_v^{-1} \boldsymbol{z}(t) - \overline{w}_d^{(l)}(t) x_d^*(t) \boldsymbol{R}_v^{-1} \boldsymbol{h}_d x_d(t) = 0$$

$$\boldsymbol{R}_v^{-1} \sum_t \overline{w}_d^{(l)}(t) x_d^*(t) \boldsymbol{z}(t) - \overline{w}_d^{(l)}(t) |x_d(t)|^2 \boldsymbol{h}_d = 0$$

$$\boldsymbol{h}_d \sum_t \overline{w}_d^{(l)} |x_d(t)|^2 = \sum_t \overline{w}_d^{(l)}(t) x_d^*(t) \boldsymbol{z}(t)$$

$$\boldsymbol{h}_d^{(l+1)} = \frac{\sum_t \overline{w}_d^{(l)}(t) \left[x_d^{(l+1)}(t)\right]^* \boldsymbol{z}(t)}{\sum_t \overline{w}_d^{(l)} \left|x_d^{(l+1)}(t)\right|^2}$$

And finally, we maximize the auxiliary function with respect to $\boldsymbol{R}_v$:

$$\nabla_{\boldsymbol{R}_v}\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \nabla_{\boldsymbol{R}_v}\sum_{t,d}\overline{w}_d^{(l)}(t)\cdot[\log p_d - \log|\boldsymbol{R}_v| - \boldsymbol{e}_d(t)^H\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)]$$

$$\nabla_{\boldsymbol{R}_v}\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,d}\overline{w}_d^{(l)}(t)\cdot\nabla_{\boldsymbol{R}_v}\log p_d + \overline{w}_d^{(l)}(t)\cdot\nabla_{\boldsymbol{R}_v}[-\boldsymbol{e}_d(t)^H\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t) - \log|\boldsymbol{R}_v|]$$

$$\nabla_{\boldsymbol{R}_v}\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,d}\overline{w}_d^{(l)}(t)\cdot\nabla_{\boldsymbol{R}_v}[-\boldsymbol{e}_d(t)^H\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t) - \log|\boldsymbol{R}_v|]$$

From the identities:

$$\frac{\partial\boldsymbol{a}^H\boldsymbol{X}^{-1}\boldsymbol{b}}{\partial\boldsymbol{X}} = \boldsymbol{X}^{-T}\boldsymbol{a}\boldsymbol{b}^H\boldsymbol{X}^{-T}$$

$$\frac{\partial\log|\boldsymbol{X}|}{\partial\boldsymbol{X}} = \boldsymbol{X}^{-1}$$

We can write

$$\nabla_{\boldsymbol{R}_v}\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,d}\overline{w}_d^{(l)}(t)\cdot[\boldsymbol{R}_v^{-T}\boldsymbol{e}_d(t)\boldsymbol{e}_d^H(t)\boldsymbol{R}_v^{-T} - \boldsymbol{R}_v^{-1}]$$

Since covariance matrix is symmetric

$$\nabla_{\boldsymbol{R}_v}\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \sum_{t,d}\overline{w}_d^{(l)}(t)\cdot[\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)\boldsymbol{e}_d^H(t)\boldsymbol{R}_v^{-1} - \boldsymbol{R}_v^{-1}]$$

$$\nabla_{\boldsymbol{R}_v}\left(\overline{\boldsymbol{\theta}}\middle|\overline{\boldsymbol{\theta}}^{(l)}\right) = \left\{\sum_{t,d}\overline{w}_d^{(l)}(t)\cdot[\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)\boldsymbol{e}_d^H(t) - \boldsymbol{I}]\right\}\boldsymbol{R}_v^{-1}$$

Equating to zero

$$\left\{\sum_{t,d}\overline{w}_d^{(l)}(t)\cdot[\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)\boldsymbol{e}_d^H(t) - \boldsymbol{I}]\right\}\boldsymbol{R}_v^{-1} = 0$$

$$\sum_{t,d}\overline{w}_d^{(l)}(t)\boldsymbol{R}_v^{-1}\boldsymbol{e}_d(t)\boldsymbol{e}_d^H(t) = \sum_{t,d}\overline{w}_d^{(l)}(t)\boldsymbol{I}$$

Since $\sum_d\overline{w}_d^{(l)}(t) = 1$

$$T\cdot\boldsymbol{I} = \boldsymbol{R}_v^{-1}\sum_{t,d}\overline{w}_d^{(l)}(t)\boldsymbol{e}_d(t)\boldsymbol{e}_d^H(t)$$

$$\boldsymbol{R}_v^{(l+1)} = \frac{1}{T}\sum_{t,d}\overline{w}_d^{(l+1)}(t)\boldsymbol{e}_d(t)\left[\boldsymbol{e}_d^{(l+1)}(t)\right]^H$$

Summary of the notation:

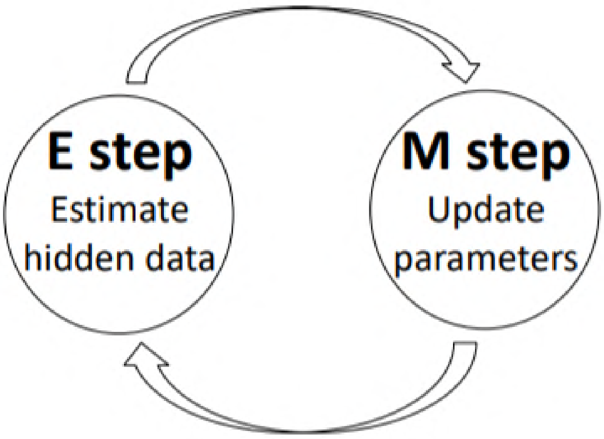
Symbol	Definition	Value
$d(t)$	d-th hypothesis indicator	
$x_d(t)$	d-th source signal	
$v_j(t)$	j-th source signal	
$z_j(t)$	j-th microphone signal	
$\boldsymbol{v}(t)$	Noise vector	$[v_0(t), \dots, v_{J-1}(t)]^T$
$\boldsymbol{z}(t)$	Observation vector	$[z_0(t), \dots, z_{J-1}(t)]^T$
p_d	d-th a priori probability	
$\boldsymbol{h}_d$	d-th steering vector	$[h_{d,0}, \dots, h_{d,J-1}]^T$
$\boldsymbol{R}_v$	Noise covariance	
$\mathcal{Z}$	Observation set	$\{z(t) : t \in \mathbb{m}_T\}$
$\mathcal{D}$	Indicator set	$\mathcal{D} = \{d(t,k) : t \in \mathbb{m}_T, k \in \mathbb{m}_K\}$
$\mathcal{X}$	Speech STFT coefficients	$\mathcal{X} \triangleq \{x_d(t,k) : t \in \mathbb{m}_T, k \in \mathbb{m}_K, d \in \mathbb{m}_D\}$
$\boldsymbol{\theta}$	Common parameter set	$\{p_d(k), \boldsymbol{h}_d(k), \boldsymbol{R}_v(k), \mathcal{X} : k \in \mathbb{m}_K, \quad d \in \mathbb{m}_D\}$

It is worth mentioning that sometimes we dropped out the dependence of k from the parameters, but this is only for aesthetic reasons. All of the parameters in the algorithm are dependent on k (frequency).

Summary of the algorithm:

Notation	Value
$f(z(t)\|d;\bar{\theta})$	$\mathcal{N}_c\{z(t); h_d \cdot x_d(t), \boldsymbol{R}_v\}$
$\bar{w}_d^{(l)}(t)$	$\frac{p_d^{(l)}(k) \cdot f(\boldsymbol{z}(t,k) \mid d; \bar{\boldsymbol{\theta}}^{(l)})}{\sum_{d' \in \mathbb{m}_D} p_{d'}^{(l)}(k) \cdot f(\boldsymbol{z}(t,k) \mid d'; \bar{\boldsymbol{\theta}}^{(l)})}$
$\bar{x}_d^{(l+1)}(t)$	$\left(\bar{u}_d^{(l)}\right)^H \cdot z(t)$
$\bar{u}_d^{(l)}$	$\frac{\boldsymbol{h}_d^H \boldsymbol{R}_v^{-1}}{\boldsymbol{h}_d^H \boldsymbol{R}_v^{-1} \boldsymbol{h}_d}\|_{\bar{\theta}=\bar{\theta}^{(l)}}$
$\bar{p}_d^{(l+1)}$	$\frac{1}{T}\sum_t \bar{w}_d^{(l)}(t)$
$\bar{h}_d^{(l+1)}$	$\frac{\sum_t \bar{w}_d^{(l)}(t)\left[x_d^{(l+1)}(t)\right]^* \boldsymbol{z}(t)}{\sum_t \bar{w}_d^{(l)}\left\|x_d^{(l+1)}(t)\right\|^2}$
$\boldsymbol{R}_v^{(l+1)}$	$\frac{1}{T}\sum_{t,d} \bar{w}_d^{(l+1)}(t)\boldsymbol{e}_d(t)\left[\boldsymbol{e}_d^{(l+1)}(t)\right]^H$

Overall we can present the EM scheme as such:



Initialization

The initialization of the algorithm's parameters in an important block in any application of the EM algorithm, this is because, as with any other nonlinear optimization strategy, the EM algorithm does not guarantee convergence to a global maximum.

If the auxiliary function convergences into a local maximum, the separation of sources can be weak, if not nonexistent at all.
Therefore, providing an appropriate initialization of the parameters is very important.

To initialize the EM algorithm in our project we had used the DUET algorithm which separates the sources weakly, then we had used the least squares estimation method in order to estimate the parameter h_d from the sources.

We initialized p_d as 1 for $d \in [0,2]$ (this is to include the silences hypothesis – that maybe no speaker is dominating in one's TF bin, and for this case $d = 0$), and R_v is initialized as I_{JxJ} for every k.

Using the DUET algorithm, we weakly separate the sources from the beginning, this way we initialize x_1, x_2.
Since we have $x_{1,2}$ we can also initialize h_1, h_2 using the least squares estimation method.

The least squares estimation method is an approach in regression analysis in order to approximate the solution of overdetermined systems by minimizing the sum of squares of the residuals (the difference between the observed value Z and the fitted value $x_{1,2}$ provided by the DUET algorithm).

The DUET algorithm

As we have mentioned, the initialization of separation between the sources is implemented with the DUET algorithm.

The DUET blind source separation method can separate any number of sources (in our case 2) using only two mixtures.
The method is valid when sources are W-disjoint orthogonal, meaning, that the supports of the windowed Fourier Transform of our signals (in the mixture) are disjoint.

The DUET algorithm is especially suited for speech separation, since the representation of speech in the STFT domain is sparse, thus, guaranteeing a W-disjoint orthogonality.

The DUET algorithm allows the estimation of the mixing parameters by clustering relative attenuation delay pairs extracted from the ratios of the TF representation of the mixtures, then, estimation of the mixing parameters is used to partition the TF representation of the mixture to recover the original sources.

Some assumptions are required for the DUET BSS method, listing these assumptions we can write:

1. Anechoic Mixing – Consider the mixtures of 2 source signals being received at a pair of microphones where only the direct path is present. In this case, we can absorb the attenuation and delay parameters of the first mixture $x_1(t)$ into the definition of the sources. The two anechoic mixtures can thus be expressed as: $x_1(t) = \sum_{j=1}^{2} s_j(t)$, $x_2(t) = \sum_{j=1}^{2} a_j s_j\big(t-\delta_j\big)$, where s_1, s_2 are the original sources, δ_j is the arrival delay between the sensors and a_j is the relative attenuation factor corresponding to the ration of attenuation of paths between sources and sensors.
2. W-disjoint Orthogonality – This assumption is the mathematical idealization of the condition that it is very likely for every TF bin in the STFT domain to be dominated by only one source.
   Since we assume the sparse assumption for the entire EM algorithm, the W-disjoint orthogonality assumption is relatively reasonable.
3. Local Stationarity – It is necessary for the DUET algorithm that:
   $$F^W\big[s_j(\cdot\ -\delta)\big](\omega,\tau) = e^{-i\omega\delta}F^W\big[s_j(\cdot)\big](\omega,\tau), \forall\delta, |\delta| \le \Delta$$
4. Microphones close together – It is crucial that the microphones are close together, since the DUET is based on the extraction of attenuation and delay mixing parameters for each TF bin, we can utilize the local stationary assumption to turn the delay in time into multiplicative factor in the STFT domain.

We require $|\omega\delta_j| < \pi, \forall\omega\forall j$, this is guaranteed when the microphones are separated by less than $\frac{\pi c}{\omega_m}$, where ω_m is the maximum frequency present in the sources and c is the speed of sound.
Since we recorded at 16kHz then the microphones must be placed within approximately 1cm of each other.

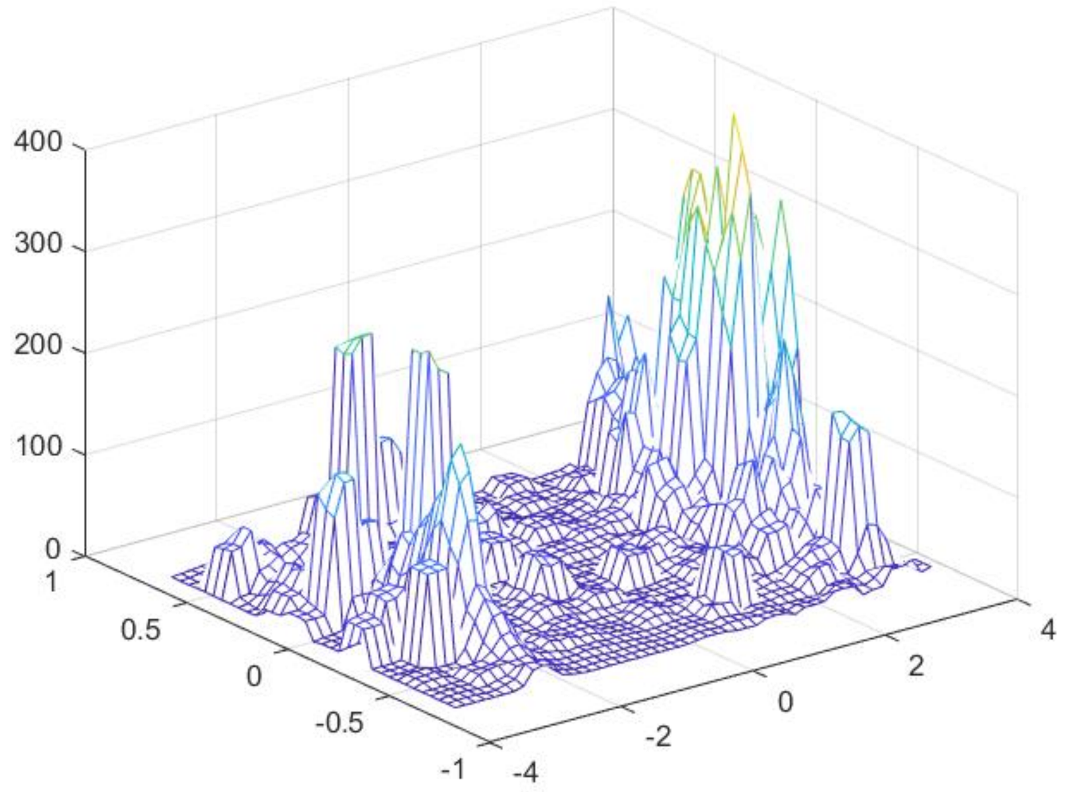
5. Different Spatial Signatures – If two sources have identical spatial signatures, meaning, that they have identical relative attenuation and delay mixing parameters, then they might be combined into one source.
In this case, the DUET method will fail to separate them from each other, so for the DUET to successfully separate the two sources, they must have different spatial signatures.

Because of all these assumptions, the DUET algorithm does not successfully separate sources in every environment.
In our project, the environment of which we record our mixture is a noisy environment, with random noise (which isn't gaussian), and a lot of echo, sometimes even $T_{60} = 610ms$.
Some assumption in this case do not exist in the way the DUET requires, for example, the anechoic mixture assumption, does not exist in a reverberant environment.

For these reasons, the DUET algorithm does not give us the result we want, but it is still useful to initialize of EM algorithm, which will take the results of the DUET algorithm and maximize them.

Even though the DUET itself does not always work properly, it still separates the two sources weakly, but this weak separation in of itself is enough for the initialization of the EM algorithm, which is stronger than the DUET algorithm and requires less assumptions.

The way that the DUET works is that first we run the DUET algorithm on the mixture and get a histogram of the probability of 2 sources:



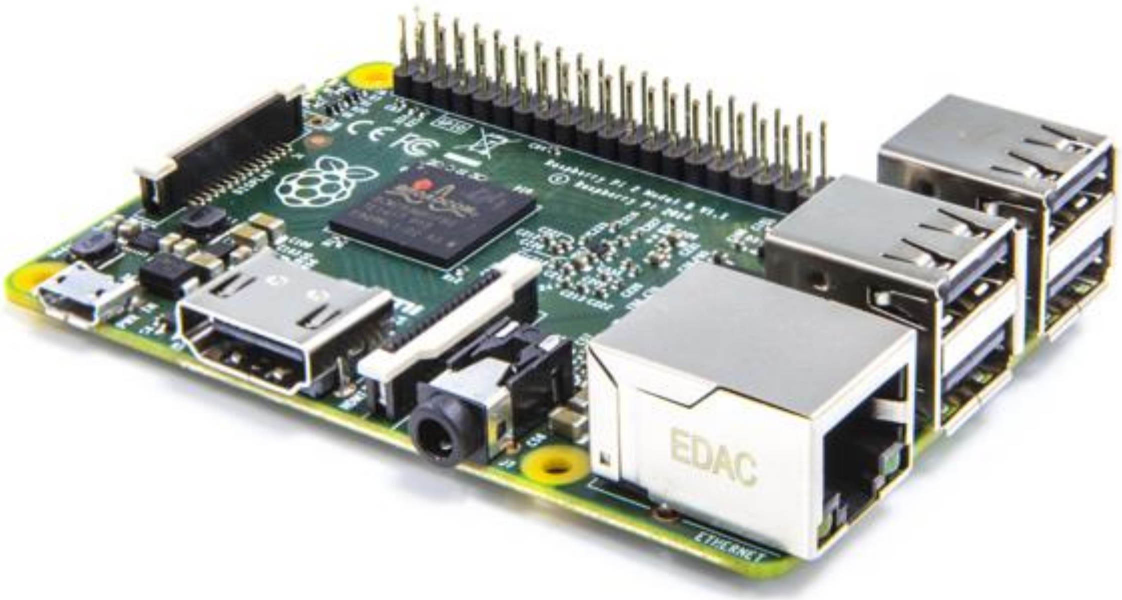
We must pick the 2 highest peaks in the histogram (right side and left) in order to initialize the best parameters for the EM algorithm.

We implemented a simple script that does it for us, so we don’t have to manually do it every time.
Then, we send the initialized parameters into the EM algorithm and start the iterative method of BSS.
After 15-25 iteration we can draw out the two sources and review the results.

In summary, the EM algorithm and its output depend on many factors such as the estimation criterion (ML or MAP – in our case ML) and the choice of the latent data.
The auxiliary function updates with each iteration and it is important that it converges into a global maximum.
The parameter initialization and the number of iterations is very important if we want to correctly converge the auxiliary function into said maximum.
We will discuss more on the sensitivity of the algorithm in the result's chapter.

Hardware

The hardware used in this project is a Raspberry pi 3 connected to a MATRIX Voice 8 MEMs microphone array as the recording device, this is connected to a personal LAPTOP which runs the MATLAB program to separate the sources.



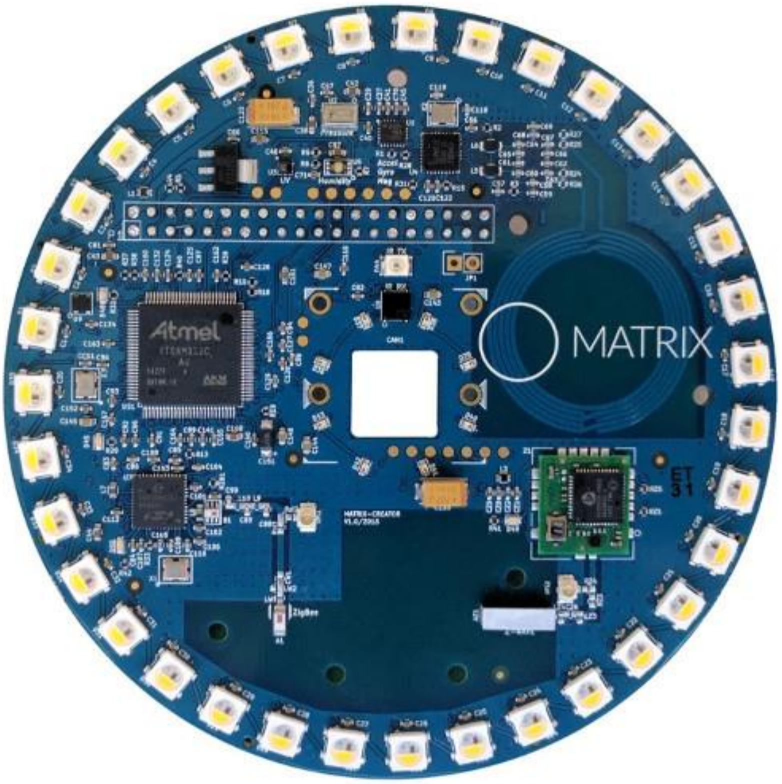
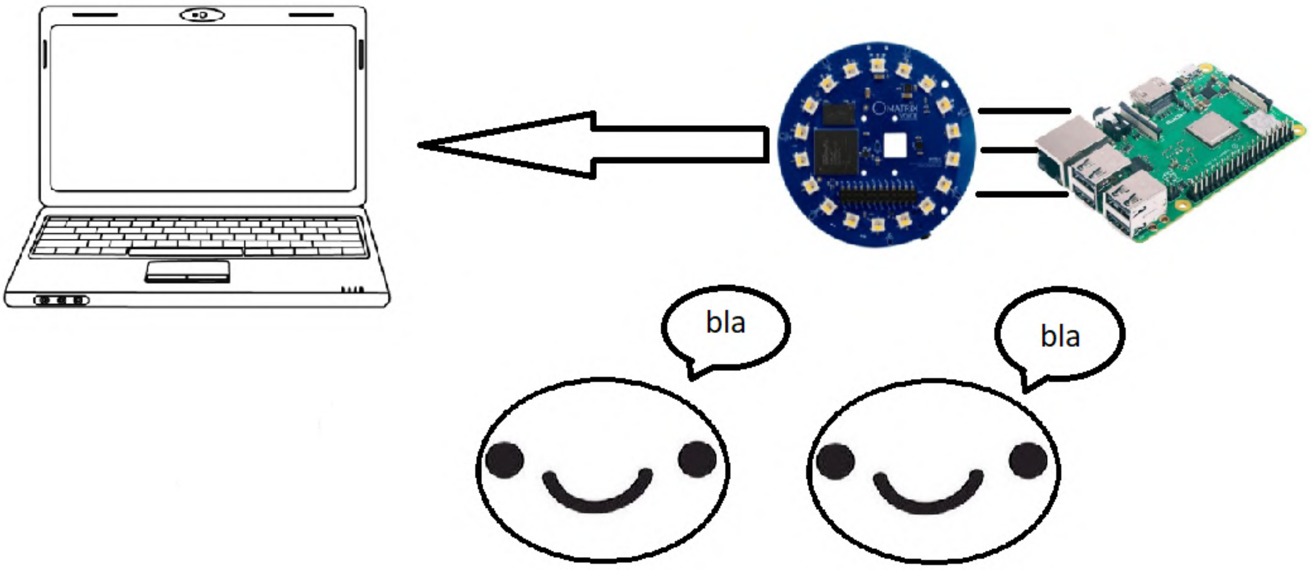


Illustration of the system can be shown as:



(We are not the best painters) 😊

After the data is sent to the computer the algorithm separates the two speakers and we can estimate the original signals.

Form of Work

In this chapter we wanted to discuss the way in which we approached the problem. Our way to approach the project, advised by Mr. Aviad Eisenberg was as so:

1. Literature review – At first, we weren't familiar at all with the BSS world, we had received some papers and articles about the EM algorithm and source separation in general and started learning about the BSS world.
2. First attempts in coding the algorithm – total failure, it was very hard to face the coding challenge hands on, so we eased our way into it with a couple of reliefs:
    1) R_v is constant and initialized in the initialization stage of the algorithm, meaning that we assume a certain noise and do not try to estimate it.
    2) h_d is a simple delta function with delays between the different microphones, to apply this we used a RIR generator function in cpp, in this case there is no reverb at all, and the sources are echo free.
    3) DUET algorithm is used to initialize the source separation in the initialization stage, meaning that we don't initialize the algorithm with eigenvalues and eigenvectors as intended in the paper.

    After many problems with stabilizing the algorithm, we successfully managed to separate the sources under this conditions, next we will try and remove these assumptions to make the algorithm more robust.
3. Running the algorithm on actual RIRs recorded at Bar Ilan University, meaning that we use actual rooms and leave behind the RIR generator function.
4. Estimating R_v as real noise, and not assuming the noise covariance matrix is constant.
5. We ended up keeping the DUET algorithm as the initialization of the EM algorithm since initializing with eigenvectors was too weak and did not work properly for our case.
6. Work on the hardware.
7. Running the algorithm on the hardware, recording through MATRIX Voice and RPI and sending the data directly to the algorithm, separating the sources in real, live environments.
8. Results and documentation.

Results

In this chapter, we will be discussing the results and quality of the algorithm. Since this is an audio separation algorithm we think that nothing beats actually hearing the results, for this reason we have made a demonstration video of the project, this video can be found in the following link:
https://www.youtube.com/watch?v=3byVhd68VZQ

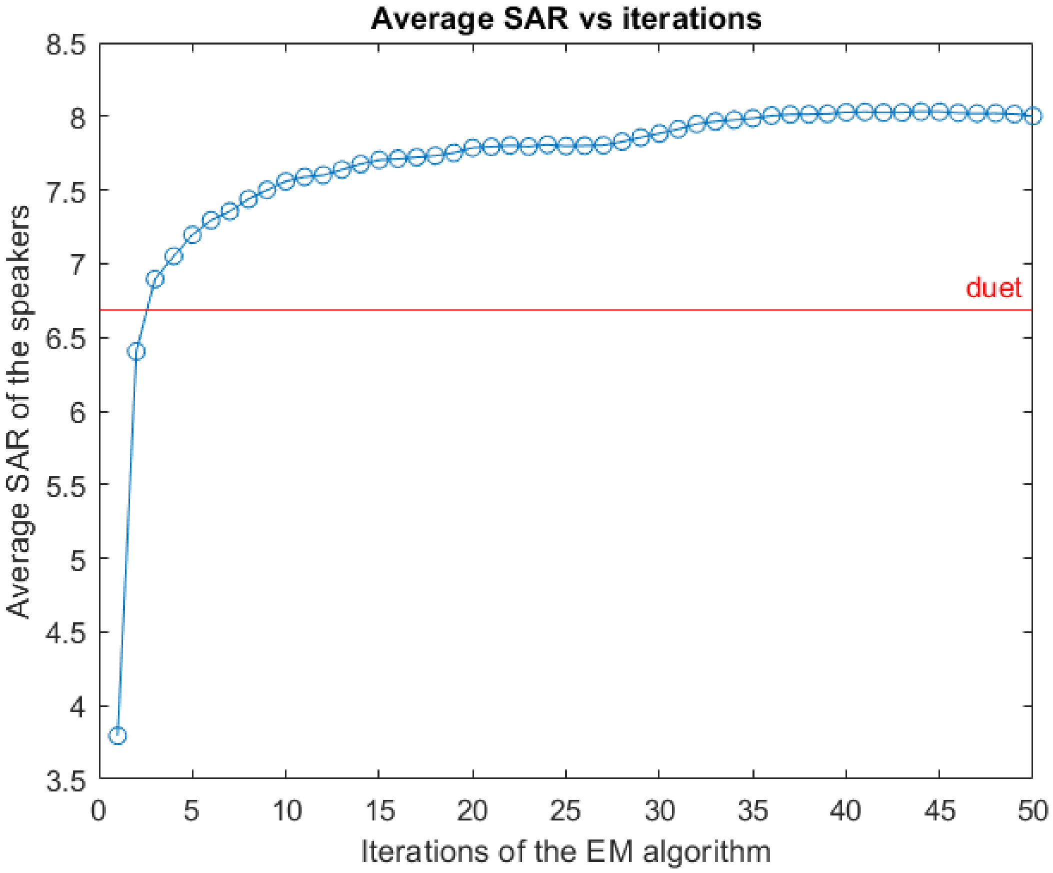
In terms of evaluating the algorithm, there are some mathematical assessments which describe the separation quality of the algorithm, this measures are:

1. Signal Interference Ratio (SIR) – is the quotient between the average received modulated carrier power and the average received co-channel interference power.
   The SIR can be calculated as:
   $$\mathrm{SIR} \equiv \mathrm{QM}\{s_{d,d}, s_{d,d'}\} = \frac{\sum_{t.k} |s_{d,d}(t,k)|^2}{\sum_{t,k} |s_{d,d'}(t,k)|^2} = \frac{\sum_{t.k} \left| \left[w_d^{(l)}(t) \cdot u_d^{(l)}(t) \right]^H \cdot [h_d(k) \cdot x_d(t,k)] \right|^2}{\sum_{t.k} \left| \left[w_d^{(l)}(t) \cdot u_d^{(l)}(t) \right]^H \cdot [h_{d'}(k) \cdot x_{d'}(t,k)] \right|^2}$$
   Where $d, d' \in \{1,2\}$.
2. Signal to Distortion Ratio (SDR) – is the measure of the quality of a signal defined as:
   $$SDR^{out} \equiv \mathrm{QM}\{s_{d,d}, x_d(t,k) - s_{d,d}\} = \frac{\sum_{t.k} |s_{d,d}(t,k)|^2}{\sum_{t,k} |x_d(t,k) - s_{d,d}(t,k)|^2} = \frac{\sum_{t.k} \left| \left[w_d^{(l)}(t) \cdot u_d^{(l)}(t) \right]^H \cdot [h_d(k) \cdot x_d(t,k)] \right|^2}{\sum_{t.k} \left| x_d(t,k) - \left[w_d^{(l)}(t) \cdot u_d^{(l)}(t) \right]^H \cdot [h_d(k) \cdot x_d(t,k)] \right|^2}$$
   Where $d, d' \in \{1,2\}$ and $SDR^{in} = \infty$.
3. Signal to Artifact Ratio (SAR) – is the measure of the ratio between the signal and artifacts which is often defined as:
   $$\mathrm{SAR} \equiv 10 \log_{10} \frac{\left|\left|e_{signal}\right|\right|^2}{\left|\left|e_{artif}\right|\right|^2}$$
   Where e_s, e_a are the signal and artifacts energy respectively.
4. Mean Square Error (MSE) – is the measure of the average of squares of errors, that is, the average squared difference between the estimated source and the actual source, it is often defined as:
   $$\mathrm{MSE} \equiv \frac{1}{N} \sum_{i=1}^{N} \left(S_i - \hat{S}_i\right)^2$$

To evaluate these measurements (SAR,SDR,SIR), we used the BSS_eval toolkit which requires both the estimated sources and the actual sources themselves. Then, we ran the evaluations for each iteration of the EM algorithm to monitor the progress of the algorithm for each step.

The following graphs show the average estimation (SAR,SDR,SIR) for the two sources as a function of the number of iterations for the EM algorithm.
The red line is the comparison of the same estimation for the DUET algorithm, meaning that the red line is the starting point after the initialization of the algorithm.

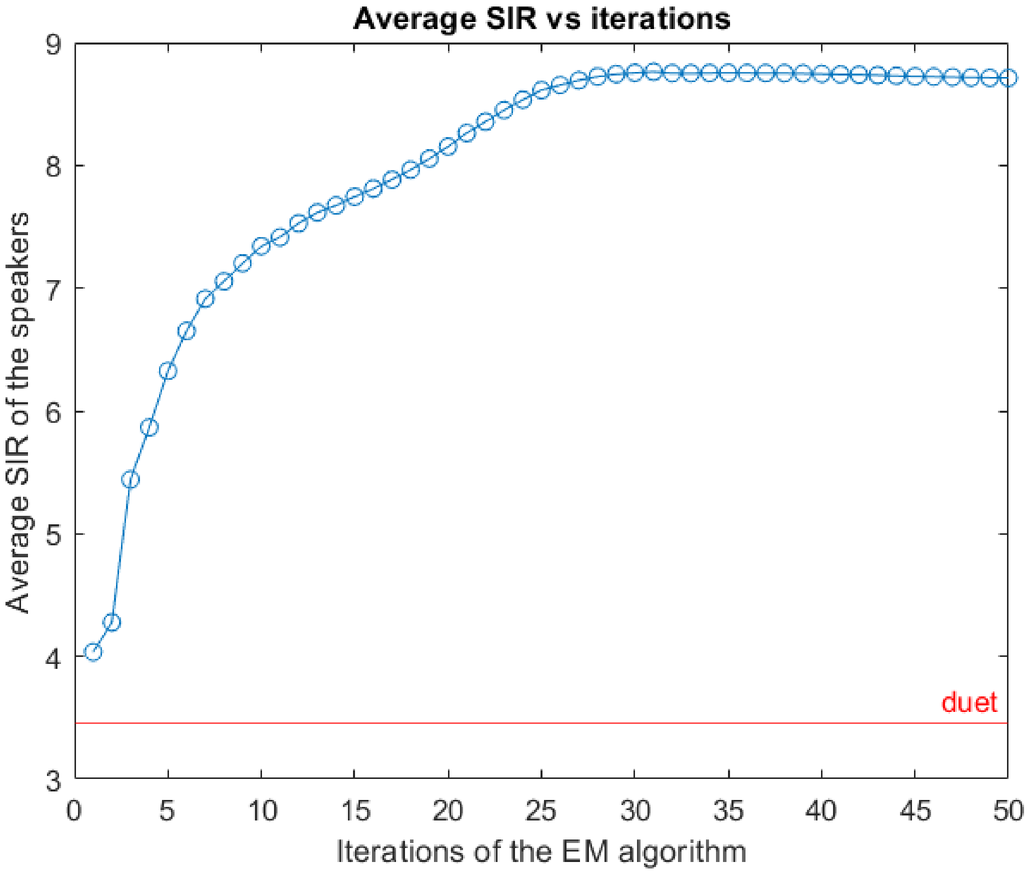
SAR evaluation:



We can see that the starting point is ≅ 6.7 and is actually lower for the first 2 iterations, but after about 28-30 iterations of the algorithm, the average SAR stabilizes at about 8, higher than the starting SAR in 1.3;

The DUET's result in this case is not bad at all means but can still be improved. The EM algorithm successfully exceeds said evaluation and maximizes it to its extent.

SIR evaluation:

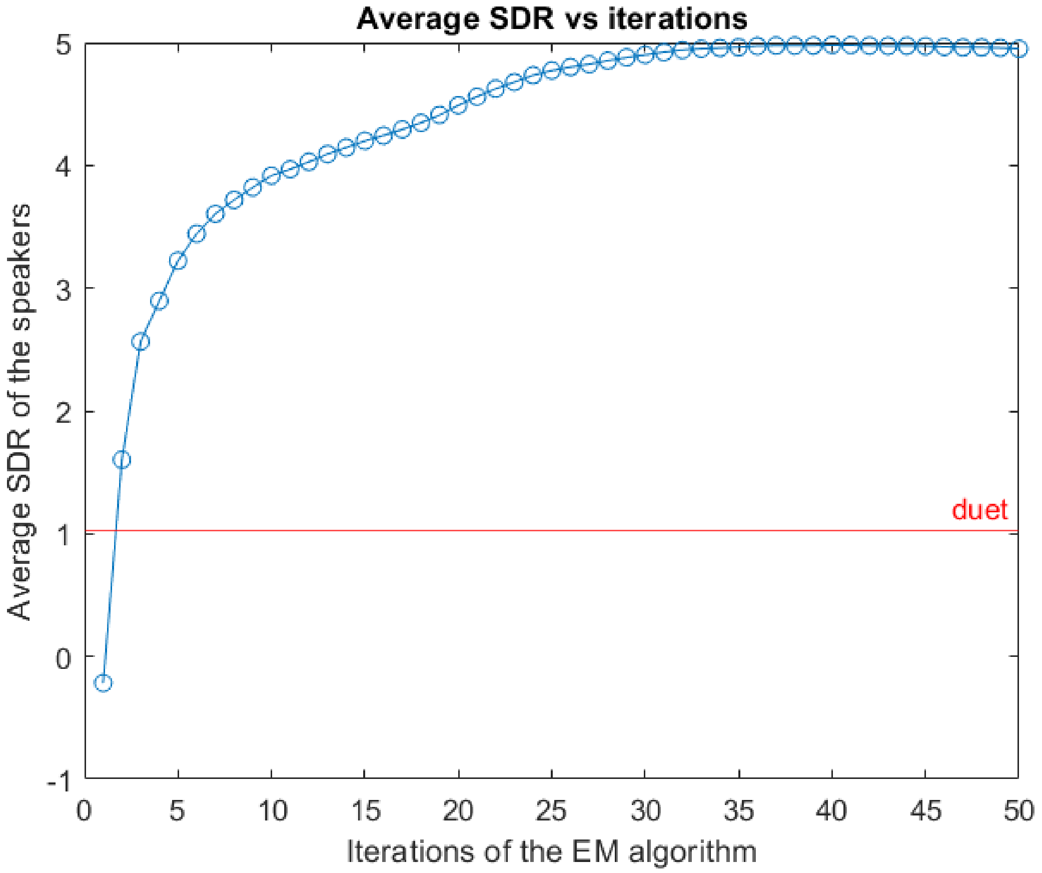


We can see that the starting point with the DUET algorithm was about 3.4 and average SIR increased rapidly with the number of iterations of the EM algorithm. The SIR stabilizes at 8.8 after ≅ 25 iterations of the EM algorithm, higher than the starting point in 5.4;

This is a major difference between the DUET's average SIR and the EM algorithm's average SIR.
This difference makes it so that you can clearly listen to the DUET's separation and distinctly hear the interferences over the estimation.
The DUET algorithm lacks the separation quality in real environments, this is because the DUET algorithm has many assumption and some of them do not always come to reality, the result is a weak separation which the EM maximizes.

This does not mean the DUET is a "bad" separation algorithm but in real environments it lacks the necessary qualities to estimate the sources without interferences, making the resulting estimates include a lot of musicals and termors.

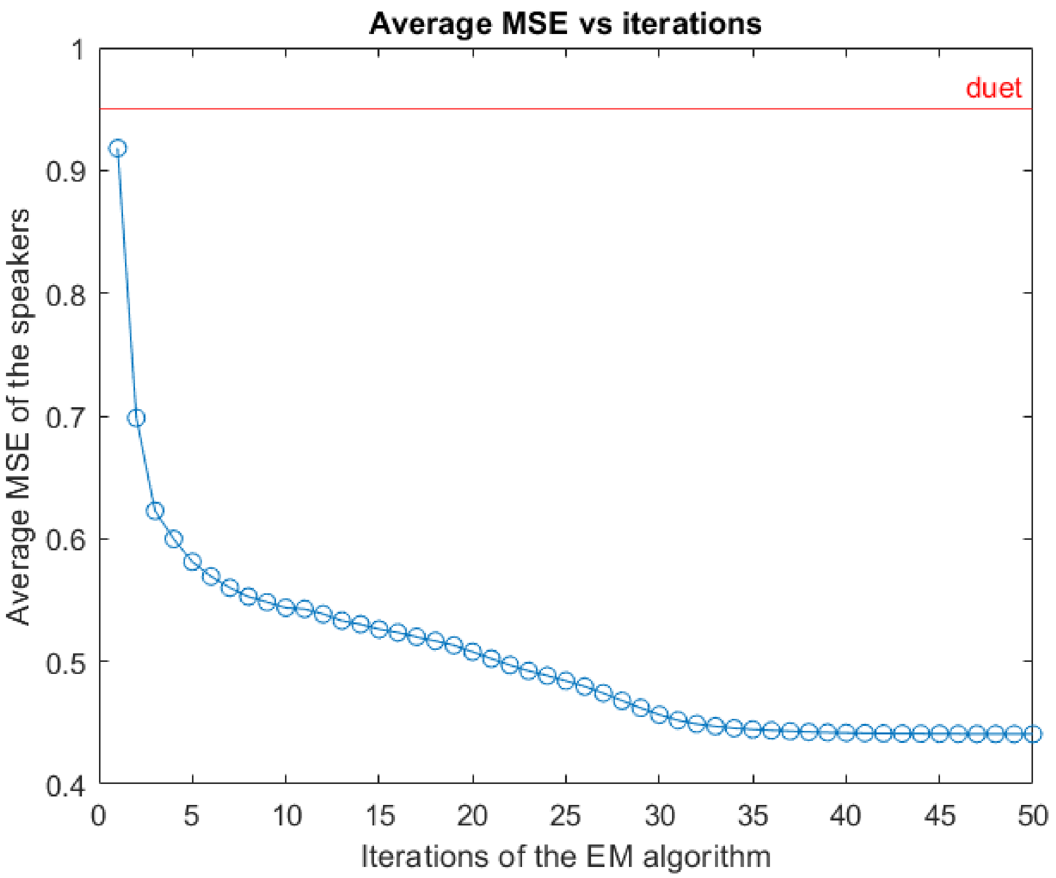
SDR evaluation:



We can see from this graph that the average SDR of the DUET's algorithm is almost 1, meaning that the DUET barely has any effect for this measurement, yet the EM algorithm improves this result by about 4, reaching to a point of SDR = 5 after about 28-30 iterations.

This can be heard in the DUET's separation as there is a lot of distortion and noise, this does not mean that the EM algorithm's separation is smooth, you can still hear noise in the separation, but it is not as noticeable as with the DUET's.

MSE evaluation:



The MSE measurement is quite different from the rest, it is not done by the BSS_eval toolkit as with the other measurements but with a simple script we designed.

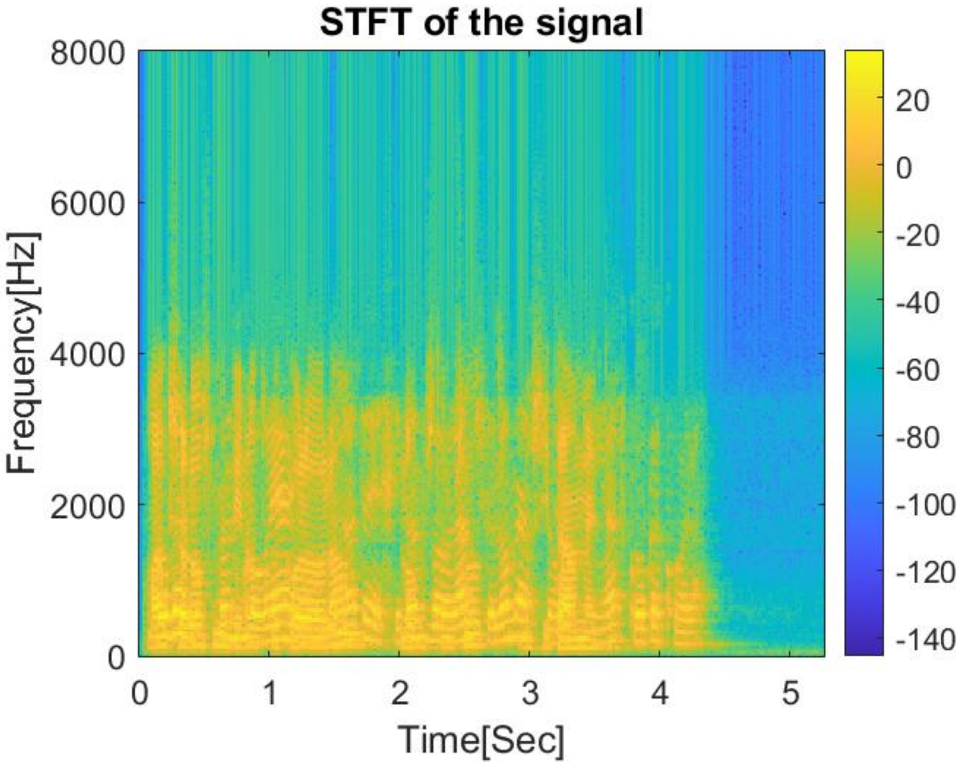
The other measurements are measured in a dB scale, while the average MSE is evaluated in a linear scale, calculated with the simple formula:

$$\text{MSE} \equiv \frac{1}{N}\sum_{i=1}^{N}\left(S_i - \hat{S}_i\right)^2$$

From the graph we can see that the average MSE of the estimated duet sources and actual sources is ≈ 0.95 while the average MSE of the estimated EM sources and actual source is ≈ 0.45 after roughly 35 iterations, though most of the MSE difference is made within the first 5 iterations of the algorithm.

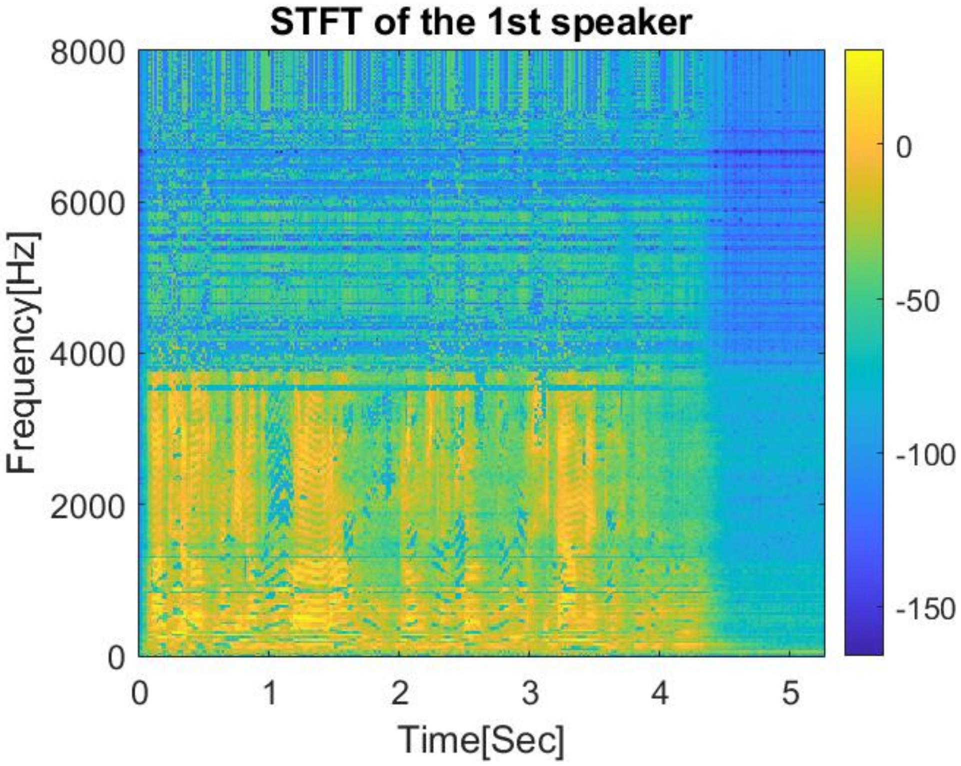
This means that the EM's estimated sources are closer to the original sources than the DUET's estimated sources and shows that the EM has successfully maximized the DUET's separation.

Let us examine the spectrogram of the mixture (STFT domain) so we can analyze it:

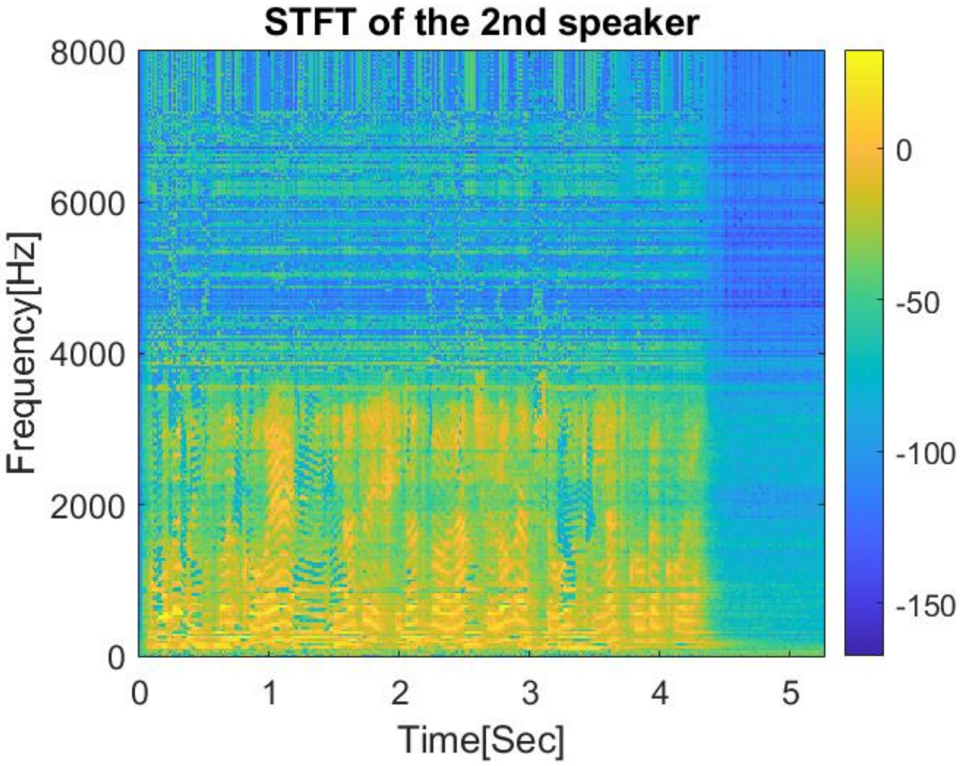


And the spectrogram of the estimated sources:

1st source:



2nd source:



We can see from the spectrograms that the algorithm masked each TF bin for its source and muted the TF bins that are uncorrelated to the current source.

Overall the results of the algorithm are quite satisfying, it was a good experience to see how mathematics and algorithmics can be used in digital signal processing, and it was refreshing to see the practical aspects of the methods we had learned in our signal processing courses.

Running time

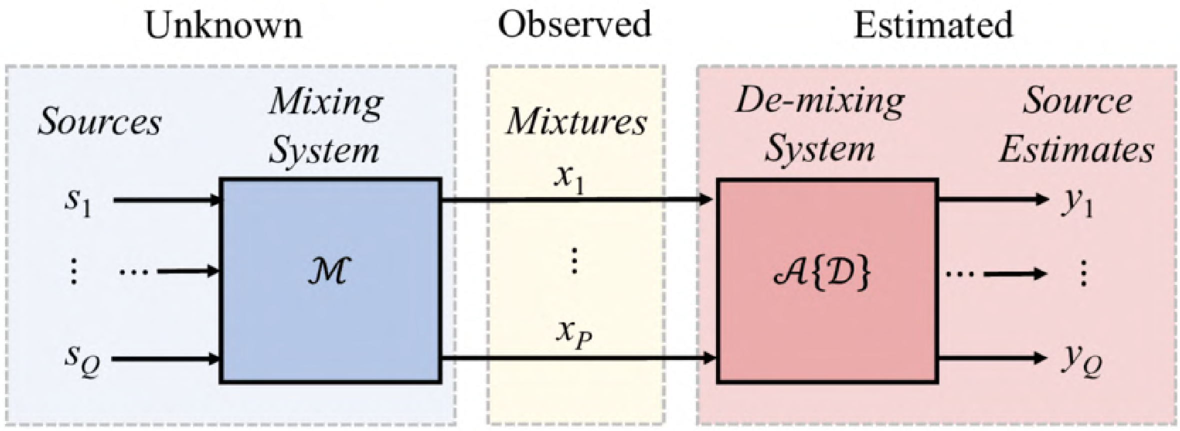
In order to improve the running time of the algorithm we have made a few assumption that are worth mentioning:

1. The STFT is symmetric for our real speech signal, therefore, there is no reason to run over all TF bins.
   We can run the algorithm on just half of the TF bins and this way we can cut the running time of the algorithm by half.
2. Iteration number of the algorithm – as we have seen from the results, there is no reason to run the algorithm for more than 20-30 iterations, and even after 15 iterations, the results are quite satisfying.
   Therefore, we usually run the algorithm for 15-20 iterations, this way the running time of the algorithm is not too long.
3. Another parameter which greatly affects the running time of the algorithm is the size of time frame and hop size (size of correlation) in the STFT transform.
   When the time frame is too big, we get less time blocks and the algorithm takes less time to run, but in this case, we lose resolution on the time signature of our signal and the resulting separation can drop in quality.
   On the other hand, if we pick the time frame too small, the time resolution can increase but the frequency resolution can drop, the algorithm may take more time to run, but the resulting separation may not increase in quality.
   What matters most is the hop in between time windows, if the correlation hop has too many samples within it, the number of time blocks may increase dramatically without reason.
   It is important to note the speech correlation does not last for very long so there is no reason to take too many samples in our correlation hop.
   Overall it is important to pick the right parameters in the STFT transform to get the best quality of separation while maintaining a satisfying separation.

It is worth mentioning that the best way to improve the running time of the algorithm is to efficiently code it.
We tried our best to consider each line of code in the process, the algorithm still takes a while to run all its processing, but the resulting running time is a lot shorter than what we had experienced at the beginning. 😊

Ideas for the future

In this project we have experienced with blind source separation.
BSS is a vast world by itself and we have only encountered a small portion of it.
The basic idea of BSS can be illustrated as:



Where some sources undergo a mixing process, both sources and the mixing process are unknown.
Then we observe the mixture and try to estimate the sources with a de-mixing process (BSS).

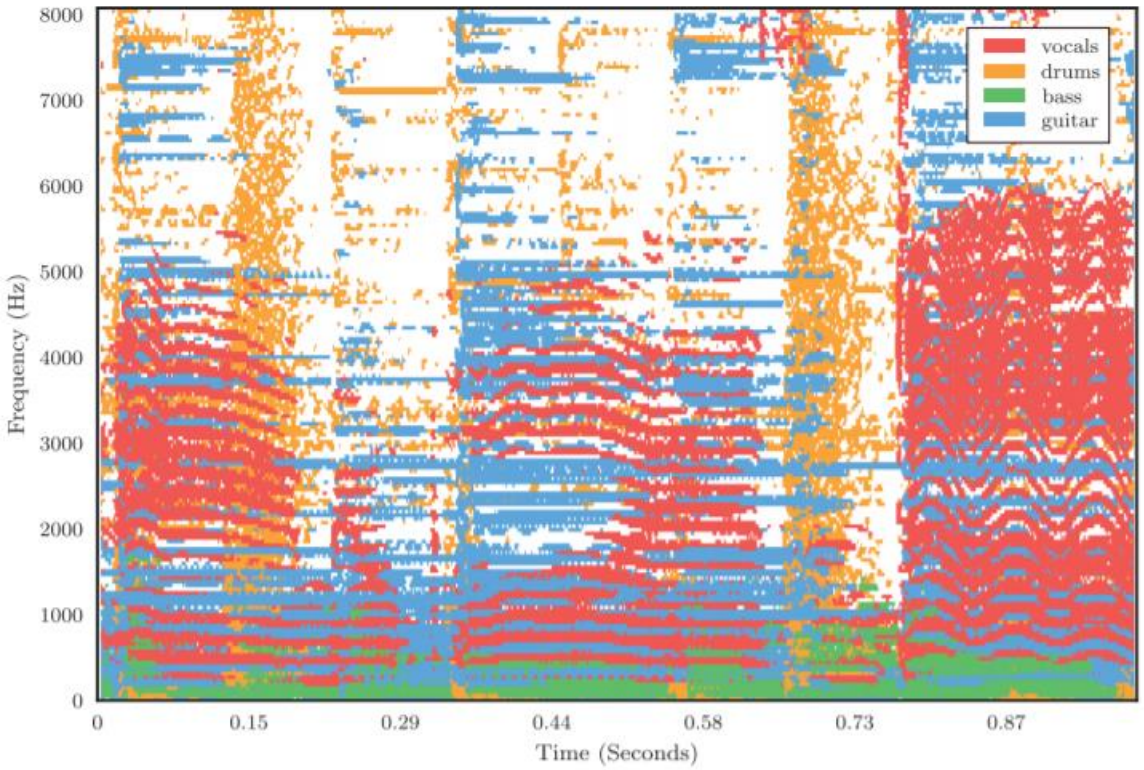
In our project, the sources were speakers (speech sources), but this does not have to always be the case.
On the following pages, let us explore some different ideas for the future:

1. Music separation – not that far off speech separation, we can think of a musical piece as a number of sources (musical tracks) mixed together and observed by the listener.

   For example, we can think of a "Beatles" song as a musical mixture with 4 different sources, with John Lennon playing the rhythm guitar and vocals, Paul McCartney playing the bass, George Harrison playing the lead guitar and Ringo Star playing the drums.
   In this case, we can think of estimating the sources as estimating each one of the tracks and extracting each one of the instruments played.

   This type of algorithm can be called "MSS" (Musical Source Separation).
   These types of algorithms usually exploit the unique properties of music signals in order to separate the target signal from the background.

An example of such separation can be illustrated as a spectrogram of the different sources:



Here, we can see each of the sources in a musical piece in the Time-Frequency domain.
The vocals plotted as red, the drums plotted as orange, the bass plotted as green and guitar plotted as blue.
We can clearly see the characteristics of the different instruments, for example, the bass guitar is obviously dominating the lower frequency while the lead guitar is more spread out since it has many different harmonics.

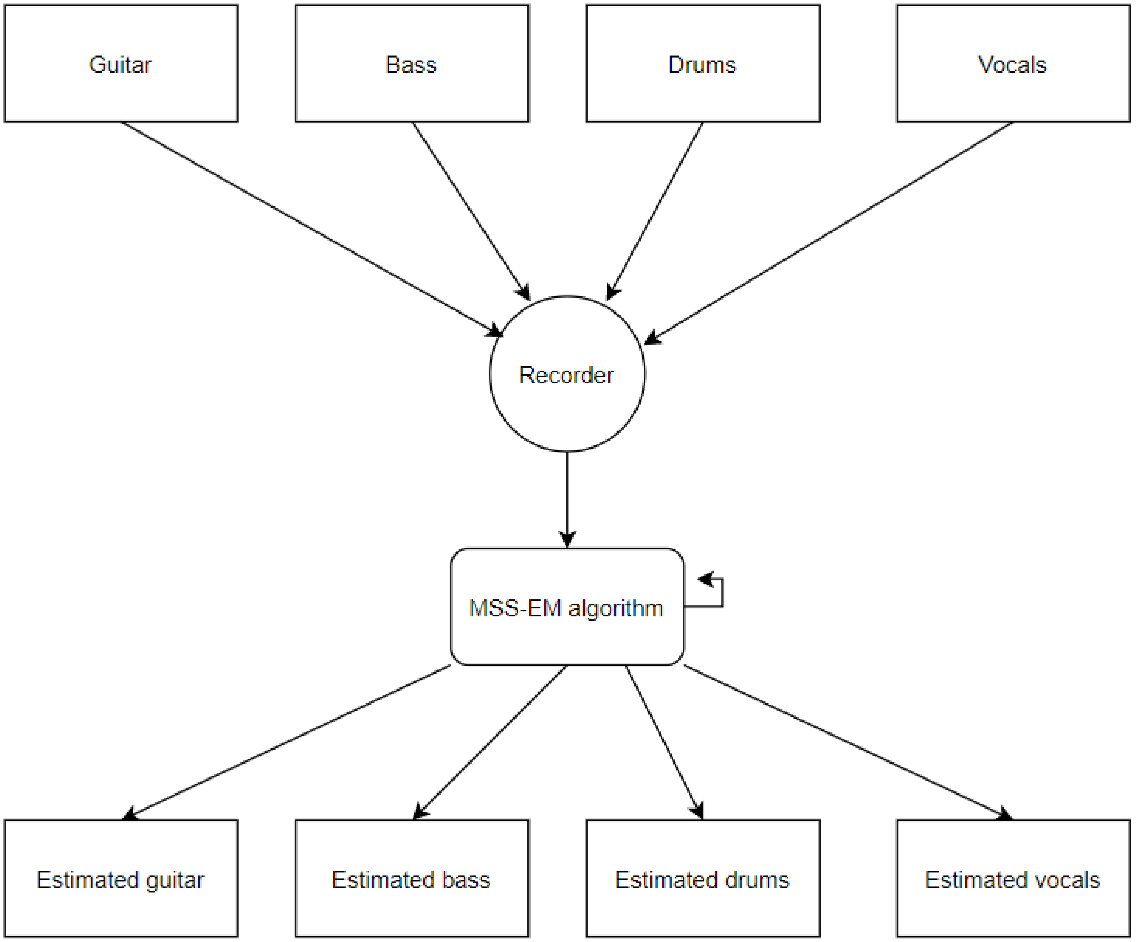
Our idea for future development is to redefine the speech sources as musical instruments and define the new EM parameters and data to maximize a new auxiliary function so that we can perform music separation.

We are well aware that this is not an easy thing to do, speech separation is vastly different from music separation and an entirely different field.
But it would be interesting to take the ideas of the EM algorithm and try to implement them for music separation.

In the paper:
Estefania Cano, Derry Fitzgerald, Antoine Liutkus, Mark Plumbley, Fabian Robert-Stöter. Musical Source Separation: An Introduction. IEEE Signal Processing Magazine, Institute of Electrical and Electronics Engineers, 2019, 36 (1), pp.31-40. ff10.1109/MSP.2018.2874719ff. ffhal-01945345f

The writer mentions the EM algorithm as a way to achieve source separation for musical instruments:
"Estimating source parameters from the mixture is not trivial, and it can be difficult to obtain good source parameters to enable successful filtering at the first try. For this reason, it is common to proceed iteratively: First, separation is achieved with the current estimated source models. These models are then updated from the separated signals, and the process repeated as necessary. This approach is rooted in the Expectation-Maximization algorithm."

A general schema for this idea can be illustrated as:

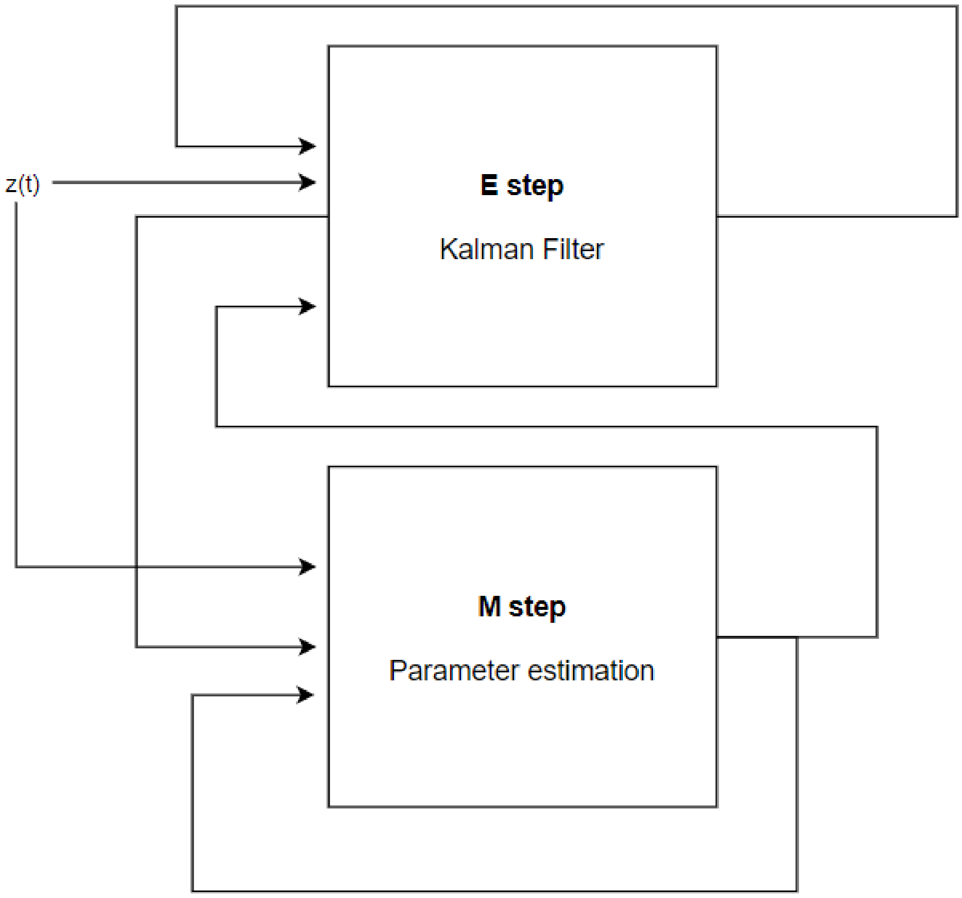


This may be an interesting idea to develop and research in the future.

2. Recursive Expectation Maximization (REM) algorithm – Another idea for future development is to develop a recursive EM algorithm for online use.
   In our project the algorithm requires the entire observation vector and only then we can start the processing, an idea for future development is online use of the EM algorithm, where in this case, the algorithm separates live recordings.

   The REM algorithm can be implemented with the Kalman Filter used in the E step of the algorithm and parameter estimation being implemented in the M step of the algorithm.

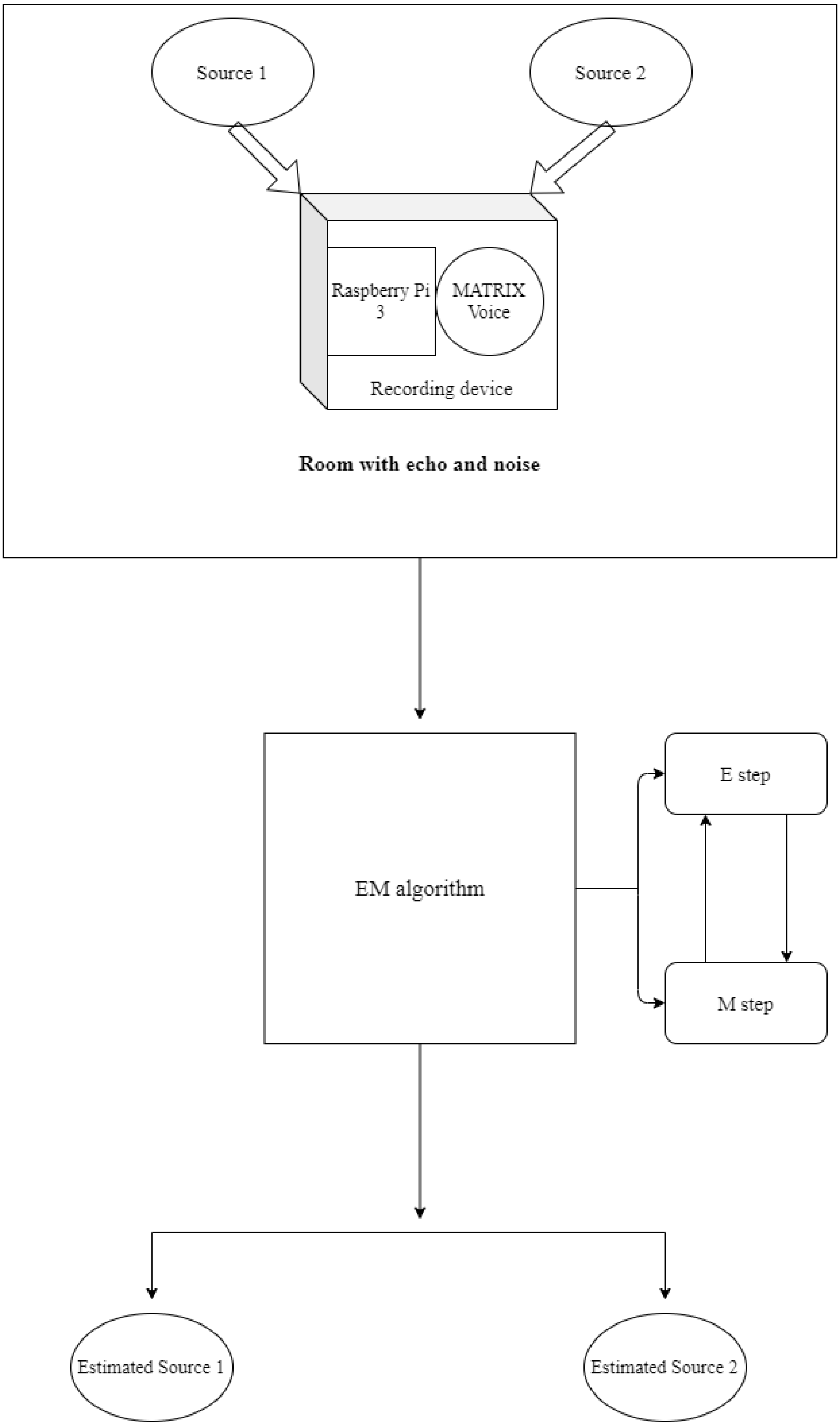
   A general schema for this algorithm can be views as:



   Where both E step and M step are recursive, while sending data between the as well.

Summary

We can summarize the algorithm presented in this project with the following illustration:



Where the two sources are recorded in a room with some background noise and echo (as with any room) using a Raspberry Pi 3 connected to a MATRIX Voice microphone array.

At early stages of development, we had the two sources recorded beforehand using a BLUE Yeti microphone and we used a RIR generator to create a simple environment for the recording (meaning that we used a room impulse response of 8 delta functions).
This way the separation is quite easy for the algorithm.

After we had successfully separated signals with the RIR generator function, we started working with actual RIRs recorded at the acoustic signal processing laboratory at the faculty of engineering, Bar Ilan University.

Finally, after managing the separate signals with actual RIRs, we could start experimenting with actual audio recording we ourselves recorded using the MATRIX Voice.

The development of the EM algorithm was especially difficult for us, we have encountered many bugs and issues with stabilizing the algorithm.
We spent a lot of time understanding the theory behind the algorithm, but when it came to practically code it, we had many difficulties.

We started out by trying to implement the initialization of parameters, but after coding the theory in the paper, this did not work.
Later on, we coded the Estimation and Maximization part of the algorithm which took us quite a while but after many tries (and searching for 1 tiny bug for a month!) we successfully done so.

The initialization was especially challenging since it is such an important part of the EM algorithm, and initializing the parameters un correctly can result in the auxiliary function convergence into a local maximum and a weak separation overall.

Our advisor, Mr. Aviad Eisenberg advised us to use the DUET method as the initialization stage of the algorithm and sent us a paper that explains the DUET algorithm.
After trying to initialize the algorithm with the DUET method we still had some difficulties with finding all the bugs in the code.

From the STFT and iSTFT functions, to the Estimation and Maximization functions, we had many unknown bugs as expected.
After the help of Aviad, we managed to find the bugs and the algorithm started working! 😊

Improvement Suggestions

One major improvement suggestion we have is to initialize the algorithm without the DUET method.

In the paper, $x_d^{(0)}(t)$ is initialized as:

$$x_d^{(0)}(t) = w_d^{(0)}(t) \cdot \left(u_d(t)^{(0)}\right)^H \cdot z(t)$$

Where $u_d(t)^{(0)}$ is calculated as $\frac{\boldsymbol{h}_d^H \boldsymbol{R}_v^{-1}}{\boldsymbol{h}_d^H \boldsymbol{R}_v^{-1} \boldsymbol{h}_d} |_{\overline{\theta}=\overline{\theta}^{(0)}}$.

The a posteriori coefficients are initialized by $w_d^{(0)}(t) = \frac{p_d^{(0)}(k) \cdot f(\boldsymbol{z}(t,k) \mid d; \overline{\boldsymbol{\theta}}^{(0)})}{\sum_{d' \in \mathbb{m}_D} p_{d'}^{(0)}(k) \cdot f(\boldsymbol{z}(t,k) \mid d\prime; \overline{\boldsymbol{\theta}}^{(0)})}$

Where $f(\boldsymbol{z}(t,k) \mid d; \overline{\boldsymbol{\theta}}^{(0)}) = \mathcal{N}_c \left(z(t); 0, h_d^{(0)} \left[h_d^{(0)}\right]^H + R_v^{(0)}\right)$.

The initial RTFs are calculated as:

$$h_1^{(0)} = \frac{\boldsymbol{q_1} + \boldsymbol{q_2}}{\sqrt{\lambda_1 + \lambda_2}}, h_2^{(0)} = \frac{\boldsymbol{q_1} - \boldsymbol{q_2}}{\sqrt{\lambda_1 + \lambda_2}}$$

Where q_1, q_2 are the eigen vectors of $\widehat{\boldsymbol{R}} = \frac{1}{T} \sum_{t \in \mathbb{m}_T} z(t) z(t)^H$ with respect to λ_1, λ_2 the largest eigen values of $\widehat{\boldsymbol{R}}$.

We tried initializing our algorithm this way many times, but this initialization was usually too weak, and the resulting separation was therefore nonexistent.

We ended up staying with the DUET algorithm as the initialization of the EM algorithm after consulting with our advisor Aviad Eisenberg about doing so, but this is still something that we would like to improve on one day.

Conclusions

Working on the project, we were exposed to the blind source separation world.
The BSS problem has many applications:

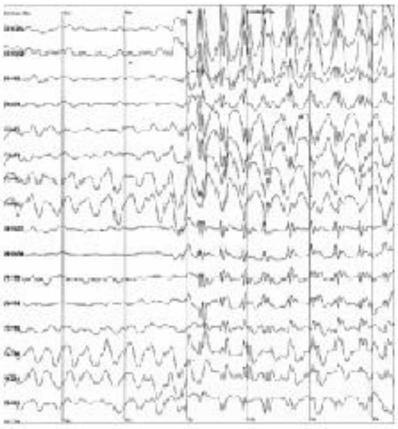
1. Cocktail party problem – a group of people talking at the same time, you have multiple microphones picking up mixed signals but the goal is to isolate the speech of a single person (this is the application of our project).
2. Medical imaging – in medical imaging such as MEG, the external sources of electromagnetic fields will significantly degrade the accuracy of the measurement.
   By applying BSS techniques, we can remove undesired artifacts from the signal.
3. Music – another application is the separation of musical signals using BSS techniques.
4. EEG – in EEG and MEG the interferences from muscle activities can mask the desired signal. However, with different BSS techniques we can separate the accurate representation of brain activity.

Cocktail Party Problem

Medical Imaging

Music

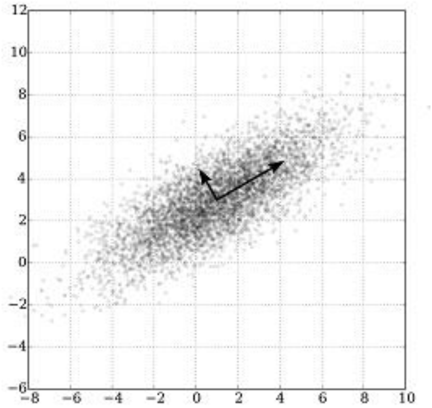
EEG



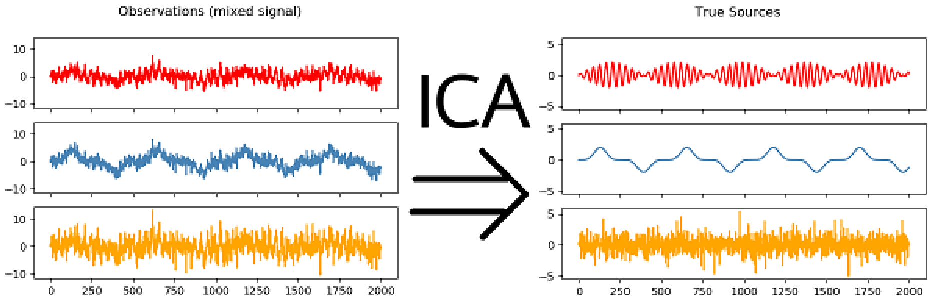
The BSS problem is yet to be solved and research in the field is still undergoing process.

In our project we focused on applying the EM algorithm to determine sources, but many other approaches are viable such as:

1. Principal Component Analysis – PCA tries to find the "best fitting" estimate to the data by minimizing the average squared distance from the data.



2. Independent Component Analysis – ICA is a computational method for separating a multivariate signal into additive subcomponents by assuming that the subcomponents are non-Gaussian signals that are statistically independent from each other.



3. Computational Auditory Scene Analysis – CASA is the method of separating audio signals in the same way that human listeners do. It is different from the other BSS methods in that it is based on the mechanisms of the human auditory system. Because of that, CASA uses no more than two microphones for the recording of an acoustic environment.

Implementing the theoretical aspects of what we had learned for 4 years was very satisfying to us.
At first we were worried that we don't know anything about BSS and practical signal processing, but the learning aspect of this project is what made it so special to us.

Signal processing in general is a very interesting topic and for this reason we wanted to study more of it and chose to do so at the end of the 2nd year of our degree.
In this project we implemented what we had learned in many courses such as:

A. Random signals and noise.
B. Statistical signal processing.
C. Digital signal processing.

We are very glad to have picked this project as our final BSc project and are very thankful for the people who have helped us achieve our goal.

Thank you for reading.
Ofek Ophir
Afek Steinberg

Bibliography

1. Schwartz, Boaz & Gannot, Sharon & Habets, Emanuël. (2017). Two Model-Based EM Algorithms for Blind Source Separation in Noisy Environment. IEEE/ACM Transactions on Audio, Speech, and Language Processing. PP. 1-1. 10.1109/TASLP.2017.2738438.

2. S. Gannot, E. Vincent, S. Markovich-Golan and A. Ozerov, "A Consolidated Perspective on Multimicrophone Speech Enhancement and Source Separation," in IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, no. 4, pp. 692-730, April 2017.

3. Rickard, Scott. (2007). The DUET blind source separation algorithm. 10.1007/978-1-4020-6479-1_8.

4. Estefania Cano, Derry Fitzgerald, Antoine Liutkus, Mark Plumbley, Fabian Robert-Stöter. Musical Source Separation: An Introduction. IEEE Signal Processing Magazine, Institute of Electrical and Electronics Engineers, 2019, 36 (1), pp.31-40. ff10.1109/MSP.2018.2874719ff. ffhal-01945345f

Appendices

We will be adding a ZIP folder with all the codes for this project.
The code is separated to:
<u>Mathematical functions</u>

- STFT – STFT transform of a signal
- iSTFT – Inverse STFT transform of a signal
- bss_eval_sources – Evaluate toolkit
- biorwin – support function (not ours)
- lnshift – support function (not ours)
- my_SNR – calculate new SNR
- NC – Gaussian Complex
- shiftcir – support function (not ours)
- tfanalysis – support function (of duet), (not ours).
- Tfsynthesis – support function (of duet), (not ours).
- twoDsmooth – support function (of duet), (not ours).

<u>Algorithm functions</u>

- Initialization – initialization of parameters
- Expectation – E step
- Maximization – M step
- duet – the DUET algorithm
- gain_ambguity – normalize gain
- Generate – generate function if RIR is used.
- Get_Input – get input in time-frequency
- <u>rir_gen – RIR generator function if needed</u>

<u>Output functions</u>

- EM – script that runs the algorithm for a generate function
- EM_live – script that runs the EM for a mixture recorded with our setup.
- Check – check script for each function
- plot_h_in_time – plot h in time (it was just there to check some stuff).
- Results – get all the results and plot with bss_eval.

To run our code, use the EM/EM live scripts, depending on your mixture.
If you have an 8-channel speech signal ready (recorded from the setup) use the EM_live script.
If you only have single channel speech signals ready (recorded from anywhere) use the EM script to generate the appropriate audio files (with RIR convolution) and run the algorithm.

All of the code is in MATLAB, all files end with .m
$F_s = 16kHz\,, J = 8\,, D = 2$

Thank you, Ofek and Afek.