Friedrich-Alexander-Universität Erlangen-Nürnberg

Lab Course

Speech Enhancement Using Microphone Arrays

International Audio Laboratories Erlangen

Prof. Dr. ir. Emanuël A. P. Habets
Friedrich-Alexander Universität Erlangen-Nürnberg
International Audio Laboratories Erlangen
Am Wolfsmantel 33, 91058 Erlangen
emanuel.habets@audiolabs-erlangen.de

AUDIO LABS

International Audio Laboratories Erlangen
A Joint Institution of the
Friedrich-Alexander Universität Erlangen-Nürnberg (FAU) and
the Fraunhofer-Institut für Integrierte Schaltungen IIS

Fraunhofer IIS

Authors:

Soumitro Chakrabarty,
Maja Taseska,

Tutors:

Soumitro Chakrabarty,
Maja Taseska,

Contact:

Soumitro Chakrabarty, Maja Taseska,
Friedrich-Alexander Universität Erlangen-Nürnberg
International Audio Laboratories Erlangen
Am Wolfsmantel 33, 91058 Erlangen
soumitro.chakrabarty@audiolabs-erlangen.de
maja.taseska@audiolabs-erlangen.de

Lab Course

Speech Enhancement Using Microphone Arrays

Abstract

This module is designed to give the students a practical understanding of performing speech enhancement using microphone arrays and demonstrate the difference between different techniques. This module is closely related to the lecture *Speech Enhancement* given by Prof. Dr. ir. Emanuël Habets.

In this exercise, the students will implement a commonly used spatial signal processing technique known as beamforming, and analyse the performance of two different beamformers, a fixed beamformer known as delay-and-sum beamformer and a signal-dependent beamformer known as minimum variance distortionless response (MVDR) beamformer, for a noise and interference reduction task. Their performances will be compared via objective measures to demonstrate the advantages of the signal-dependent beamformers.

1 Introduction

Microphone arrays play an important role in applications like hands-free communication in smartphones and teleconferencing systems. The spatial diversity offered by microphone arrays can be used to extract the speech signal of interest from the microphone signals corrupted by noise, reverberation and interfering sources. A typical method for this is to filter and sum the microphone signals such that the signal from the so-called *look direction* is reinforced while signals from all other directions are attenuated. Such a method is known as beamforming.

This module will give a practical understanding of such methods applied for the task of speech enhancement. The document provides the theoretical background necessary to understand the formulation of the beamforming methods and the practical aspects regarding their implementation.

For the experiments in this module, we consider a scenario where a microphone array on a hands-free device captures two sound sources in a room. As illustrated in Figure 1, one of the sources is the desired source and the other is an interferer. The task in this experiment is to process the microphone signals to enhance the speech of the desired source while suppressing the interfering speech and noise.

The general work flow for this module is illustrated in Figure 2. The microphone signals are first transformed into the time-frequency domain via short time Fourier transform (STFT) to obtain

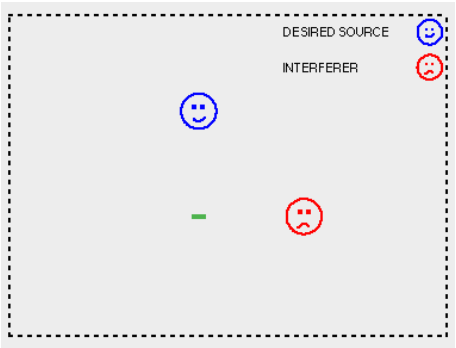

Figure 1: Illustration of the scenario considered in this experiment. The green line represents the microphone array.

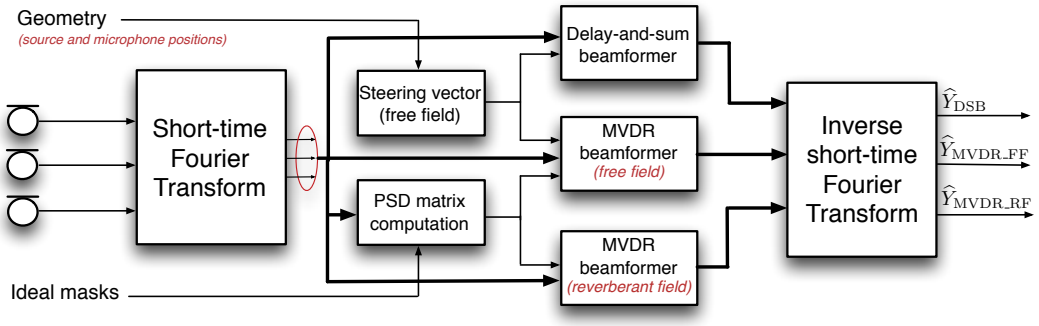

Figure 2: Block diagram of the general work flow for this module. The task for each block is described in the corresponding section.

the input signals for the beamforming algorithms. An overview of the tasks for implementing the different beamformers in this module is given as follows:

- The first task is to compute the steering vector using the geometric information, i.e., the source and microphone positions. The steering vector, along with the STFT domain microphone signals, is given as an input for the implementation of the signal-independent delay-and-sum beamformer (DSB). This is further explained in Section 4.
- The next task is to compute and apply the DSB to the microphone signals to obtain the filtered output (explained in Section 4.1).
- For the signal-dependent minimum variance distortionless response (MVDR) beamformer, we need to compute the power spectral density (PSD) matrices for the desired source, the interference and the noise signal, which is the next task. The PSD matrices are computed using the microphone signals and the ideal masks for each of the above mentioned signals. The ideal mask specifies at which time frequency points the PSD matrix needs to be updated. This is described in Section 5.1.
- The next task is to compute the MVDR filter for a free-field propagation model (described in Section 2.2.1) using the steering vector and the PSD matrix of the undesired (interfering speech + noise) signals, and apply the filter to the input microphone signals to obtain the filtered output. The task is further explained in Section 5.2.
- The final task for this module is to compute the MVDR filter for a reverberant propagation model (explained in Section 2.2.2) using the PSD matrix for the desired and the undesired signals, and apply the filter to the input signals to obtain the filtered output. This task is explained in Section 5.3.

The filtered outputs are finally transformed back to the time domain using an inverse STFT. The performance of the different beamforming algorithms is evaluated using objective measures. The objective measures used in this module are interference reduction (IR), noise reduction (NR) and signal distortion index (SDI). These measures are further explained in Section 6.

2 Signal and propagation model

2.1 Signal Model

To explain the formulation of the beamforming techniques, we first need to define a signal model. Consider a discrete time-domain signal model (shown in Figure 3), where N microphones capture

the desired source signal and the interference in presence of additive noise. Then the nth microphone signal for $n = 1, \ldots, N$ is given by:

$$\begin{aligned} y_n(t) &= g_{n,d}(t) * s_d(t) + g_{n,i}(t) * s_i(t) + v_n(t), \\ &= x_{n,d}(t) + x_{n,i}(t) + v_n(t), \end{aligned} \tag{1}$$

where $s_d(t)$ is the desired source signal and $s_i(t)$ is the interfering speech signal. The acoustic impulse responses between the nth microphone and the sources are denoted by $g_{n,d}(t)$ and $g_{n,i}(t)$ for the desired and the interfering source, respectively. The variable $v_n(t)$ denotes the additive noise. The desired source signal and the interfering speech signal received at the nth microphone are denoted by $x_{n,d}(t)$ and $x_{n,i}(t)$, respectively. The sample-time index is represented by t and $*$ denotes linear convolution. Since the frequency characteristics of a speech signal vary over time, the processing of the received signals is done in the time-frequency domain. The time domain signals are transformed to the time-frequency domain via short-time-Fourier-transform (STFT). The STFT representation of (1) is given by:

$$\begin{aligned} Y_n(m,k) &= G_{n,d}(m,k)S_d(m,k) + G_{n,i}(m,k)S_i(m,k) + V_n(m,k), \\ &= X_{n,d}(m,k) + \underbrace{X_{n,i}(m,k) + V_n(m,k)}_{U_n(m,k)}, \end{aligned} \tag{2}$$

where the uppercase letters denote the time-frequency domain counterparts of the terms in (1) for the kth frequency bin, and m denotes the time-frame index. $U_n(m,k)$ denotes the undesired signals, which is the sum of the interference and the noise signals. We can express the N STFT domain microphone signals in vector notation as:

$$\begin{aligned} \mathbf{y}(m,k) &= \mathbf{g}_d(m,k)S_d(m,k) + \mathbf{u}(m,k), \\ &= \mathbf{x}_d(m,k) + \mathbf{u}(m,k), \end{aligned} \tag{3}$$

where $\mathbf{y}(m,k) = [Y_1(m,k), Y_2(m,k), \ldots, Y_N(m,k)]^T$, and $\mathbf{x}_d(m,k)$, $\mathbf{g}_d(m,k)$ and $\mathbf{u}(m,k)$ are defined similarly.

We can write the desired source signal vector $\mathbf{x}_d(m,k)$ as a function of the received signal at the first microphone:

$$\mathbf{x}_d(m,k) = \mathbf{d}(m,k)X_{1,d}(m,k),$$

The microphone signals can also be written as:

$$\mathbf{y}(m,k) = \mathbf{d}(m,k)X_{1,d}(m,k) + \mathbf{u}(m,k).$$

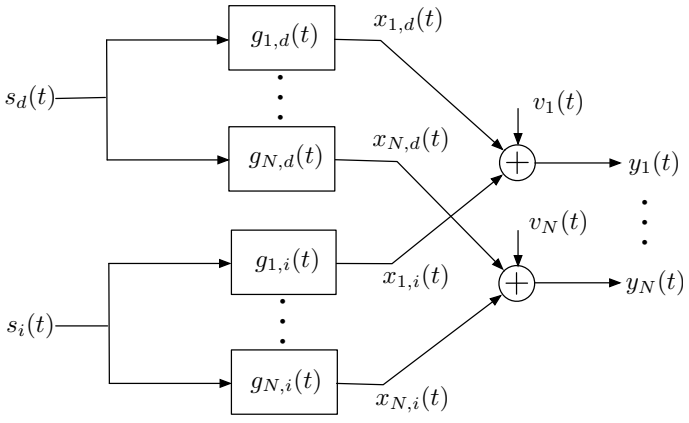

Figure 3: Time-domain signal model.

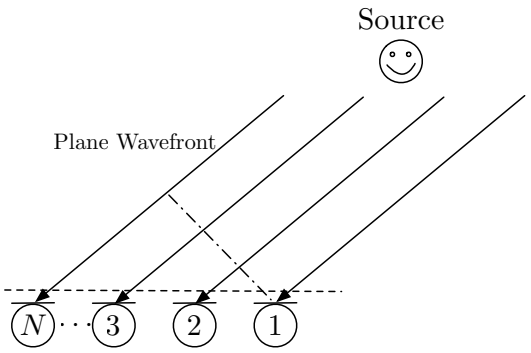

Figure 4: Illustration of the free-field model of signal propagation.

2.2 Propagation Model

For the given signal model, we consider 2 different signal propagation models: the *free-field model* where each microphone receives only the direct path signal, and the *reverberant model* where each microphone receives a large number of reflected signals in addition to the direct path signal. The propagation effects are modelled by the propagation vector $\mathbf{d}(m,k)$. For the two models, the formulation of the propagation vector is described as follows.

2.2.1 Free-field model

The free-field model is illustrated in Figure 4. The vector of acoustic transfer functions for the desired source is denoted by $\mathbf{g}_d(k) = [G_{1,d}(k), G_{2,d}(k), \ldots, G_{N,d}(k)]^T$. In the free-field model, the acoustic transfer functions are considered to be time-independent. The acoustic transfer function corresponding to the nth microphone is given by

$$G_{n,d}(k) = A_{n,d}(k)\exp\left(j\frac{2\pi k}{K}f_s\tau_{n,d}\right),$$

where $A_{n,d}(k)$ is the attenuation factor for the nth microphone due to propagation effects and $\tau_{n,d}$ is the absolute signal delay of the desired source signal at the nth microphone. The propagation vector for the free-field model is given by:

$$\begin{aligned}\mathbf{d}(k) &= \frac{\mathbf{g}_d(k)}{G_{1,d}(k)} \\ &= [1, D_2(k), \ldots, D_N(k)]^T,\end{aligned} \tag{4}$$

with

$$D_n(k) = \frac{G_{n,d}(k)}{G_{1,d}(k)} = \frac{A_{n,d}(k)}{A_{1,d}(k)}\exp\left(j\frac{2\pi k}{K}f_s\Delta_{\tau_{n,d}}\right).$$

where $\Delta_{\tau_{n,d}}$ is the time difference of arrival (TDOA) of the desired signal at the nth microphone with respect to the 1st microphone. f_s and K denote the sampling frequency and the total number of frequency bins, respectively. This is the formulation of the propagation vector considered for the delay and sum beamformer, and the MVDR beamformer (free field), explained later in Sections 4 and 5.2, respectively.

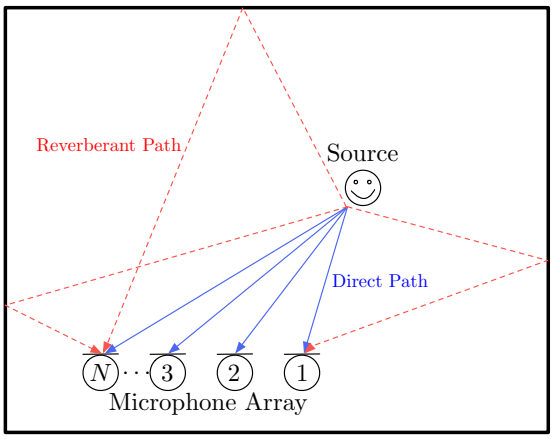

Figure 5: Illustration of the reverberant model of signal propagation.

2.2.2 Reverberant model

The reverberant model is illustrated in Figure 5. In the reverberant model, the propagation vector is time-frequency dependent and given by

$$\begin{aligned}\mathbf{d}(m,k) &= \frac{\mathbf{g}_d(m,k)}{G_{1,d}(m,k)} \\ &= [1, D_2(m,k), \ldots, D_N(m,k)]^T,\end{aligned} \tag{5}$$

with

$$D_n(m,k) = \frac{G_{n,d}(m,k)}{G_{1,d}(m,k)}.$$

It is generally difficult to further simplify this expression for the reveberant model. A way to estimate the propagation vector for the MVDR beamformer (reverberant field) will be presented in Section 5.3.

2.3 Beamforming

Our aim is to obtain an estimate of the desired source signal at the first microphone, i.e., $X_{1,d}(m,k)$. This is done by applying a filter to the observed microphone signals and summing across the array

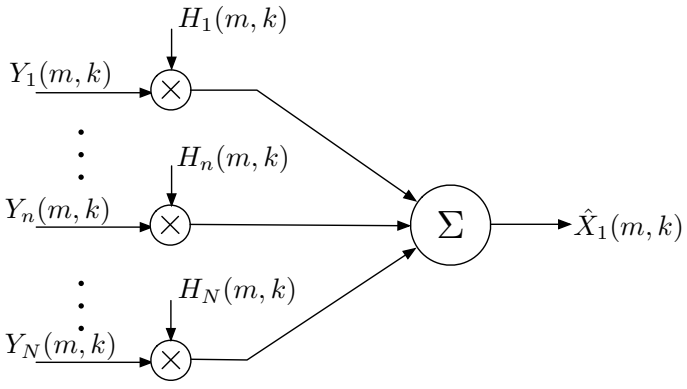

Figure 6: General block diagram for a beamformer.

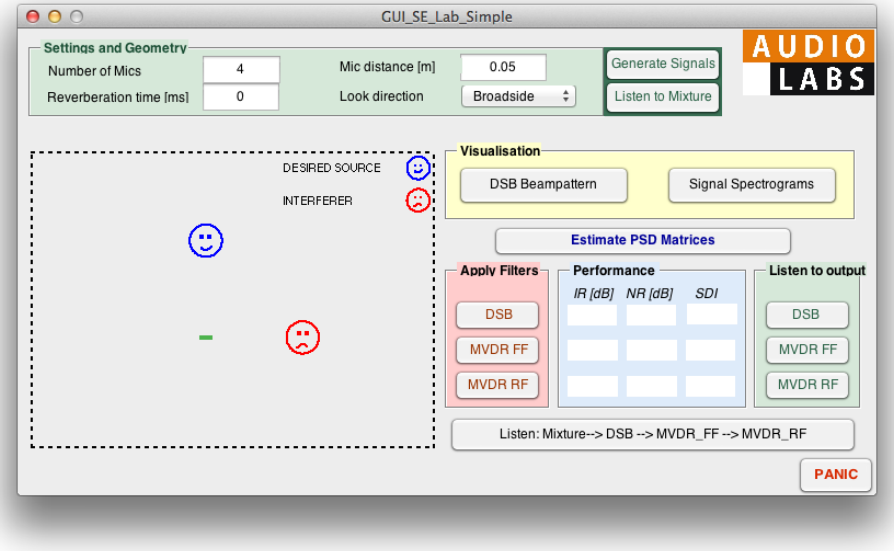

Figure 7: GUI.

elements (shown in Figure 6) and is given by:

$$\hat{X}_{1,d}(m,k) = \mathbf{h}^H(m,k)\mathbf{y}(m,k),$$

where $\mathbf{h}(m,k) = [H_1^*(m,k), H_2^*(m,k), \ldots, H_N^*(m,k)]^T$ is a filter of length N. Now, with the given framework the aim is to develop an analytical expression for the filter $\mathbf{h}(m,k)$, which is given by the different beamforming techniques explained in the Sections 4 and 5.

3 Experimental setup

A graphical user interface (GUI) is provided for the experiments in this module. A snapshot of the GUI is shown in Figure 7.

NOTE - In the following sections, parameters in the GUI are denoted in Green, variables that are part of the provided MATLAB codes are denoted in Red, and MATLAB function names are denoted in Blue.

For this module, the experimental setup consists of a room with 2 speakers and a uniform linear array (ULA) placed inside the room. The GUI is launched by running the MATLAB script GUI_SE_Lab_Simple. In the Settings and Geometry panel of the GUI, the following settings can be varied:

- Number of Mics - number of microphones in the array.
- Mic distance - inter-microphone distance, in meters.
- Reverberation time - the time for the sound to decay to a level 60 dB below its original level. This is given in milliseconds.
- Look Direction - this is the direction in which the desired source is located. Broadside corresponds to the desired source being located in front of the array whereas Endfire corresponds to the desired source being placed at the side of the array, i.e., the place where the interferer is positioned in Figure 7.

Analysis 1

- For all the experiments, the parameters in the Settings and Geometry panel of the GUI should be set as:
  - Number of Mics = 4.
  - Mic distance = 0.05
  - Reverberation time - for every experiment this parameter is varied in the range of 0-600 ms in steps of 200 ms.
  - Look Direction - the performance of each beamformer is analysed for both the specified look directions.
- Once the parameters for the Settings and Geometry are set, use the Generate Signals button to generate the microphone signals. Use the Listen to Mixture button to play back the mixture signal.

In the following sections, we describe the tasks for implementing the different beamformers and analysis of their performance.

4 Delay and sum beamformer

The delay and sum beamformer (DSB) is a fixed beamformer, i.e., the parameters of the beamformer are fixed and are not signal dependent. As the name suggests, this beamformer works by delaying the signals from certain microphones and then summing them afterwards. To explain this further, consider an array of N microphones. When the microphone array picks up a signal coming from an angle other than 90 degrees, every consecutive microphone receives the signal with an increased delay. This is because the signal entering from an angle needs to travel an additional distance to the next microphone in the array. This fact is exploited here in order to obtain a constructive interference in terms of the desired signal and a destructive interference in terms of the signals from noise or interfering sources.

To obtain a constructive interference in terms of the desired source, the filter needs to satisfy

$$\mathbf{h}^H(k)\mathbf{d}(k) = 1, \tag{6}$$

where $\mathbf{d}(k)$ is the propagation vector for the free-field model, given in Section 2.2.1. This propagation vector for the free-field model is also known as *steering vector*.

4.1 Steering vector (free field)

The first task in this module is to implement a DSB. For the DSB, first the steering vector needs to be computed. This task is performed by implementing parts of the fcn_compute_steerVec_linArray function. The input parameters of the function are:

- arrayCenter - coordinates of the center of the microphone array. Only X and Y dimensions are required for implementation.
- sourcePos - coordinates of the desired source. Only X and Y dimensions are required for implementation.
- numMics - number of microphones in the array.
- micDist - inter-microphone distance.

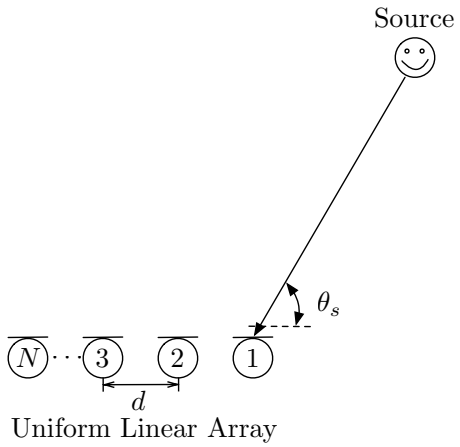

Figure 8: Reference diagram for Lab Experiment 1. θ_s is the angle of incidence of the desired source, d is the inter-microphone distance and N is the total number of microphones of the ULA. In practice, θ_s is computed with respect to the center of the array.

- freqvec - vector containing the frequencies for which the steering vector should be computed.

The output parameter of the function is the steering vector steerVec.

Lab Experiment 1

- As a first part of computing the steering vector, given the desired source position (sourcePos) and the position of the center of the array (arrayCenter), compute the direction-of-arrival θ_s of the desired source with respect to the array center. Implement the computation in the fcn_compute_steerVec_linArray function.

Considering the free-field model described in Section 2.2.1, the steering vector $\mathbf{d}(k)$ given in Equation **??** and 5 can be simplified as:

$$\mathbf{d}(k) = [1, \exp(j2\pi f_k \Delta_{\tau_2}), \ldots, \exp(j2\pi f_k \Delta_{\tau_N})], \tag{7}$$

where $f_k = \frac{2\pi k}{K} f_s$ denotes the frequency value corresponding to the kth frequency bin.

Lab Experiment 2

- Given the direction-of-arrival of the source signal θ_s (computed in Lab Experiment 1), (shown in Figure 8), the inter-microphone distance d and the speed of sound c, derive the expression for the time difference of arrival Δ_{τ_n} at the nth microphone.
- With the expression of the TDOA and given the vector of length K with frequencies f_k for which the steering vector needs to be computed, freqvec, implement Equation 7 in the fcn_compute_steerVec_linArray function.

Analysis 2

- Once the steering vector is computed, the beampattern of the DSB can be visualized using the DSB Beampattern button on the GUI.

  ∗ **Please contact one of the tutors when you reach this point.**

4.2 Delay-and-sum beamformer (Implementation)

Given the steering vector, the next task is to apply the DSB filter. Using the condition given in Equation 6, the DSB filter is given by:

$$\mathbf{h}(k) = \frac{1}{N}\mathbf{d}(k). \tag{8}$$

Here, the time index has been omitted since the delay and sum beamformer is a fixed beamformer. The DSB filter is applied to the microphone signals in the fcn_applyDSB function where the input parameters are:

- Y - spectrum of the microphone signals.
- X - clean speech signal spectra of the desired source at the microphones.
- V - noise spectrum at the microphones.
- arrCenter - coordinates of the center of the microphone array.
- sourcePos - coordinates of the desired source.
- numMics - number of microphones in the array.
- micDist - inter-microphone distance.
- freqvec - vector containing the frequencies at which the steering vector should be computed

The output parameters of this function are:

- Ydsb - spectrum of the signal obtained after applying the DSB filter to the microphone signals Y.
- Xdsb - spectrum of the signal obtained after applying the DSB filter to the clean speech signal of the desired source X. This output is only required for the performance evaluation.
- Vdsb - spectrum of the noise signal obtained after applying the DSB filter to the noise signals at the microphones V. This output is only required for the performance evaluation.

As a part of the implementation, the function fcn_compute_steerVec_linArray needs to be called within the fcn_applyDSB function.

Lab Experiment 3

- Given the steering vector (computed in Lab Experiment 2), implement Equation 8 in the fcn_applyDSB function to obtain the DSB filter.
- Apply this filter to the microphone signals Y to obtain the filtered output Ydsb (refer to Equation 6).
- Also apply the computed filter in a similar manner to X and V to obtain Xdsb and Vdsb, which are required for performance evaluation.

  NOTE: The MATLAB command `C = (B)'` gives $\mathbf{C} = \mathbf{B}^H$, i.e., the Hermitian of the matrix $\mathbf{B}$. The simple transpose of the matrix, $\mathbf{C} = \mathbf{B}^T$, is given by `C = (B).'`.

Analysis 3

- Once the implementation of applying the DSB is done, the function can be run from the GUI with the DSB button in the Apply filters panel. When the filter is applied, the performance evaluation measures are automatically computed and are displayed in the adjoining Performance panel of the GUI.
- Set the Look Direction to Broadside and mark the interference reduction (IR), noise reduction (NR) and the signal distortion index (SDI) values in the corresponding figures provided at the end of this document. Perform this task for different reverberation times, ranging from 0 to 600 ms in steps of 200 ms.
- Now, change the Look Direction to Endfire in the GUI and apply the DSB again. Repeat the above mentioned tasks for this modified configuration.
- Set the number of microphones (Number of Mics) to 10. Listen to the input and output for the reverberation times of 0 and 600 ms.
- Change the geometric parameters in the Settings and Geometry panel of the GUI according to Table 1 (given at the end of this document) and repeat the above mentioned steps to complete Table 1. Perform this task only for Broadside.

  ∗ **Please contact one of the tutors once you finish the tasks in this section.**

5 Minimum variance distortionless response (MVDR) beamformer

The next task is the implementation of the MVDR beamformer. The MVDR beamformer is a signal dependent beamformer i.e. the filter coefficients for the MVDR beamformer depend on the statistical properties of the received signals. The aim of the MVDR beamformer is to minimize the power of the undesired signal components at the output while ensuring that the desired signal is not distorted. Mathematically, this can be formulated as

$$\mathbf{h}_{\mathrm{MVDR}}(m,k) = \arg\min_{\mathbf{h}} \mathbf{h}^H(m,k)\mathbf{\Phi}_{\mathbf{u}}(m,k)\mathbf{h}(m,k) \;\text{ subject to }\; \mathbf{h}^H(m,k)\mathbf{d}(m,k) = 1, \tag{9}$$

where the constraint in the second part ensures a distortionless response. The power spectral density (PSD) of the undesired (interfering speech + noise) signals is denoted by $\mathbf{\Phi}_{\mathbf{u}}(m,k)$ and given by:

$$\mathbf{\Phi}_{\mathbf{u}}(m,k) = \mathcal{E}\{\mathbf{u}(m,k)\mathbf{u}^H(m,k)\}. \tag{10}$$

As can be seen Equation 9 is a constrained optimization problem, which can be solved using Lagrange multipliers. The obtained solution is given by:

$$\mathbf{h}_{\text{MVDR}}(m,k) = \frac{\mathbf{\Phi}_{\mathbf{u}}^{-1}(m,k)\mathbf{d}(m,k)}{\mathbf{d}^{H}(m,k)\mathbf{\Phi}_{\mathbf{u}}^{-1}(m,k)\mathbf{d}(m,k)}. \tag{11}$$

Given this formulation, it can be seen that we need to estimate the PSD matrix for the undesired (interfering speech + noise) signals. In the following section, we present a recursive algorithm for the estimation of the PSD matrices of the undesired signals $\mathbf{u}(m,k)$ and the desired signal $\mathbf{x}_d(m,k)$ that is later used to obtain an estimate of the propagation vector $\mathbf{d}(m,k)$ for the reverberant model.

5.1 Power spectral density (PSD) matrix computation

The first task for the implementation of MVDR beamformer is the recursive estimation of the PSDs. In this module, the task is to implement a general recursive PSD estimation method by completing sections of the fcn_recursive_PSD_estimation function. This function is later called within the GUI framework to estimate the PSD matrices of the desired speech (required later) and undesired (interfering speech + noise) signals.

NOTE: The task is to only implement the general recursive PSD estimation algorithm. The desired speech PSD Phi_d and the undesired (interfering speech + noise) signal PSD Phi_u are computed by calling this function separately for the desired and the undesired signals. The input parameters of the fcn_recursive_PSD_estimation function are:

- spectrum - STFT coefficients of the microphone signals.
- mask - indicates at which time-frequency points the desired or the undesired signal is dominant. This is used to determine at which time-frequency points an update needs to be made.
- alpha - constant averaging factor.

The output parameter of the function is the estimated PSD allPSDs.
The theoretical formulation of the recursive estimation method can be given by:

$$\hat{\mathbf{\Phi}}(m,k) = \mathcal{I}(m,k)[\alpha\hat{\mathbf{\Phi}}(m-1,k) + (1-\alpha)\mathbf{y}(m,k)\mathbf{y}^{H}(m,k)] + (1-\mathcal{I}(m,k))\hat{\mathbf{\Phi}}(m-1,k), \tag{12}$$

where $\mathcal{I}(m,k)$ is the indicator parameter (mask) that is used to determine at which time frequency points the relevant PSD matrix needs to be updated. The mask is different for the desired and undesired signals. If at a certain time-frequency point, the indicator parameter for the desired speech signal is 1, then the desired speech signal PSD is updated, otherwise if the indicator parameter for the undesired signals is 1 then the undesired signal PSD is updated. This indicator parameter (mask) is computed using an oracle mechanism and is not a part of the exercise. For practical implementation, Equation 12 can be simplified as:

$$\hat{\mathbf{\Phi}}(m,k) = \alpha'(m,k)\hat{\mathbf{\Phi}}(m-1,k) + (1-\alpha'(m,k))\mathbf{y}(m,k)\mathbf{y}^{H}(m,k), \tag{13}$$

where the modified update factor $\alpha'(m,k)$ (currentAlpha) is given by:

$$\alpha'(m,k) = \alpha + (1-\mathcal{I}(m,k))(1-\alpha). \tag{14}$$

For this exercise, it is only required to implement the general PSD estimation method given by Equation 13. The computation of the modified update factor (currentAlpha) is implemented using Equation 14.

Lab Experiment 4

- Given the spectrum of the microphone signals (spectrum), implement Equation 13 and 14 in the fcn_recursive_PSD_estimation function.
- Once the implementation is done, use the Estimate PSD Matrices button to estimate the desired signal PSD Phi_d and the undesired signal PSD Phi_u.

5.2 MVDR beamformer (free field)

Once the PSD matrix of the undesired signals is computed, the next task is to implement the MVDR beamformer filter and apply it to the microphone signals. Here, the task is to implement the MVDR beamformer with the steering vector computed in Lab Experiment 2. This task is to be done by completing the fcn_applyMVDR_FF function. The input parameters of the function are:

- Y - spectrum of the microphone signals.
- X - clean speech signal spectra of the desired source at the microphones.
- V - noise spectrum at the microphones.
- steerVec - steering vector computed in Lab Experiment 2.
- Phi_u - PSD matrix of the undesired (interfering speech + noise) signal.

The output parameters of the function are:

- Ymvdr - filtered microphone signals after the MVDR beamformer filter has been applied to Y.
- Xmvdr - filtered clean speech signal after the MVDR beamformer filter has been applied to X. This is only required for performance evaluation.
- Vmvdr - filtered clean speech signal after the MVDR beamformer filter has been applied to V. This is only required for performance evaluation.

Lab Experiment 5

- Considering $\mathbf{d}(m,k)$ as the computed steering vector (steerVec) and given the PSD matrix of the undesired signal (Phi_u), implement Equation 11 to obtain the MVDR filter.
- Apply this filter to Y, X and V to obtain the filtered outputs Ymvdr, Xmvdr and Vmvdr, respectively.

Note: For computing the inverse of the undesired signals' PSD matrix Phi_u, a function called my_inv is provided in the code. Please do not use any other function for this purpose. Also, while implementing Equation 11, use the min_val variable to avoid the "division by zero" problem.

Analysis 4

- Once the implementation of applying the MVDR filter with steering vector is done, the function can be run from the GUI with the MVDR FF button in the Apply filters panel. When the filter is applied, the performance evaluation measures are automatically computed and are displayed in the adjoining Performance panel of the GUI.
- Set the Look Direction to Broadside and mark the interference reduction (IR), noise reduction (NR) and the signal distortion index (SDI) values in the corresponding figures provided at the end of this document. Perform this task for different reverberation times, ranging from 0 to 600 ms in steps of 200 ms.
- Now, change the Look Direction to Endfire in the GUI and apply the MVDR filter with steering vector again. Repeat the above mentioned tasks for this modified configuration.

  ∗ **Please contact one of the tutors once you finish the tasks in this section.**

5.3 MVDR beamformer (reverberant field)

The final task in this module is the implementation of the MVDR beamformer by considering the propagation vector $\mathbf{d}(m,k)$ as a relative transfer function rather than the fixed propagation vector considered in the previous section. This task is to be done by completing the fcn_applyMVDR_RF function. The input parameters of this function are:

- Y - spectrum of the microphone signals.
- X - clean speech signal spectra of the desired source at the microphones.
- V - noise spectrum at the microphones.
- Phi_d - PSD matrix of the desired source signal.
- Phi_u - PSD matrix of the undesired (interfering speech + noise) signal.

The output parameters of the function are:

- Ymvdr - filtered microphone signals after the MVDR beamformer filter has been applied to Y.
- Xmvdr - filtered clean speech signal after the MVDR beamformer filter has been applied to X. This is only required for performance evaluation.
- Vmvdr - filtered clean speech signal after the MVDR beamformer filter has been applied to V. This is only required for performance evaluation.

The propagation vector considered here is formulated in Section 2.2.2. In practice, an estimate of $D_n(m,k)$ can be obtained using

$$D_n(m,k) = \frac{G_{n,d}(m,k)}{G_{1,d}(m,k)} = \frac{\mathcal{E}\left\{X_{n,d}(m,k)X^*_{1,d}(m,k)\right\}}{\mathcal{E}\left\{|X_{1,d}(m,k)|^2\right\}}. \tag{15}$$

where $\mathcal{E}$ is the expectation operator. The numerator denotes the cross-correlation between the STFT coefficients of the desired speech at the nth and the 1st microphone, and the denominator denotes the auto-correlation for the desired speech signal at the 1st microphone. The required correlations are part of the PSD matrix of the desired source signal (Phi_d).

Lab Experiment 6

- Given the PSD matrix of the desired source signal (Phi_d), compute the propagation vector as formulated in Equation 6.
- With the computed propagation vector, implement Equation 11 to obtain the MVDR filter.
- Apply this filter to Y, X and V to obtain the filtered outputs Ymvdr, Xmvdr and Vmvdr, respectively.

Analysis 5

- Once the implementation of applying the MVDR filter with generalised transfer function is done, the function can be run from the GUI with the MVDR RF button in the Apply filters panel. When the filter is applied, the performance evaluation measures are automatically computed and are displayed in the adjoining Performance panel of the GUI.
- Set the Look Direction to Broadside and mark the interference reduction (IR), noise reduction (NR) and the signal distortion index (SDI) values in the corresponding figures provided at the end of this document. Perform this task for different reverberation times, ranging from 0 to 600 ms in steps of 200 ms.
- Now, change the Look Direction to Endfire in the GUI and apply the MVDR filter with generalised transfer function again. Repeat the above mentioned tasks for this modified configuration.
- Listen to the inputs and outputs of the MVDR FF and MVDR RF for reverberation time of 600 ms.

  ∗ **Please contact one of the tutors once you finish the tasks in this section.**

6 Performance evaluation measures (Additional Reading)

The performance of the beamformers is measured in terms of objective measures. In this module, we use three different objective measures for the evaluation of the performance of the implemented beamformers. These measures are explained as follows.

6.1 Interference reduction (IR)

The first objective measure for evaluation is the interference reduction. It evaluates the suppression of the interfering speech signal achieved by the filter. Here, we compute the average difference between the segmental power of the interfering clean speech signal and the segmental power of the filtered version of it. It is formulated as:

$$\mathrm{IR[dB]} = \frac{1}{Q}\sum_{q=0}^{Q-1} 10\log\left(\frac{\sum_{t=qL}^{(q+1)L-1}|\hat{x}_i(t)|^2}{\sum_{t=qL}^{(q+1)L-1}|x_{1,i}(t)|^2}\right), \tag{16}$$

where $\hat{x}_i(t)$ is the filtered version of the interfering speech signal and $x_{1,i}(t)$ denotes the interfering signal at the reference microphone. Here, q is the segment index and L denotes the length of each segment.

6.2 Noise reduction (NR)

Noise reduction evaluates the suppression of additive noise achieved at the output. It is computed similarly to the IR. Instead of the interfering speech signal, we use the noise signal and its filtered version to evaluate the NR. It is formulated as:

$$\text{NR[dB]} = \frac{1}{Q}\sum_{q=0}^{Q-1} 10\log\left(\frac{\sum_{t=qL}^{(q+1)L-1}|\hat{v}(t)|^2}{\sum_{t=qL}^{(q+1)L-1}|v_1(t)|^2}\right), \tag{17}$$

where $\hat{v}(t)$ is the filtered version of the noise signal and $v_1(t)$ denotes the noise signal at the reference microphone. The variables q and L are defined similarly as before.

6.3 Signal distortion index (SDI)

The signal distortion index measures the amount of distortion in the filtered version of the desired source signal with respect to the clean desired source signal at a reference microphone. It is formulated as:

$$\text{SDI} = \frac{1}{Q}\sum_{q=0}^{Q-1}\left(\frac{\sum_{t=qL}^{(q+1)L-1}|\hat{x}_d(t)-x_{1,d}(t)|^2}{\sum_{t=qL}^{(q+1)L-1}|x_{1,d}(t)|^2}\right), \tag{18}$$

where $\hat{x}_d(t)$ is the filtered version of the desired source signal and $x_{1,d}(t)$ is the clean speech signal of the desired source at a reference microphone. The variables q and L are defined similarly as before.

Performance	$N=3$		$N=6$	
	$d=3$cm	$d=6$cm	$d=3$cm	$d=6$cm
IR [dB]				
NR [dB]				
SDI				

Table 1: Performance Analysis for the DSB with varying number of microphones (Number of Mics) ($N=3$ and 6) and inter-microphone distance (Mic Distance) ($d=3$ cm and 6 cm). Look Direction - **Broadside**

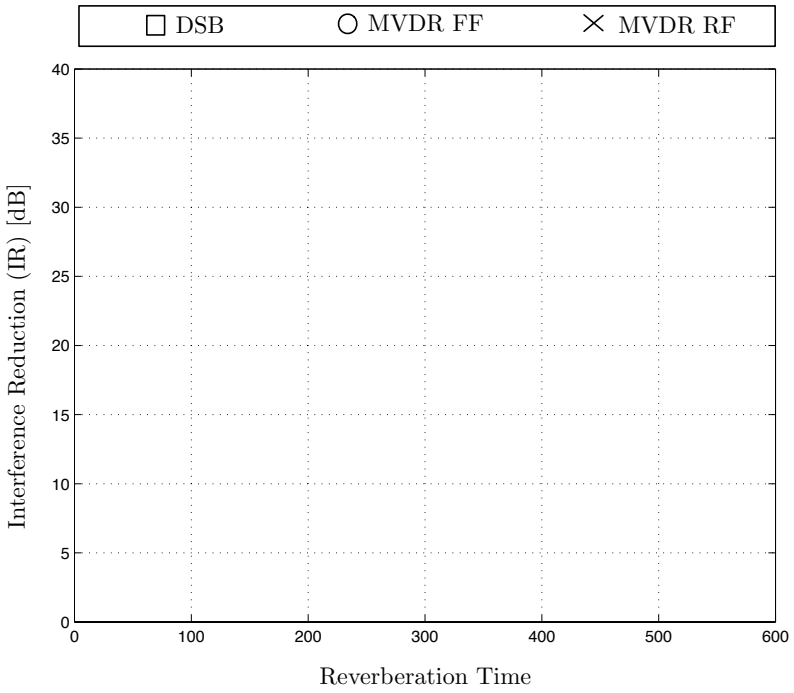

Figure 9: Figure for plotting interference reduction (IR). Look-Direction - **Broadside**

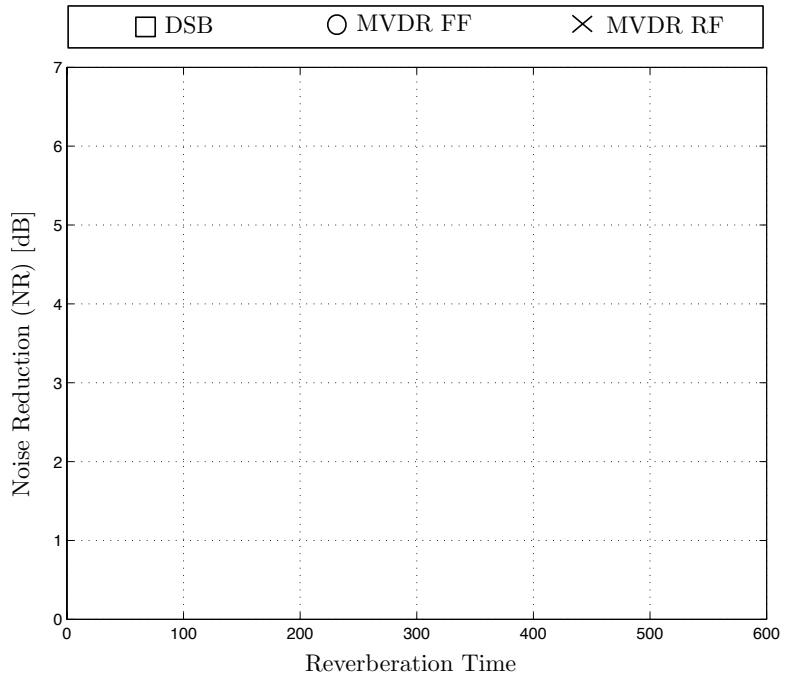

Figure 10: Figure for plotting noise reduction (NR). Look-Direction - **Broadside**

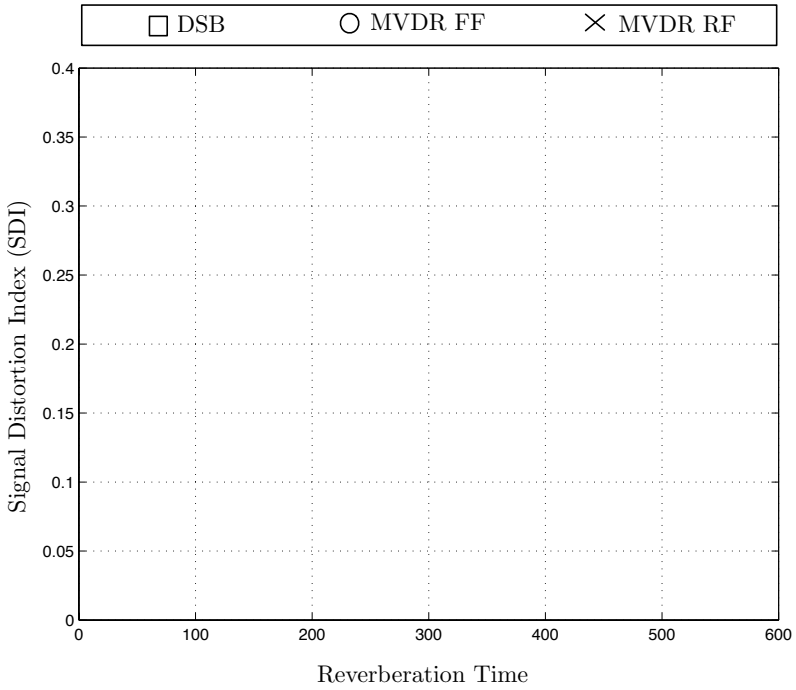

Figure 11: Figure for plotting signal distortion index (SDI). Look-Direction - **Broadside**

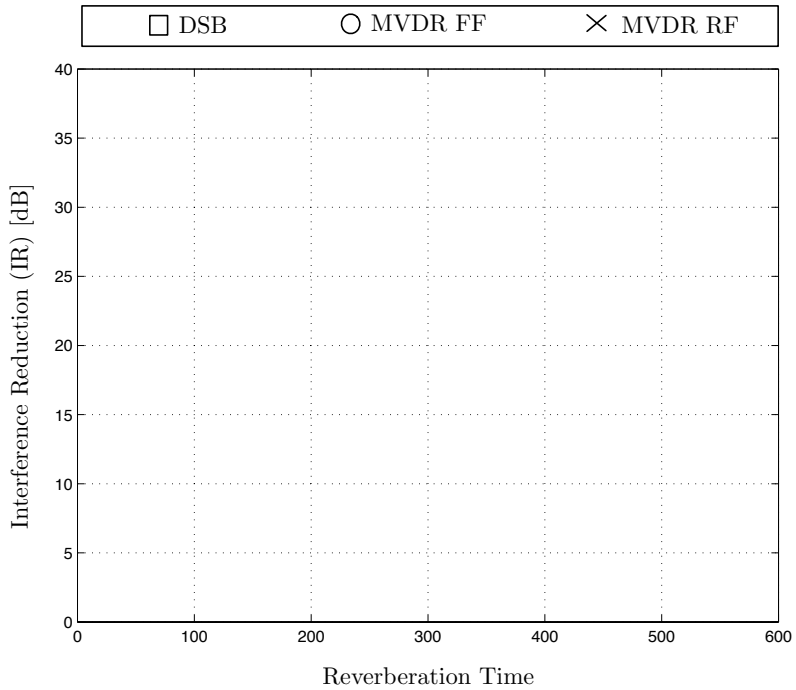

Figure 12: Figure for plotting interference reduction (IR). Look-Direction - **Endfire**

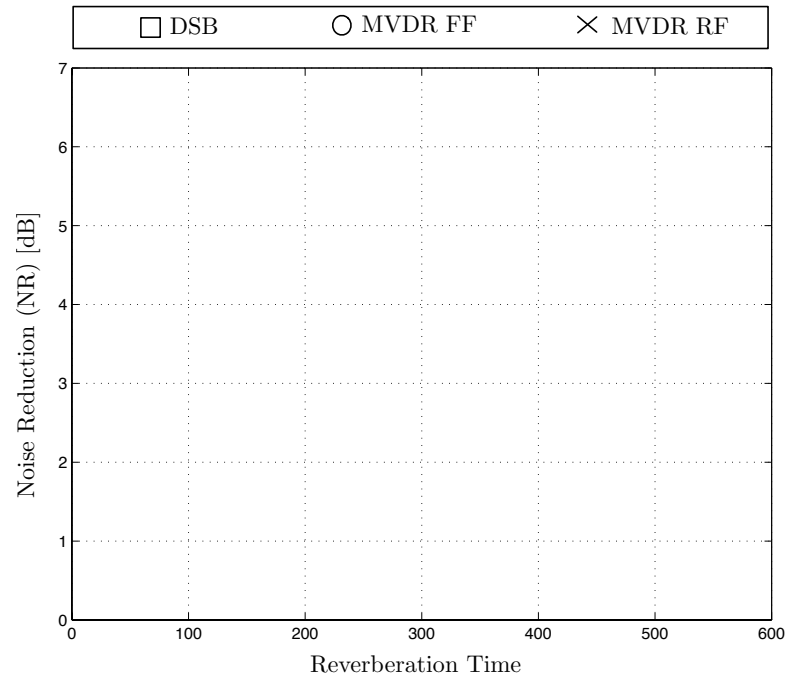

Figure 13: Figure for plotting noise reduction (NR). Look-Direction - **Endfire**

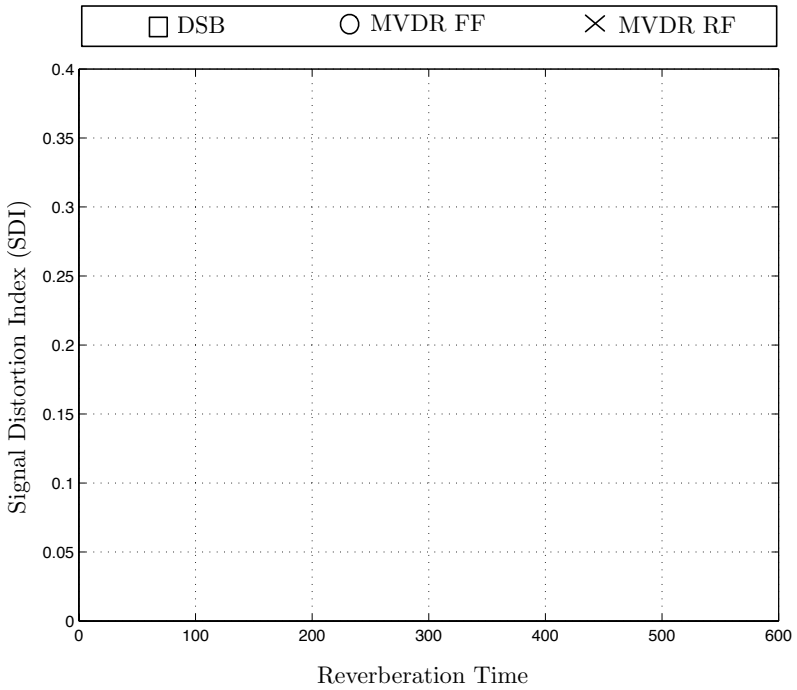

Figure 14: Figure for plotting signal distortion index (SDI). Look-Direction - **Endfire**