Philipp Merz, BSc.

Speech Signal Enhancement for In-Ear Headphones

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Elektrotechnik-Toningenieur

submitted to

Kunstuniversität Graz

Supervisor

Univ.-Prof. Dr. Alois Sontacchi

Graz, August 2021

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Date	Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Datum	Unterschrift

Abstract

In-Ear headphones that can be used for telephony often have two microphones on the outside to enhance the user's speech signal by means of beamforming. If such an earphone also has hybrid active noise cancellation, it will also contain a third microphone on the inside, facing the ear canal. The aim of this thesis is to construct a beamforming system from the inside microphone and only one outside microphone, reducing the number of required microphones on such an earphone to two. To do so, the general properties of first order microphone arrays and the attenuation of the signal arriving at the inside microphone are studied and measured. Based on the results, an adaptive system is designed which compensates this attenuation for arbitrary earphone wearing conditions and maximizes the noise suppression by steering the beam pattern. The achievable SNR improvement strongly depends on the direction and spectrum of the disturbance. With pink noise and a source position behind the user up to 10dB improvement can be achieved, the median over all directions and wearing conditions is 3.3dB.

Kurzfassung

In-Ear Kopfhörer mit Telefonie Funktion haben häufig zwei Mikrofone an der Außenseite, um mittels Beamforming das Sprachsignal des Nutzers zu verbessern. Verfügt ein solcher Kopfhörer außerdem über hybride, aktive Geräuschunterdrückung, so ist ein weiteres Mikrofon im Inneren, in Richtung Gehörgang zeigend notwendig. Ziel dieser Arbeit ist es ein Beamforming System zu entwerfen, dass mit dem inneren und nur einem äußeren Mikrofon arbeitet und damit die Anzahl der benötigten Mikrofone für einen solchen Kopfhörer auf zwei reduziert. Dazu werden zunächst die allgemeinen Eigenschaften von Mikrofonarrays erster Ordnung und die Dämpfung des Signals am innenliegenden Mikrofon untersucht und vermessen. Mit den Ergebnissen wird ein adaptives System entwickelt, dass die Dämpfung bei beliebigen Tragesituationen kompensieren kann und die Geräuschunterdrückung durch Ausrichtung der Richtcharakteristik maximiert. Die damit erreichte SNR Verbesserung ist stark von Richtung und Spektrum des Störgeräuschs abhängig. Bei pinkem Rauschen aus einer Position hinter dem Nutzer können bis zu 10dB Verbesserung erreicht werden, im Median über alle Richtungen und Tragesituationen werden 3.3dB erreicht.

Contents

List of Figures

List of Tables

1. Introduction

The subject of this thesis is the study of feasibility, implementation and evaluation of an algorithm for enabling a differential microphone array with an earphone's in-ear microphone.

Recent models of true-wireless in-ear headphones are packed with various sensors, including typically three microphones. Two of which are located on the outside and are used as an array for telephony, the third one is facing the ear canal and is used for active noise cancellation.

In some acoustic designs for such earphones, especially the loose-fit type (those without a silicone ear tip, such as the Apple Airpods) an opening of the cavity in front of the speaker connects the outside to the ear-canal and consequently to the in-ear microphone. The hypothesis of this thesis is: the acoustical path from outside to in-ear microphone is well defined and can be compensated so that the in-ear microphone can be used as part of a microphone array with one outside microphone.

The benefit of this is would be that only one outside microphone is required for telephony, saving cost and valuable space on the earphones' miniature design.

The algorithm that was developed uses a two-stage approach to enable the differential microphone array. A first stage to compensate the acoustic attenuation that the in-ear microphone is subject to and a second stage that adjusts a delay to steer the beam pattern null in the direction of the biggest noise interference. Unlike many other algorithms on the subject of microphone arrays, this one does not use block processing. Instead, the algorithm was designed to match the specific architecture of the ams AS3460 audio chip, which is a tandem system of one DSP optimized for fast and efficient calculation of a small set of filtering operations and another DSP used for logic and adaption operations.

Structure of the thesis The first chapter introduces the general theory of first order microphone arrays, containing a comparison of additive and subtractive arrays with their individual advantages and drawbacks. This is followed by a chapter dealing with the acoustic properties of loose-fit in-ear headphones. A lumped parameter model is used to study the passive attenuation of earphones. The third chapter deals with characterizing an earphone and taking the theoretical findings from the first two chapters to simulate, based on the characterization measurements, an ideal differential microphone array with the in-ear microphone in the frequency domain. Analysis of the results of these simulations are the basis for the algorithm design, which is explained in chapter five. The algorithm is put to the test in chapter six, where the evaluation of an extensive measurement series attempts to capture the performance of the algorithm. In the appendix, one can find additional measurement data, algorithm implementation details and a description of a filter simulation tool. Each chapter features an introduction and a short summary named „chapter contents“ to ease navigating between chapters.

2. First Order Microphone Arrays

Introduction A first order microphone array is an arrangement of two omnidirectional microphones, which are placed in close proximity so that they record the same sound field at two different positions. This spatial sampling of the sound field makes it possible to obtain a directional response from otherwise omnidirectional microphones. The many different approaches for designing such a system can be classified into two categories:

- Additive arrays, in which signals from the desired direction are time-aligned to cause constructive interference. Signals from other directions are attenuated by lower degrees of constructive interference down to destructive interference.
- Differential arrays, in which the microphones are placed at such close proximity that their difference becomes an approximation of the spatial derivative of the acoustic pressure field, which is directional.

Chapter Contents The following chapter is dedicated to describing the concept, frequency response and properties for the simplest example out of each category: the delay and sum beamformer and a basic differential microphone array.

2.1. General Set-Up

Geometry For most applications of microphone arrays it is sufficient to assume that all sound sources are in the far field of the array. Therefore, a plane wave model can be used to analyze the behavior of the array.

Figure 2.1 shows a first order microphone array consisting of

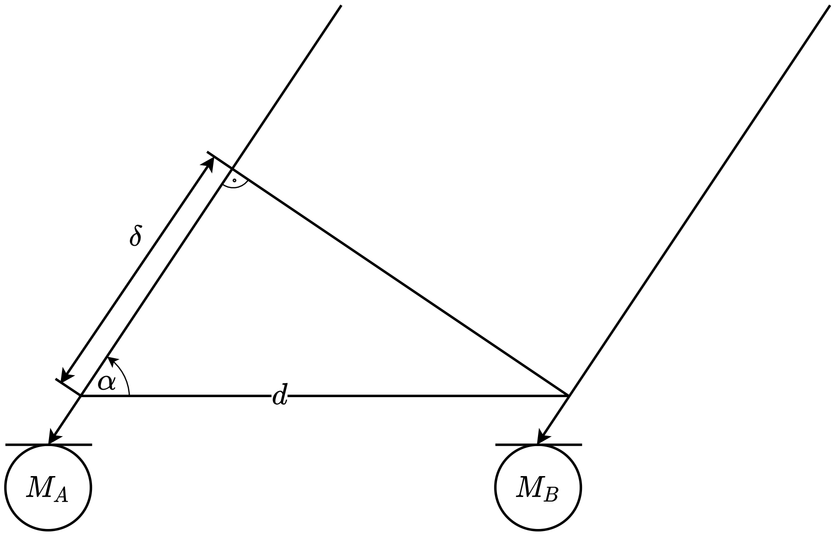

Figure 2.1.: General first order microphone array geometry.

- two microphones, labeled as M_A and M_B,
- at a relative distance or also called aperture of d,
- with their output signals being $x_A(t)$ and $x_B(t)$,
- and a plane wave carrying the signal $s(t)$
- arriving at the reference microphone M_B at $t = 0$
- with an angle of incidence α.

Signals Depending on the angle of incidence at which the plane wave arrives at the array, it does not arrive at both microphones at the same time. In figure 2.1 the wave travels the additional distance of δ to arrive at microphone M_A compared to M_B. It is visible that the microphone output signal $x_A(t)$ only differs from $x_B(t)$ by a delay Δ, that the path difference δ introduces. This path difference and therefore, also the delay Δ is a function of the angle of incidence:

$$\Delta(\alpha) = \frac{\delta(\alpha)}{c} = \frac{\cos(\alpha)d}{c} \tag{2.1}$$

Where c is the speed of sound.

For calculating the interference that occurs when the two microphones' signals are mixed together it is helpful to express them in the frequency domain. The Fourier transform of a signal will be denoted using the same symbol but capitalized. In the frequency domain the delay can be expressed by a complex exponential:

$$\begin{aligned} X_B(j\omega) &= S(j\omega) \\ X_A(j\omega,\alpha) &= S(j\omega)e^{-j\omega\Delta(\alpha)} \end{aligned} \tag{2.2}$$

2.2. Delay and Sum Beamformer

Concept The delay and sum beamformer aims to achieve perfect constructive interference for a sound arriving from one direction at the array. Sounds from all other directions undergo lower degrees of interference, down to destructive interference. The direction of maximum gain can be controlled by adjusting the delay.

Frequency Response To achieve perfect constructive interference for a sound arriving from direction α at the array, the delay $\Delta(\alpha)$ needs to be compensated at microphone M_B's signal. This is done using a fractional delay of τ. The output of the delay and sum beamformer is the signal of one microphone mixed with the other's delayed signal:

$$\begin{aligned} Y(j\omega,\alpha) &= X_B(j\omega)e^{-j\omega\tau} + X_A(j\omega) \\ &= S(j\omega)e^{-j\omega\tau} + S(j\omega)e^{-j\omega\Delta(\alpha)} \\ &= S(j\omega)e^{-j\omega\tau}\left(1 + e^{j\omega(\tau-\Delta(\alpha))}\right) \end{aligned} \tag{2.3}$$

The directional frequency response is defined as:

$$|H(j\omega,\alpha)| = \left|\frac{Y(j\omega,\alpha)}{S(j\omega)}\right| \tag{2.4}$$

$$|H(j\omega,\alpha)| = \sqrt{2 + 2\cos\left(\omega\left(\tau - \frac{\cos(\alpha)d}{c}\right)\right)} \tag{2.5}$$

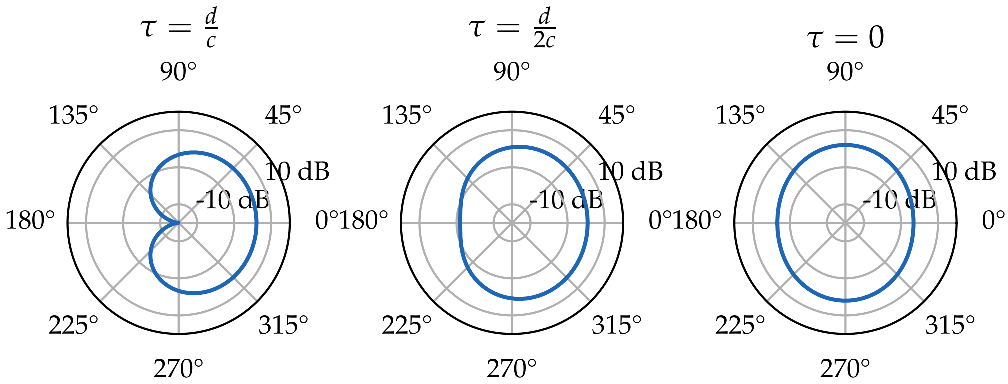

Figure 2.2.: Effect of varying the delay time τ for a delay and sum beamformer with an aperture of $d = 10$cm at a frequency of 800 Hz

The impact of the various parameters on the beamformer's properties can be studied based on this frequency response.

Properties Varying the delay time controls the steering direction, but also changes the beam pattern. At $\tau = 0$ it is almost an omnidirectional pattern and fades towards a cardioid at the maximum delay of $\tau = \frac{d}{c}$. This is shown in figure 2.2.

For low frequencies with long wavelengths compared to the aperture, there is almost no directionality. Once the wavelength becomes too small compared to the aperture, aliasing occurs and distorts the beam pattern (see figure 2.3 and 2.4). This means that the beamformer's bandwidth is fixed by its aperture.

A way of expressing the directivity is to calculate the ratio between the gain at the steering direction α_0 and the gain averaged over the whole space. This is called the directivity factor: (Benesty and Chen, 2012)

$$G(\alpha_0) = \frac{|H(j\omega, \alpha_0)|^2}{\int_{-\pi}^{\pi} |H(j\omega, \alpha)|^2 d\alpha} \tag{2.6}$$

Where α is the angle of incidence. And derived from it is the directivity

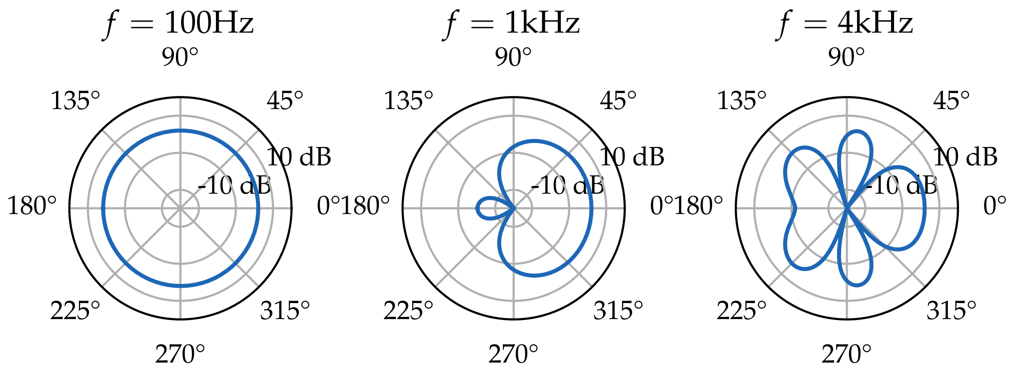

Figure 2.3.: Effect of varying the frequency for a delay and sum beamformer with an aperture of $d = 10$cm and $\tau = \frac{d}{c}$. There is no directionality for low frequencies while at higher frequencies spatial aliasing occurs.

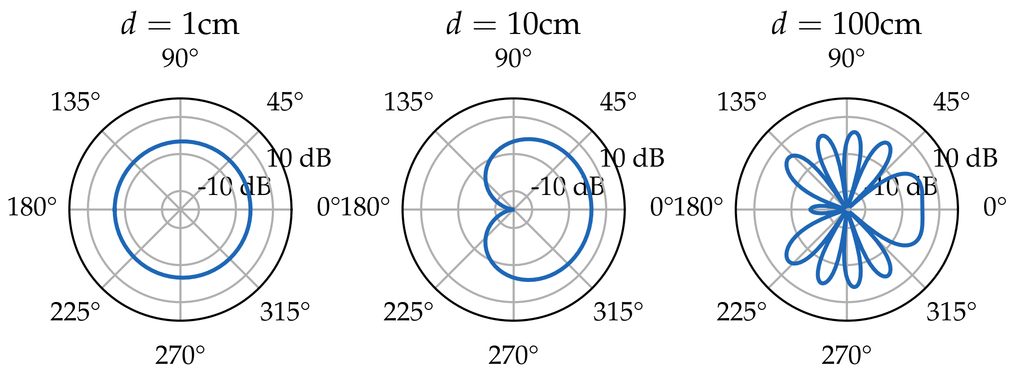

Figure 2.4.: Effect of varying the aperture for a delay and sum beamformer at a fixed frequency of 1kHz and $\tau = \frac{d}{c}$. There is no directionality when using a small aperture while at big apertures spatial aliasing occurs.

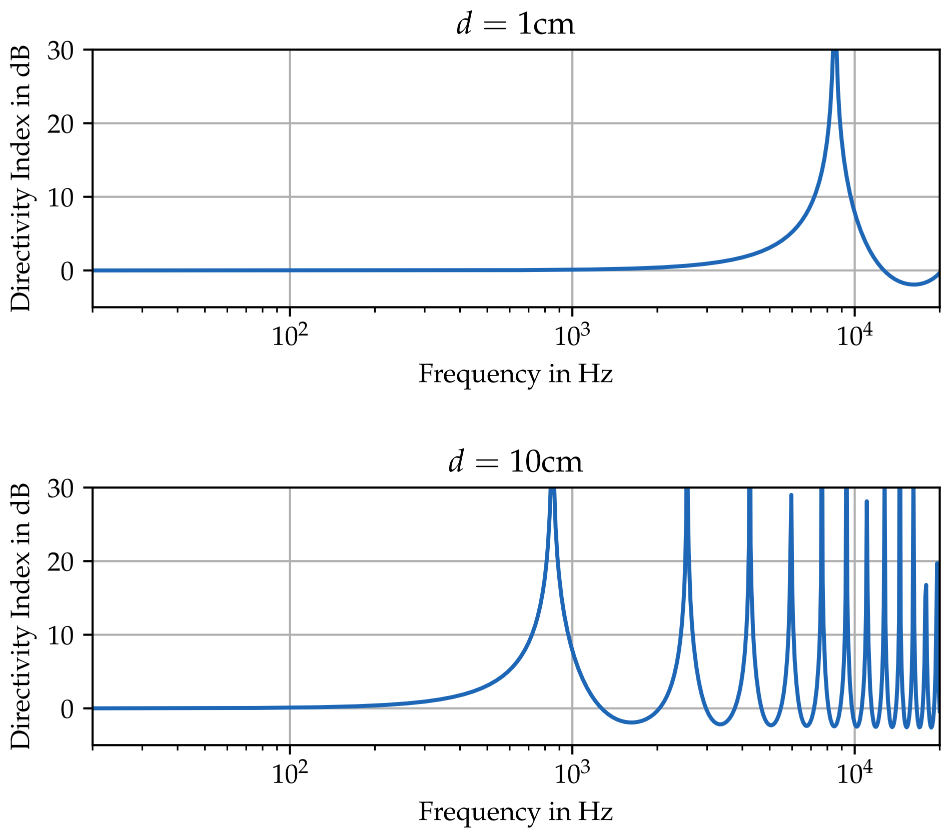

Figure 2.5.: Directivity Index for two delay and sum beamformers with different apertures. From this diagram one can read from which frequency some directionality can be achieved and from which frequency aliasing occurs (the first maximum)

index:

$$D(\alpha_0) = 10 \log_{10}(G(\alpha_0)) \tag{2.7}$$

A numerical evaluation of this is shown in figure 2.5. It can be used to estimate the bandwidth of the beamformer: At an aperture of $d = 10$cm and requiring at least the directivity index of a standard cardioid (4.08dB) the bandwidth of the delay and sum beamformer would be 584Hz to 848Hz.

2.3. Differential Microphone Array

Concept The differential microphone array (DMA) uses a much smaller aperture than the delay and sum beamformer and subtracts signals instead of adding them. This is because it works in the opposite way compared to the delay and sum beamformer. Its aim is to achieve perfect *destructive* interference for a sound arriving from one direction at the array, i.e. suppressing sounds from one direction. (Buck, 2002)

Frequency Response To achieve perfect destructive interference for a sound arriving from direction α, the delay $\Delta(\alpha)$ needs to be compensated at microphone M_A's signal before it gets subtracted from M_B's signal:

$$Y(j\omega, \alpha) = X_B(j\omega)e^{-j\omega\tau} - X_A(j\omega) \tag{2.8}$$

$$H(j\omega, \alpha) = \frac{Y(j\omega, \alpha)}{S(j\omega)} = e^{-j\omega\tau}\left(1 - e^{j\omega(\tau - \Delta(\alpha))}\right) \tag{2.9}$$

Since the aperture is very small compared to the wavelength, the approximation $e^x \approx 1 + x$ can be used to calculate an approximation of the directional response in which the impact of the parameters is more obvious:

$$|H(j\omega, \alpha)| \approx \omega\left(\tau - \frac{\cos(\alpha)d}{c}\right) \tag{2.10}$$

With $\tau = \frac{d}{c}$ meaning that the null is steered to the $0°$ direction (end-fire configuration), there is a cardioid pattern:

$$|H(j\omega, \alpha)| \approx \frac{\omega d}{c}(1 - \cos(\alpha)) \tag{2.11}$$

Properties Often first order DMAs are used with a fixed steering direction and a steerable beam pattern is achieved by constructing two cardioid DMAs in a back to back configuration. (Elko and Anh-Tho Nguyen Pong, 1995) Like at the delay and sum beamformer, varying the delay time for the DMA also changes the beam pattern. Here it fades from a figure of eight to a cardoid, while not the direction of maximum gain is steered but the

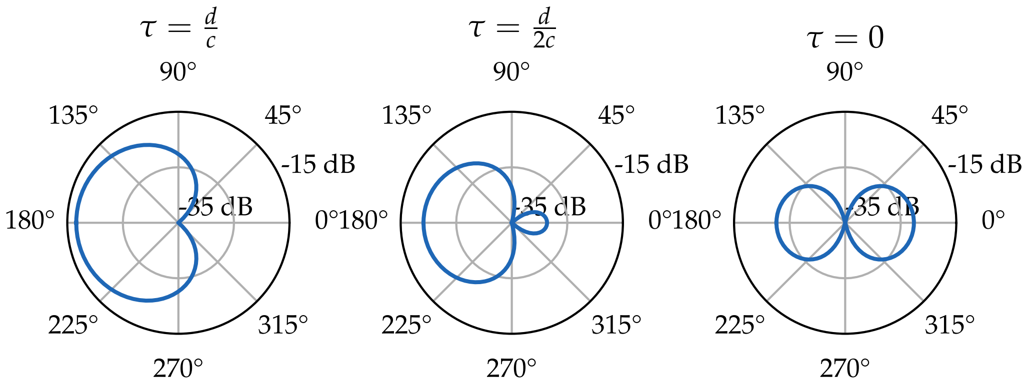

Figure 2.6.: Effect of varying the delay time τ for a differential microphone array with an aperture of $d = 5$mm at a frequency of 800Hz

direction of two symmetric nulls, which become a single null at the cardioid pattern. (see figure 2.6)

The DMA's beam pattern is frequency independent up to the aliasing frequency, which is typically much higher than the delay and sum beamformer's due to the small aperture. (see 2.7) Its bandwidth towards lower frequencies is still limited though, because the DMA's sensitivity has a high pass characteristic of 6dB per Octave below the aliasing frequency. Compensating it leads to an amplification of sensor noise. Figure 2.8 shows the sensitivity of a DMA from three directions, without compensation of the high pass. The bandwidth of the DMA is therefore limited by the aliasing frequency due to the aperture and by how much sensor noise amplification (white noise gain) is tolerated.

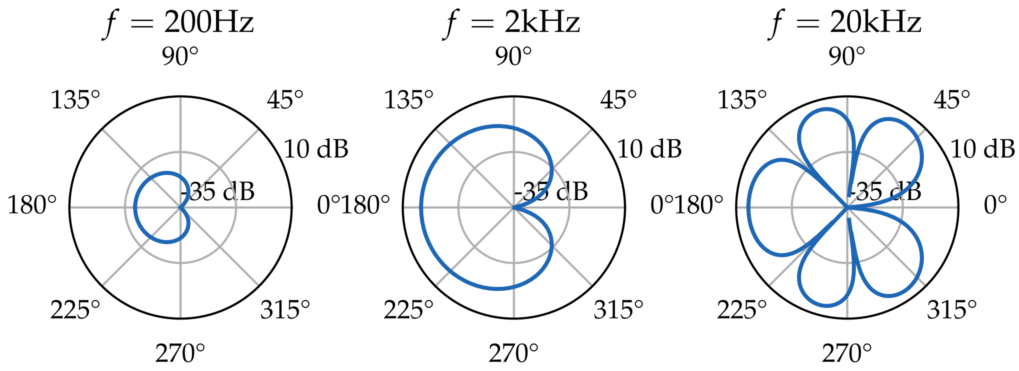

Figure 2.7.: Effect of varying the frequency for a differential microphone array with an aperture of $d = 2$cm and $\tau = \frac{d}{c}$. The beam pattern shape is frequency independent until aliasing occurs, but its scaling changes.

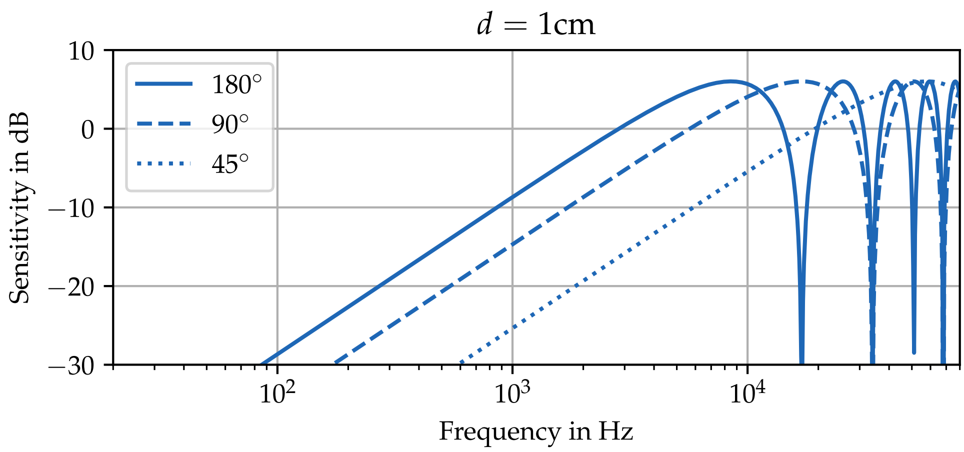

Figure 2.8.: Sensitivity of a differential microphone array with an aperture of $d = 1$cm, with the null steered to 0°, from three different angles

3. Loose Fit In-Ear Headphones

Introduction The term „loose-fit" denotes the kind of in-ear headphone which sits between tragus and antitragus and does not use a foam or rubber part to completely seal the ear canal. (see figure 3.2) Recent popular examples of such earphones are the „Apple Airpods" (see figure 3.1) and the „Microsoft Surface Buds". The loose-fit results in a variable gap or leak between headphone and ear canal, which impacts acoustic properties of the earphone by

- acting as a ventilation and thereby easing the occlusion effect,
- reducing the radiation efficiency of the speaker and thereby reducing its low-frequency response
- and introducing a path for ambient noise to enter the ear and thereby reducing attenuation.

Reducing the occlusion effect and not having a rubber piece exert pressure on the ear canal enhances wearing comfort compared with ear canal sealing headphones[1], while the reduced attenuation and low frequency response are less desirable and require certain measures in the design to compensate, such as active noise cancellation, addition of a bass tube and selection of a larger speaker.

Ams has developed and manufactured a set of prototypes of loose-fit headphones, named „ams loose-fit reference design", (see figure 3.3) which will be used throughout this thesis for all measurements and experiments.

Chapter Contents In this chapter the acoustic elements of a loose-fit in-ear headphone design are examined regarding their impact on the headphone's attenuation. To do so the ams reference loose-fit headphone is used as an

[1] https://ams.com/-/ams-earbud-survey-consumer

Figure 3.1.: Popular example of a loose-fit type in-ear headphone: „AirPods“ by Apple. Image Source: https://commons.wikimedia.org/wiki/File:AirPods.jpg

example and is simplified to a lumped parameter model. Model simulations and measurements show that the front vent and leakage dominate the attenuation.

3.1. Design Overview

Loose fit headphones became popular in 2001 by being the type of headphone included with the first generation of Apple's iPod. Headphone designs from that time typically had the speaker, only covered by a grille, facing the ear canal directly. In recent designs, the speaker is aslant to the ear canal and has a volume of air in front of it which is connected by an outlet to the ear canal.

The ams loose-fit reference design is of the latter type. Figure 3.2 shows a typical wearing condition of it. Besides the speaker, these earphones are also equipped with three microphones for active noise cancellation and telephony and have several openings with acoustic functions. All elements are labeled and can be seen in figure 3.3.

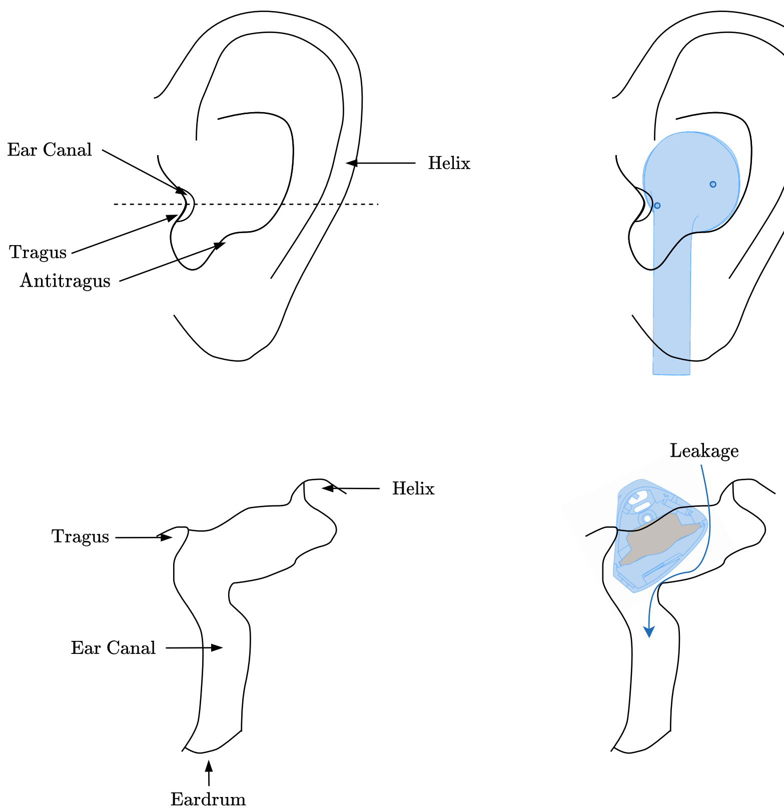

Figure 3.2.: Left: Annotated drawings of the human ear. The dashed line indicates the position of the sectional drawing on the bottom left. Right: the same drawing of a human ear, but wearing the ams loose-fit in-ear headphone. The arrow indicates a typical leakage path.

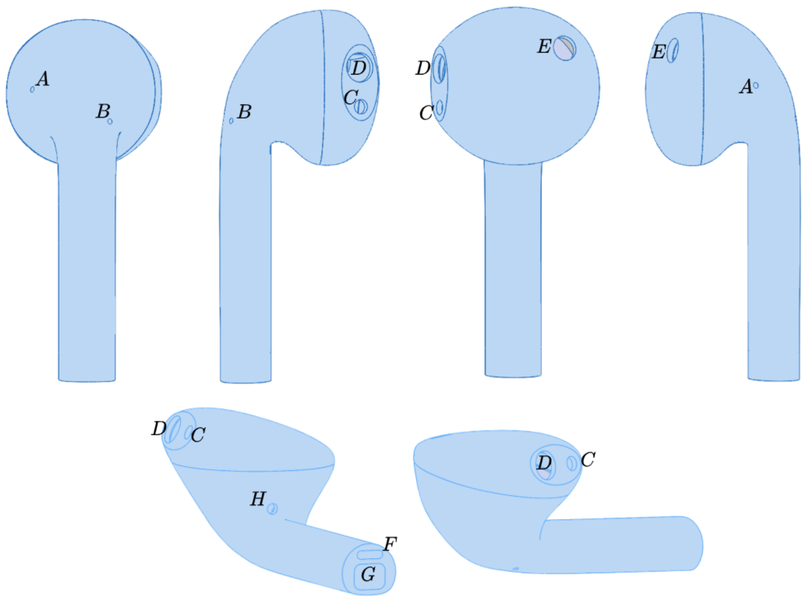

Figure 3.3.: Annotated drawings of the ams reference design loose-fit earbud.
A - speech microphone
B - feed-forward ANC microphone
C - feedback ANC microphone
D - speaker port to ear canal
E - front vent
F - bass tunnel outlet
G - cable outlet
H - rear vent

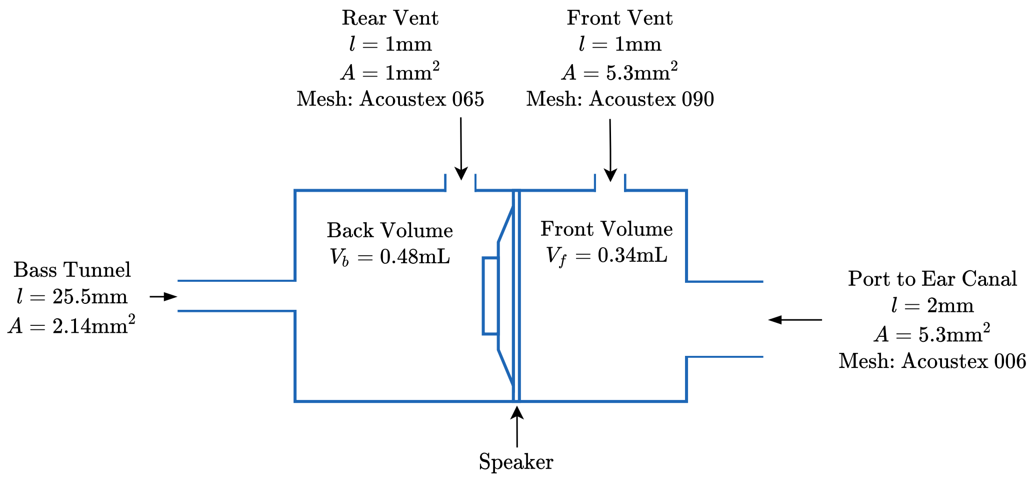

Figure 3.4.: Simplified model of the earbud. The meshes' names refer to the data sheet from Saati (`https://www.saati.com/`). The number at the end is the specific air flow resistance in MKS Rayls.

3.2. Lumped Parameter Model

The behavior of electroacoustic systems is commonly studied using either the finite element method (FEM) or lumped parameter models (LPM). FEM is the more complex and computationally expensive one, but can be highly accurate, while LPM can only yield accurate results in a limited frequency range. This range depends on the size of the system to be studied. This means that for regular loudspeakers LPM can only be used to study the low frequency behavior and the more complex FEM is necessary to study higher frequencies. Since in-ear headphones are small, a LPM can be applied up to 4kHz and therefore is the more convenient choice.

The first step towards the LPM is to identify the components. Front vent, rear vent, bass tunnel, speaker port, speaker and the air volumes behind and in front of the speaker form the acoustically relevant elements of the earphone. These elements are summarized in a simple „box model". Such a model of the earphone can be seen in figure 3.4

The next step is to transform the acoustic elements into lumped electrical components, by using the pU-analogy and relations as described by Beranek, 1954.

Volume A volume or acoustic compliance is represented as a capacitance. Since the acoustic pressure inside the volume is relative to the pressure outside of it, the capacitor also needs a reference point *outside*, which is the ground potential in the electrical domain. The capacitance is given as:

$$C = \frac{V}{\rho_0 c^2} \tag{3.1}$$

where V is the volume, $\rho_0 = 1.2041\text{kgm}^{-3}$ is the air density and $c = 343\text{ms}^{-1}$ the speed of sound at room temperature.

Tube A tube or acoustic mass is represented as an inductance. The inductance of a tube terminating in an infinite baffle at one end is given as:

$$L = \frac{\rho_0(l + 0.85r)}{\pi r^2} \tag{3.2}$$

where r is the radius of the tube and l is its length. By the factor $0.85r$ the end correction is already included.

Mesh A fine meshed piece of cloth acts as an acoustic resistance. The value of its specific air flow resistance Z is usually obtained by measurement or data sheet and the resistance is given by:

$$R = \frac{Z}{A} \tag{3.3}$$

where A is the area of the mesh. For the reference design meshes from Saati were used and the calculations rely on their data sheet.[2]

Speaker For a limited frequency range the speaker can be represented as a series connection of a capacitance, inductance and resistance. The values of these were measured as part of the speaker's Thiele and Small parameters.

[2] http://saati.de/images/filtration/acoustics/tds-acoustex.pdf

Values Table 3.1 shows the component values calculated for the loose-fit reference design.

Table 3.1.: Component values for LPM

Element	Resistance in MΩ		Inductance in H		Capacitance in pF	
Front vent	$R_{fv} =$	16.98	$L_{fv} =$	477.9		
Rear vent	$R_{rv} =$	6.05	$L_{rv} =$	1781.5		
Bass tunnel			$L_{bt} =$	13167.4		
Port to ear	$R_{ep} =$	1.13	$L_{ep} =$	477.9		
Front cavity					$C_{fc} =$	2.40
Rear cavity					$C_{rc} =$	3.39
Speaker	$R_{AS} =$	2.61	$L_{AS} =$	2758.34	$C_{AS} =$	193.00

Structure The structure of the equivalent electrical network is based on Tikander, 2007 and can be seen in figure 3.7. Headphone attenuation is defined as the difference of sound pressure between wearing and not wearing a headphone. Consequently, the SPICE simulation was set-up with one output voltage V_2 representing the open ear and another voltage V_1 representing the ear with headphone. The modified IEC711-coupler model can be seen in figure 3.8 and was adapted from Jønsson et al., 2003 with one additional component to match the leakage adapter used in the ams laboratory. This adapter for an IEC711 type coupler has a set of closeable openings with various cross-sectional areas to simulate different leakages. These opening, as well as the space between the coupler and the rubber seal introduce an additional volume of approximately 2.1mL in front of the coupler, (see figure 3.5 and 3.6) which is modeled by the capacitor C_1 in figure 3.8.

Verification The model is verified by comparing it against measurements. Passive Attenuation was measured using exponential sweeps to obtain first the transfer function $A2E_{\text{open}}(j\omega)$ of a test speaker to the open coupler/artificial ear and second the transfer function $A2E(j\omega)$ of the test speaker to the coupler closed by the earphone. The passive attenuation

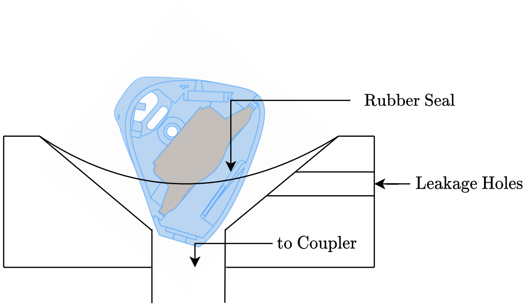

Figure 3.5.: Sketch of the cross-section of the leakage adapter. The leakage adapter allows to make reproducible measurements of loose-fit in-ear headphones in different leakage conditions and is fitted to an IEC711 type coupler.

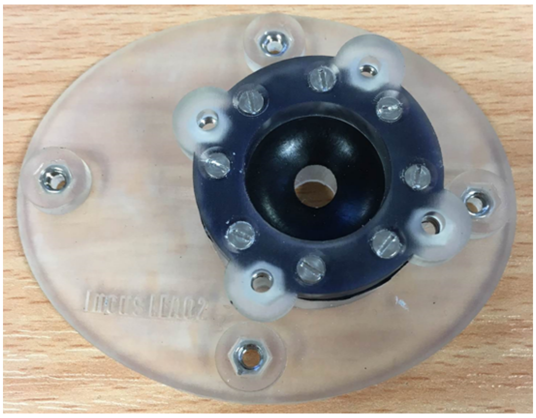

Figure 3.6.: Picture of the leakage adapter.

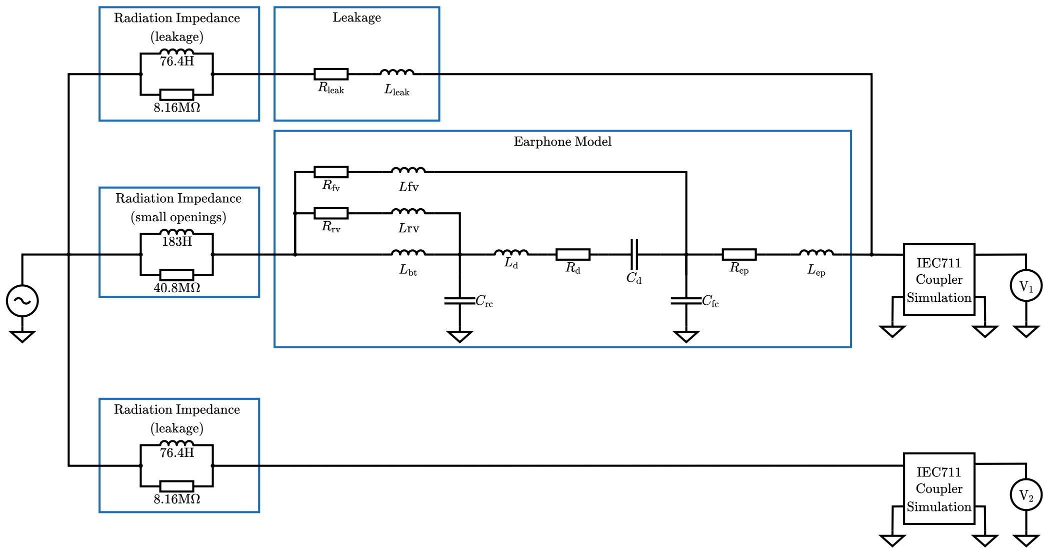

Figure 3.7.: Electrical network for simulating the headphone's passive attenuation.

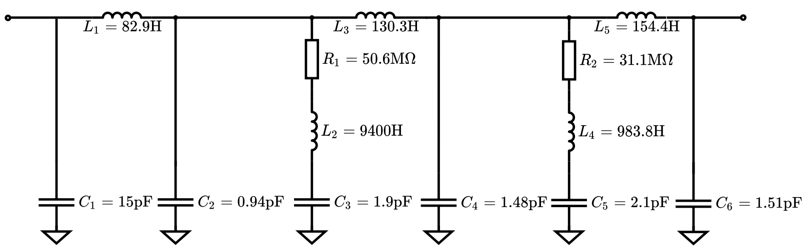

Figure 3.8.: Electrical network for simulating the modified IEC711 coupler

$H(j\omega)$ is the ratio of the two:

$$H(j\omega) = \frac{A2E_{\text{open}}(j\omega)}{A2E(j\omega)} \tag{3.4}$$

The different leakage situations were measured by opening and closing the aforementioned side openings of the leakage adapter. The component values used to simulate the leakage can be seen in table 3.2. The values were determined by matching the 2mm^2 case and from there by halving the values for each doubling of the leakage area.

Table 3.2.: Component values in the leakage path circuit (see figure 3.7) to simulate different amounts of leakage.

Area in mm^2	Component configuration	
0	open leakage path circuit	
2	$R_{\text{leak}} = 7.6\,\text{M}\Omega$	$L_{\text{leak}} = 10.4\,\text{kH}$
4	$R_{\text{leak}} = 3.8\,\text{M}\Omega$	$L_{\text{leak}} = 5.2\,\text{kH}$
8	$R_{\text{leak}} = 1.9\,\text{M}\Omega$	$L_{\text{leak}} = 2.6\,\text{kH}$
28	$R_{\text{leak}} = 380\,\text{k}\Omega$	$L_{\text{leak}} = 520\,\text{H}$

The earphone was measured and simulated in several configurations, i.e. with one or two of the acoustically relevant openings covered. This way it was verified that the model predicts the impact of each component accurately. In figure 3.9 the measurement and simulation of the unaltered earphone are shown, in figure 3.10 the front vent was closed. The simulation matches the measurements up to approximately 4kHz, which is expected. For higher frequencies the wavelengths become too short compared to the earphone's geometry and the assumptions underlying the lumped parameter model are not valid anymore. Further measurements for the verification of the model are shown in Appendix B.

3.3. Passive Attenuation

From the simulation and measurements it was found that the front vent and the leakage have the biggest impact on the earphone's attenuation. (see

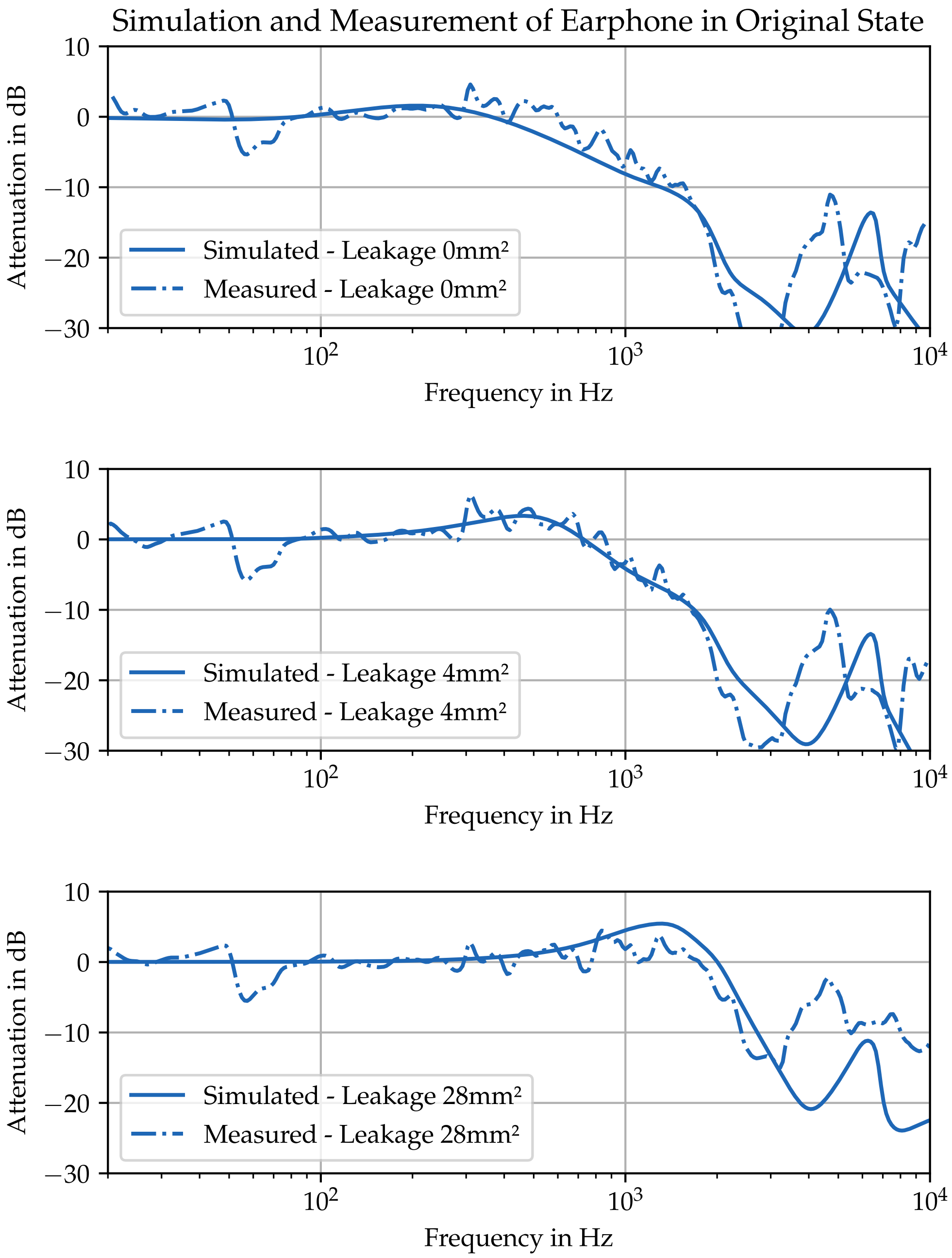

Figure 3.9.: Comparison of simulated and measured passive attenuation of earphone at three different leakages. The earphone was in its original state.

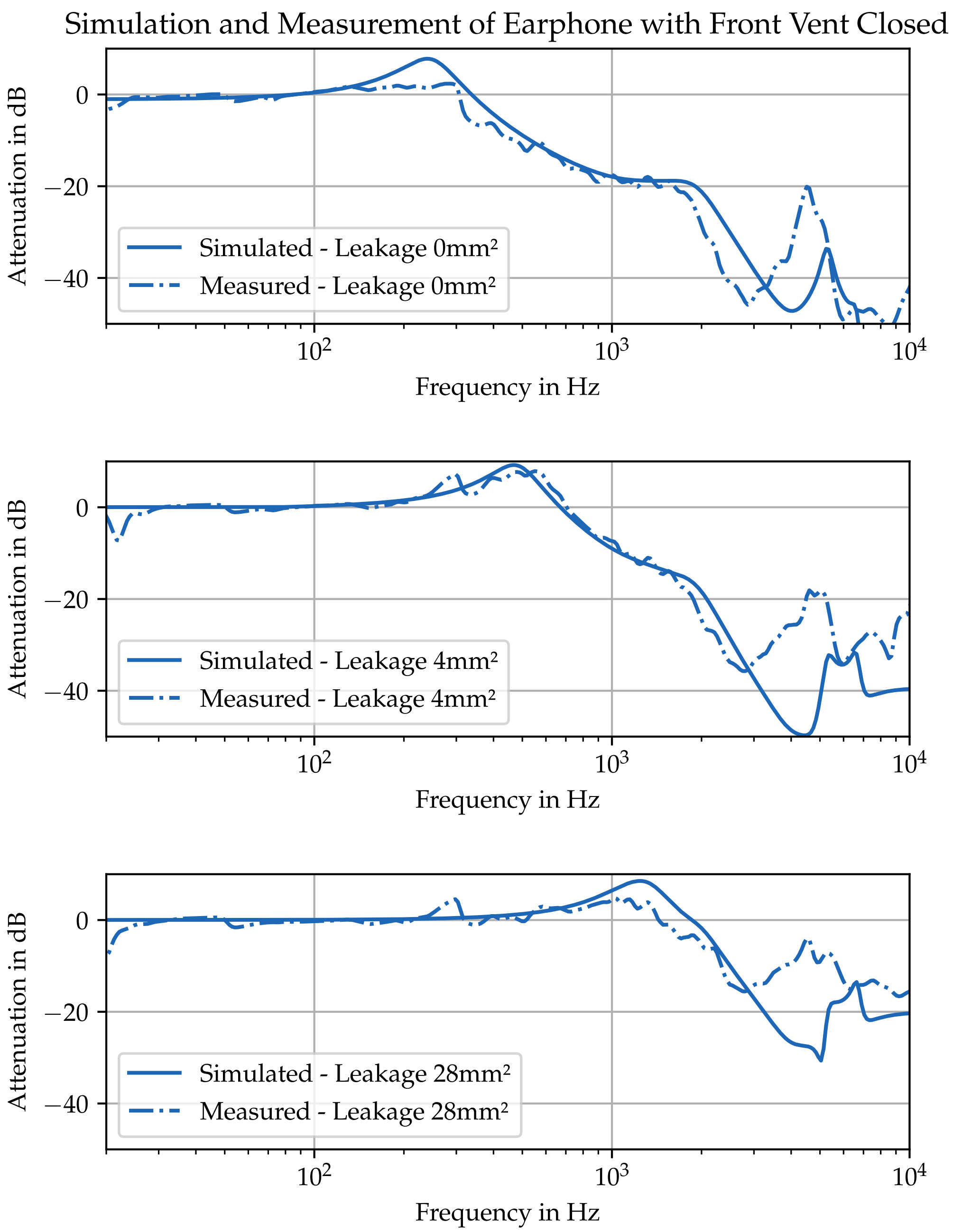

Figure 3.10.: Comparison of simulated and measured passive attenuation of earphone at three different leakages. Here the front vent was turned off in the simulation and covered by putty for the measurement.

figure 3.11 and 3.12) For the design of the beamformer or DMA it would be an advantage to have a single dominant path by which outside sounds arrive at the feedback microphone, because the array would then behave partly like an imagined regular microphone array consisting of the feed forward microphone and a microphone placed at the position where the sound enters the headphone. For low leaks, this is the case and sound enters mostly through the front vent, at higher leakages the leakage becomes the dominant path. In between both paths work simultaneously. This means that the algorithm for a DMA consisting of the feed-forward and feedback microphone not only needs to compensate the attenuation that the feedback microphone is subject to, but must also be able to adapt to changing leakage conditions.

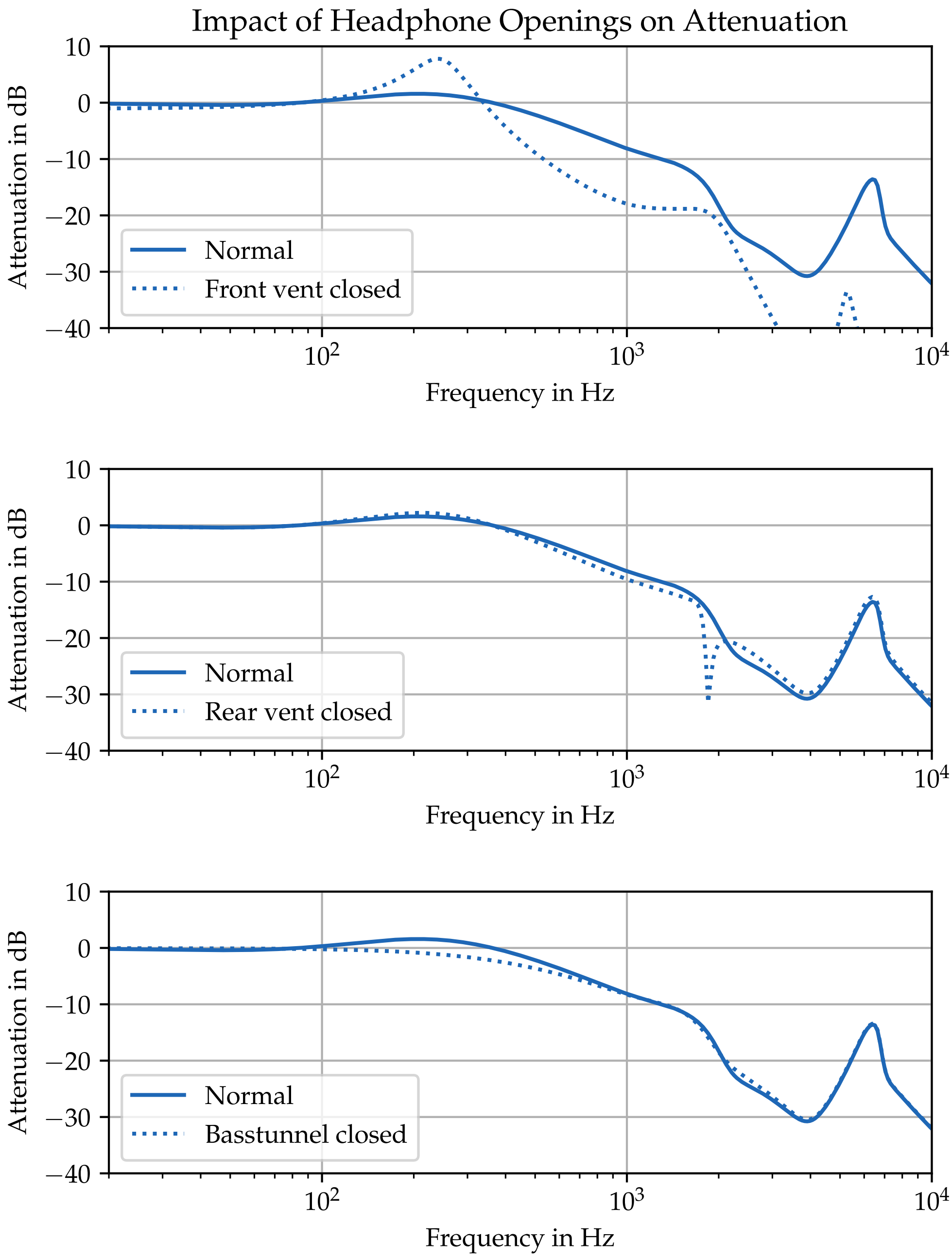

Figure 3.11.: Simulated impact of the headphone's openings on the passive attenuation at low leakage

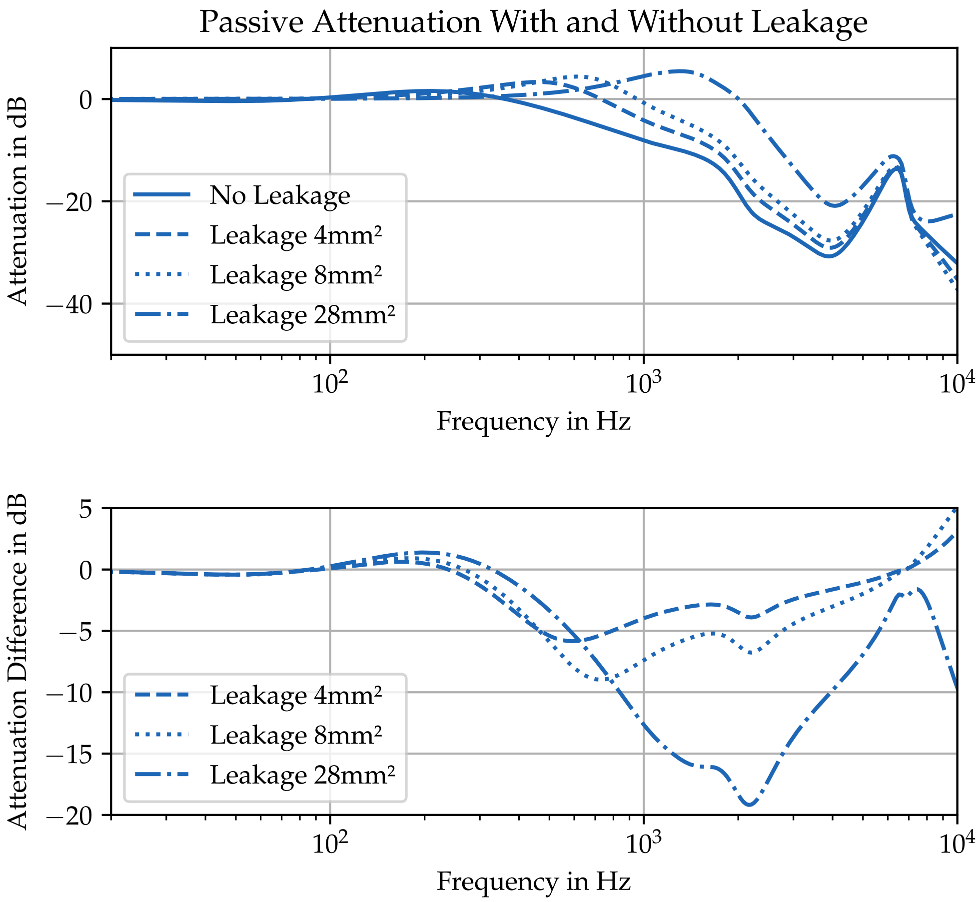

Figure 3.12.: Top: Impact of increased leakage on the passive attenuation. Bottom: Difference in attenuation between 0mm^2 leakage and increased leakages

4. Numerical Evaluation

Introduction Following the theoretical examination of differential microphone arrays and the acoustical properties of the earphone, the next step is to confirm these findings by measurements and derive the properties of the adaptive system to enable the DMA. The relations from section 2 show that a microphone array can be completely described by the microphones' frequency response regarding a plane wave arriving from an arbitrary direction. These direction dependent frequency responses can be measured accurately by performing several frequency sweep measurements with the sound source positioned along a few selected directions. Ideally, the measurement is conducted under free-field conditions with the sound source at a great distance from the array to assure that the plane wave conditions are met. With the facilities available the distance to the speaker was 1m and the room was only damped and not anechoic. Therefore, the conditions were not ideal, but good enough to obtain valid results for the relevant frequency range. The process of this measurement is called characterization. The characterization provides the information on which the design of the algorithm is based.

Chapter Contents A measurement and simulation method for estimating a microphone array's sensitivity or beam pattern is introduced and applied to a regular as well as to the feed-forward to feedback microphone array.

4.1. Measurement Setup

The measurements were conducted in the „acoustic chamber" at the ams audio laboratory, which is a noise isolated and damped room of approximately $9\,\mathrm{m}^2$. At 1.2 m height and 1 m distance to the test speaker, the earphones were placed in a silicone ear, which was connected to ear canal simulator. The silicone ear was placed with the sagittal plane parallel to the floor, so it could be rotated along what would be the elevation angle on an upright standing person. Rotation was performed in 32 evenly spaced increments. An exponential sweep was produced and recorded through the earphone's PDM microphones by the AudioPrecision (TM) acoustic measurement system, which also performed the transformations and smoothing of the transfer function. ApX, the measurement processing software of the AudioPrecision (TM) system, implements smoothing by passing „the raw response data through a constant-Q bandpass filter that is swept with the continuous sweep signal."(*APx500 User's Manual* 2018)

Transfer Functions There are three microphones on the ams loose-fit reference design: the feed-forward microphone (FF), the speech microphone (FF2) and the feedback microphone (FB). (see 3.3) The transfer function from the test speaker („Ambient Noise Source") to each of these shall be called $A2FF(j\omega,\alpha)$, $A2FF2(j\omega,\alpha)$ and $A2FB(j\omega,\alpha)$ where α is the angle of incidence.

Figure 4.1 shows the raw impulse responses at the feedback and feed forward microphone obtained by the measurement system for the 0° direction. Multiple reflections are visible. To rule out that this impacts the results too much, the $\frac{A2FB}{A2FF}$ transfer function was calculated using the complete impulse responses as well as the windowed impulse responses, which exclude all reflections. (the window is indicated in figure 4.1) The results of these calculations can be seen in figure 4.2. The phase deviations in the 400 Hz to 4kHz region are small, therefore the complete response was chosen for the simulations, since it can be obtained more conveniently through the ApX analyzer without additional processing.

The way of calculating this transfer function by deconvolution of one DFT with the other works without using any information about the stimulus signal. Any reflection of the stimulus signal or other disturbances are treated as a part of the stimulus signal. Therefore, such disturbances do not distort the measured transfer function, but instead distort the direction to which the measured transfer function applies. This means that in the presence of a strong reflection or noise source, the measured transfer function is not the one for a stimulus from the intended direction, but instead a mix of transfer functions for the direction of the intended stimulus and direction of the disturbances.

For a point source, which a coaxial speaker as used in this experiment approximately is, the plane wave assumption only holds if the array is in the speaker's far field. This depends on the ratio of the frequency and distance to the speaker. The array is in the far field if: (*DEGA-Empfehlung 101 Akustische Wellen und Felder* 2006)

$$r < \frac{c}{f} \tag{4.1}$$

where $r = 1\,\mathrm{m}$ is the distance between array and speaker, $c = 343\mathrm{ms}^{-1}$ is the speed of sound and f is the frequency. For this setup, the array is in the far field for any frequency above 346 Hz and results below this frequency can be considered imprecise.

The measurement setup can be seen in figure 4.3 and the used equipment is listed in table 4.1. The geometric relations of the angles are displayed in figure 4.4.

Table 4.1.: Equipment list for frequency response measurements

Equipment	Model
Test speaker	KSDigital C8-Coax
Measurement device	AudioPrecision(TM) APx525
Measurement software	AudioPrecision(TM) APx500 5.0
Artificial ear	G.R.A.S. KB5011
Ear canal simulator	G.R.A.S. RA0045

4.2. Characterization of a Regular Microphone Array

The two microphones on the outside of the ams earphone are placed in a distance suitable to be used as a differential microphone array (see figure 3.3)

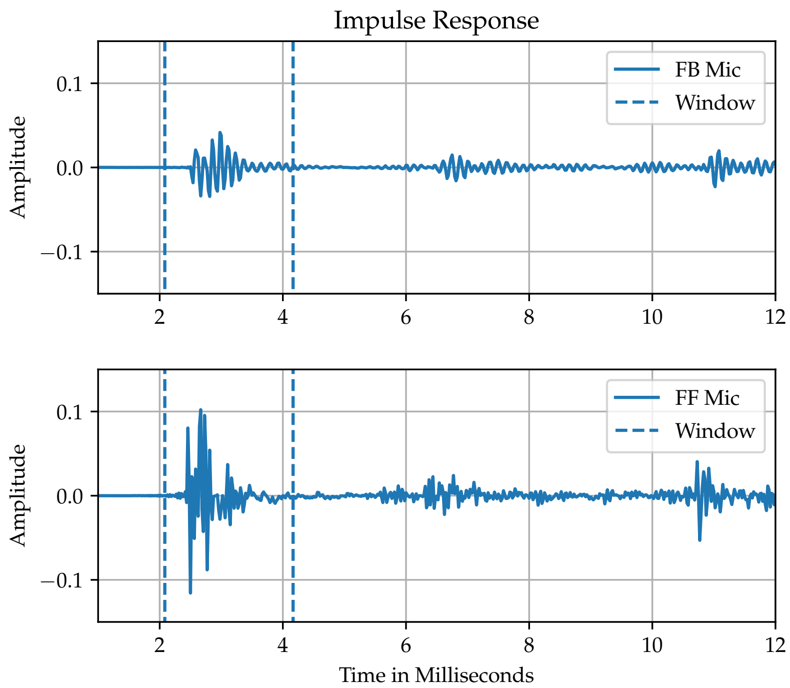

Figure 4.1.: Impulse responses at feedback and feed forward microphone obtained during the characterization measurement. It is visible that the measurement conditions are not free of reflections.

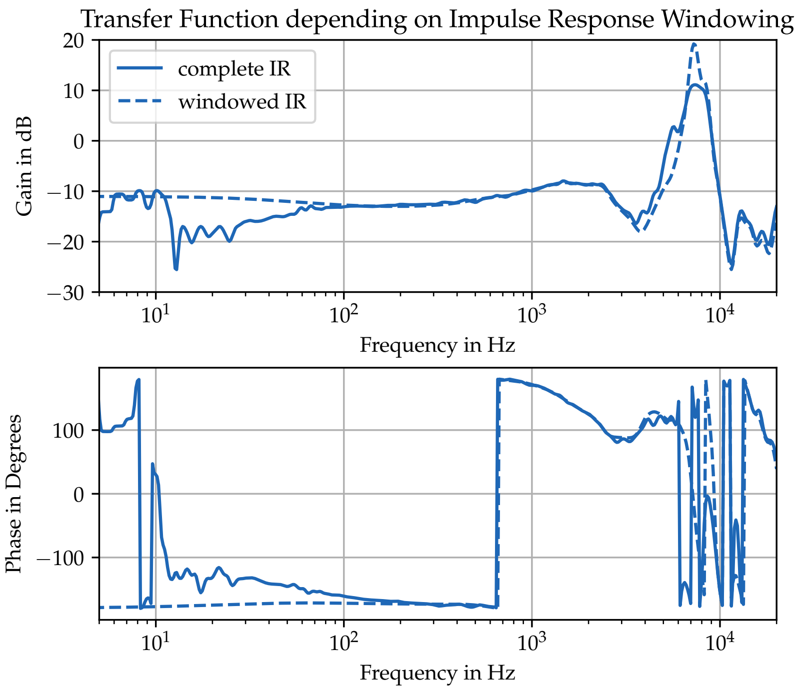

Figure 4.2.: $\frac{A2FB}{A2FF}$ transfer function calculated from the full length impulse response compared with the same transfer function calculated from the shortened impulse response, both smoothed with a 1/9th octave bandwidth.

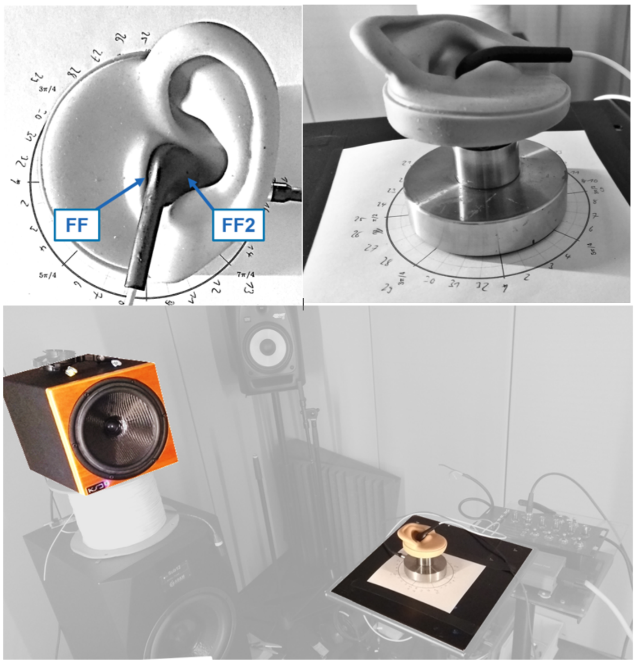

Figure 4.3.: Measurement conditions for the directional characterization

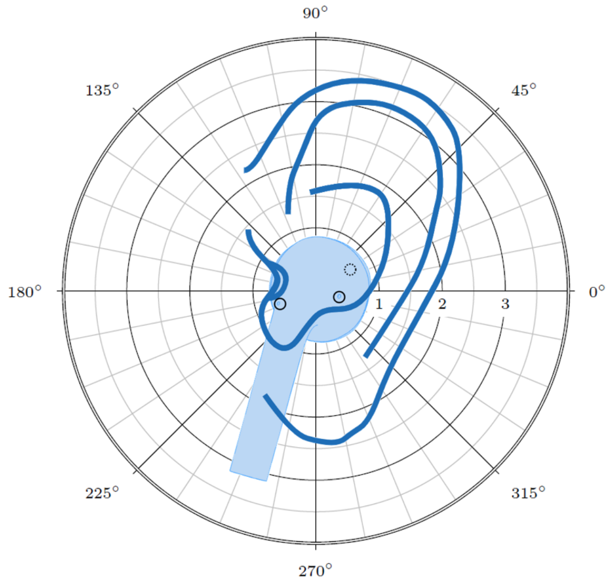

Figure 4.4.: Coordinate system for directional characterization. The position of the front vent on the ear-facing surface of the earphone is indicated with a dashed-line circle.

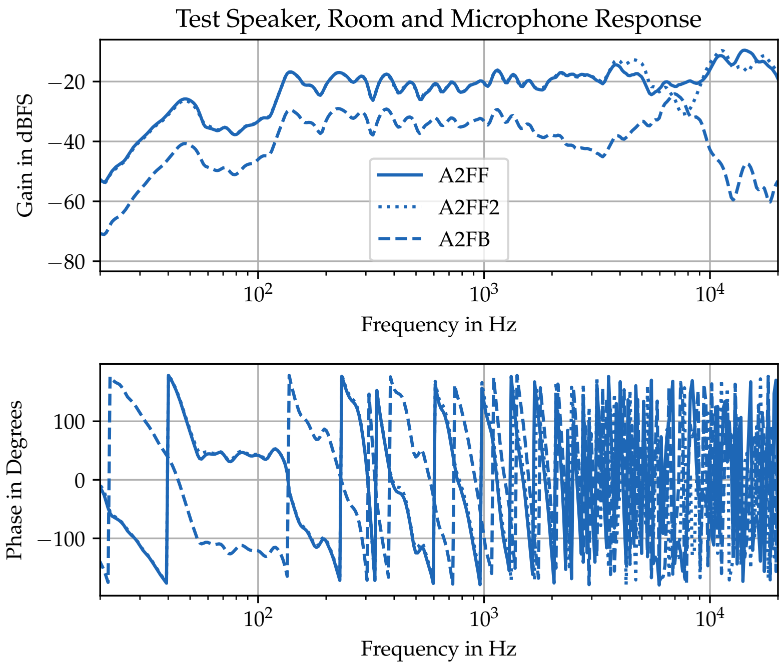

Figure 4.5.: Combined speaker, room and microphone response for each of the three microphones in the earphone, i.e. *A2FF* for the feed-forward, *A2FF*2 for the speech and *A2FB* for the feedback microphone. Visible is that the response of the two feed-forward microphones is consistent up to 4kHz. The feedback microphone has approx. 10dB less sensitivity and is phase inverted.

and are included in the characterization, to serve as an example of a regular first order array to later compare the array consisting of the feed-forward and feedback microphone with, as well as verifying the measurement method.

Transfer Functions Measured are the transfer functions $A2FF(j\omega,\alpha)$, $A2FF2(j\omega,\alpha)$ and $A2FB(j\omega,\alpha)$ and the transfer function from the earphone's speaker to the feedback microphone $D2FB(j\omega)$. As the driver's low frequency response diminishes with increasing leakage, this measurement can be used to observe the amount of leakage. Figure 4.5 shows the ambient transfer functions, figure 4.6 shows the driver response, as seen by the feedback microphone at various leakages.

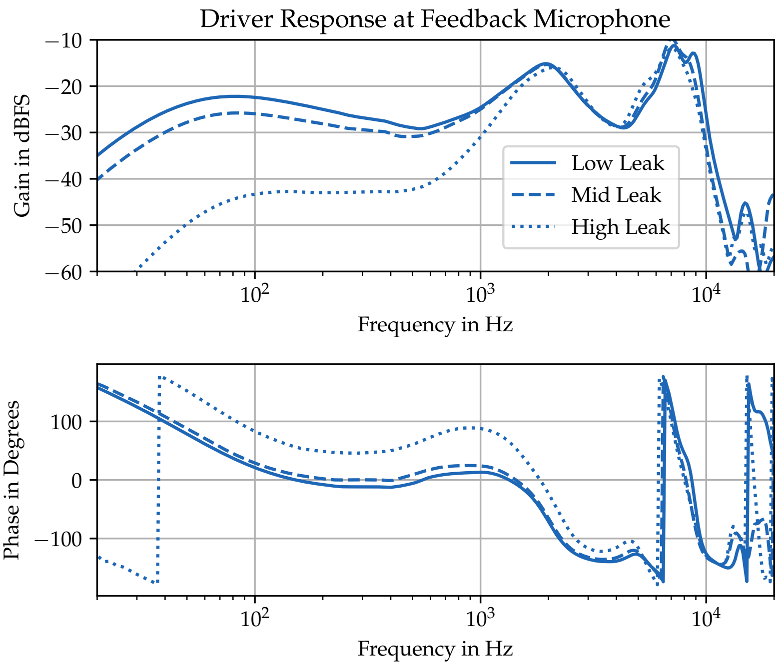

Figure 4.6.: Frequency response of the earphone's speaker as seen by the feedback microphone. Comparing with these curves is a reliable method for determining the leakage state of the earphone in the ear.

Evaluation Between FF and FF2 there is a distance of $11\,\mathrm{mm}$[1] and the delay time used in the simulation therefore is set to $\tau = 0.03\,\mathrm{ms}$[2] (endfire steering). The directional response or sensitivity of a DMA made up of FF and FF2 can then be calculated using equation 2.8:

$$Y(j\omega, \alpha) = A2FF2(j\omega, \alpha) - A2FF(j\omega, \alpha)e^{j\omega\tau} \tag{4.2}$$

No calibration was performed to determine the true transfer function of the ambient stimulus at the earphone, but it was found to be sufficient to compensate speaker and room response by averaging:

$$A2FF_{\text{avg}}(j\omega) = \frac{1}{64}\sum_{i=1}^{32} A2FF(j\omega, \alpha_i) + A2FF2(j\omega, \alpha_i) \tag{4.3}$$

Where α_i is the i^{th} out of the 32 measured directions. Analogous to equation 2.9 the sensitivity of the real DMA is:

$$H(j\omega, \alpha) = \frac{A2FF2(j\omega, \alpha) - A2FF(j\omega, \alpha)e^{j\omega\tau}}{A2FF_{\text{avg}}} \tag{4.4}$$

Figure 4.7 shows the sensitivity for three directions and figure 4.8 shows the beam pattern at three frequencies. While the measurement suffer from a measurement environment that is far from being anechoic, at large the results confirm the theoretic relations studied in section 2.3.

4.3. Characterization of the Feedback Microphone Array

Sensor Imperfections A DMA's performance is sensitive to sensor imperfections, i.e. differences in the microphones' frequency responses. Even

[1]measured on the projection of the earphone onto the horizontal plane of the measurement setup

[2]Due to the very small aperture, the deviations would be small, but in general the delay time for a microphone should be calibrated to the atmospheric conditions. For the conditions in the laboratory $c = 343\,\mathrm{m\,s^{-1}}$ is used.

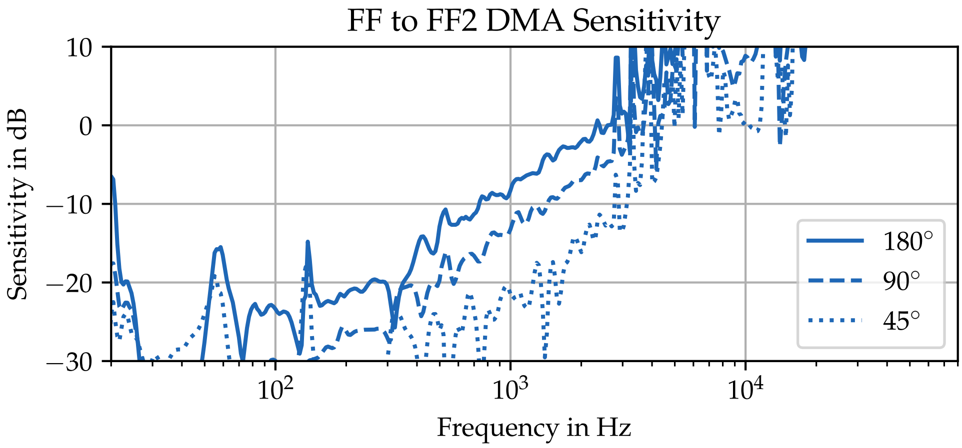

Figure 4.7.: Measured sensitivity of the differential microphone array consisting of the FF and FF2 microphone on the ams loose-fit earphone. Within the usable bandwidth of the DMA, this measurement confirms the simulation shown in figure 2.8. This was measured at a low leakage. The angles refer to those shown in the coordinate system in figure 4.4.

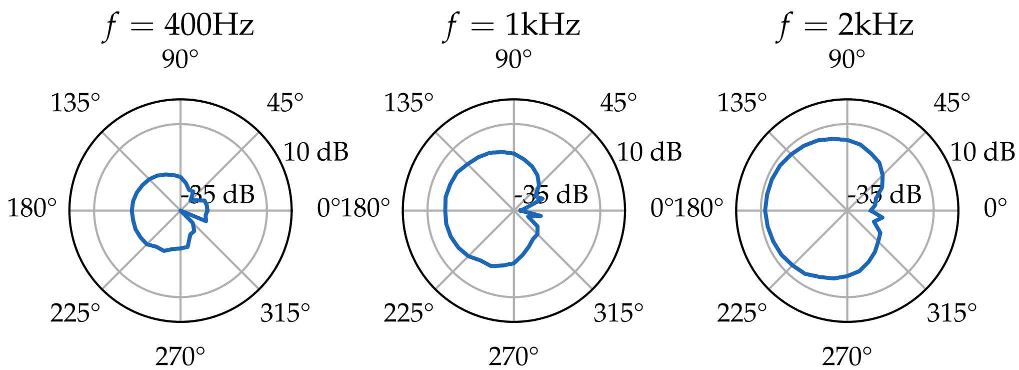

Figure 4.8.: Measured directivity pattern of the feed forward to speech microphone DMA at three center frequencies with a 1/9th octave bandwidth. The angles refer to those shown in the coordinate system in figure 4.4.

small deviations can severely degrade the arrays' performance. Typically, the sensor mismatch is compensated by a calibration procedure or an adaptive scheme. (Buck, 2002)

In the case of using the feed-forward and feedback microphone as an array, the feedback microphone's transfer function deviates by tens of dBs from the feed-forward microphone, namely due to the passive attenuation of the earphone.

Passive Attenuation Approximation A way to capture the passive attenuation as well as all other differences in frequency response between these two microphones is to calculate the passive attenuation approximation:

$$P(j\omega, \alpha) = \frac{A2FB(j\omega, \alpha)}{A2FF(j\omega, \alpha)} \tag{4.5}$$

Figure 4.9 shows this transfer function for three directions.

If, instead of trying to compensate the feedback microphone's frequency response, the feed-forward microphone's signal is filtered with $P(j\omega, \alpha)$ the new formulation of the DMA's sensitivity is equal to a feed-forward noise-canceling problem:

$$H(j\omega, \alpha) = A2FF(j\omega, \alpha)\frac{A2FB(j\omega, \alpha_0)}{A2FF(j\omega, \alpha_0)} - A2FB(j\omega, \alpha) \tag{4.6}$$

Where α_0 is the null-steering direction. This means that a direction α_0 is selected and the feed-forward microphone's signal is filtered by the passive attenuation approximation of that direction, to achieve perfect cancellation for that direction. For other directions, the cancellation will be less effective, due to the phase mismatch. The previously used fractional delay element is now implied in $P(j\omega, \alpha_0)$.

Identifying the array's main axis For the microphone array on the earphone's outside, it was valid to assume that the array's main axis is parallel to the sagittal plane, since it is a standard first order array with visible sensor positions. For the feedback to feed forward microphone array this

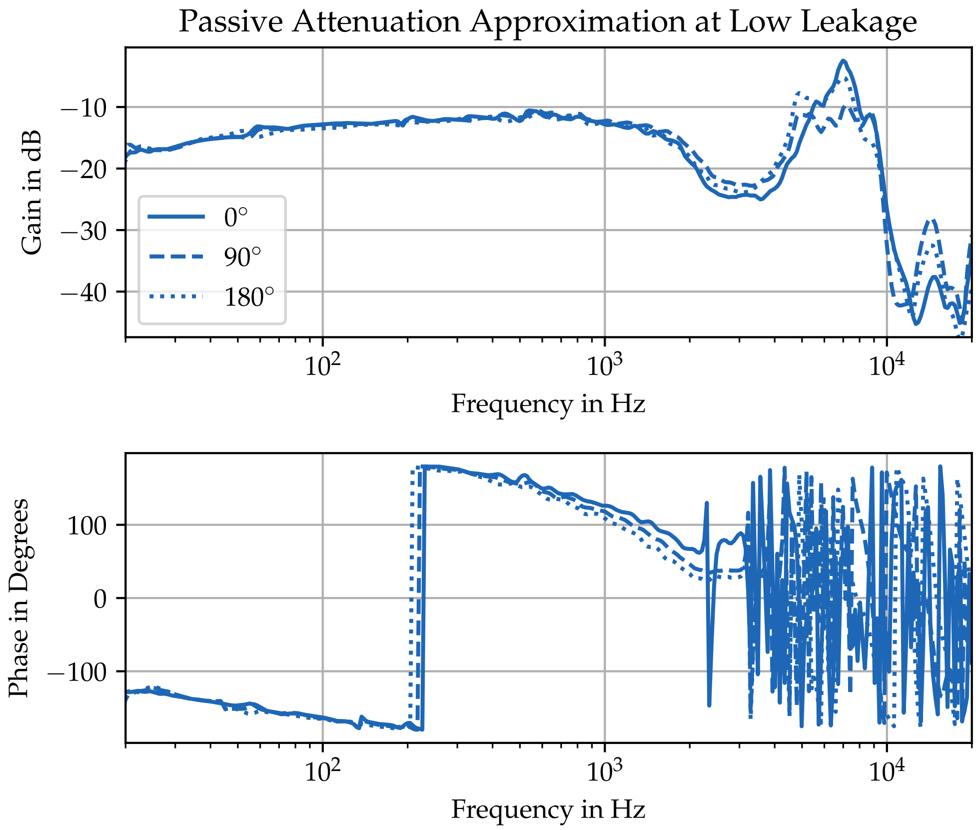

Figure 4.9.: Passive attenuation approximation at low leakage for three directions.

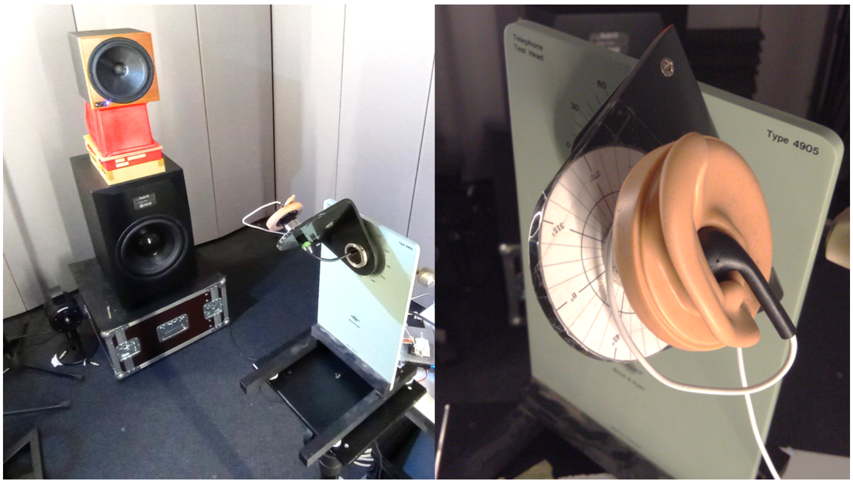

Figure 4.10.: Alternative measurement setup for the directional characterization with one more degree of freedom.

assumption was verified separately. The alternative setup visible in figure 4.10 also allows the rotation of the sagittal plane relative to the speaker. With this, the passive attenuation approximation was recorded for 113 ambient noise source positions covering a hemisphere over the sagittal plane of the ear. The array's axis can be identified by selecting the two transfer functions that span the widest phase range. (see figure 4.11) This is equivalent to finding the two endfire steering directions of the DMA. The results indicate that the main axis is at $0°$ elevation, i.e. the sagittal plane. In azimuth, the endfire directions are at $33.75°$ and $202.5°$, which matches the line from feed forward microphone to front vent (see figure 4.4).

Simulation with ideal Filter In the real application a filter has to be used to approximate $P(j\omega, \alpha_0)$, while for the purpose of simulating the DMA with transfer functions, it can be applied directly. Here, the speaker and room response are compensated by:

$$A2FB_{\text{avg}}(j\omega) = \frac{1}{32}\sum_{i=1}^{32} A2FB(j\omega, \alpha_i) \tag{4.7}$$

$$H'(j\omega, \alpha) = \frac{H(j\omega, \alpha)}{A2FB_{\text{avg}}(j\omega)} \tag{4.8}$$

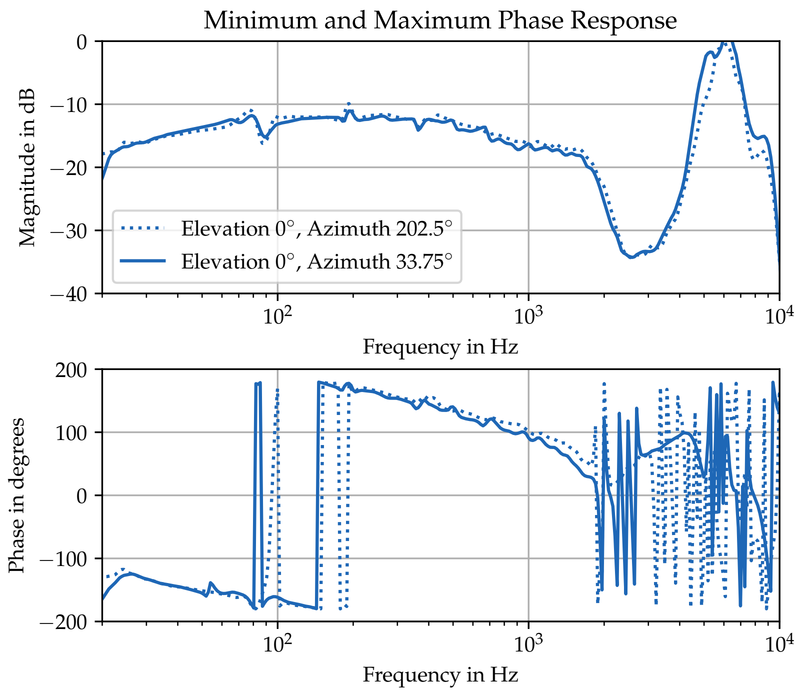

Figure 4.11.: The directional passive attenuation approximations with the highest and lowest phase response selected from 113 directions along a hemisphere over the sagittal plane.

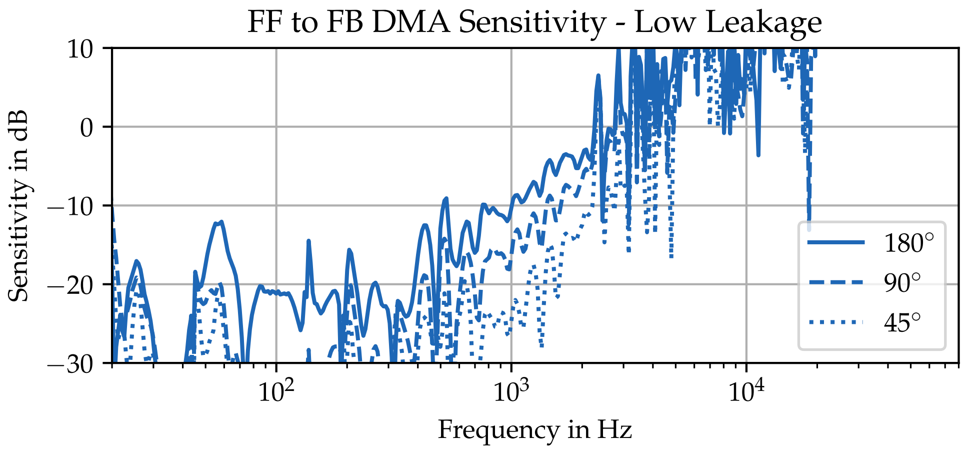

Figure 4.12.: Measured sensitivity of the differential microphone array consisting of the feed forward and the feedback microphone on the ams loose-fit earphone, measured at a low leakage wearing condition.

Figure 4.12 and 4.13 show the sensitivity of the simulated feed-forward feedback DMA using the ideal filter for high and low leakage, figure 4.14 shows the beam pattern at 1kHz for the three leakage conditions.

These results show that in general a DMA like directional response is possible for all leakage conditions. The filter matched to the passive attenuation approximation must be adjusted to follow the leakage conditions.

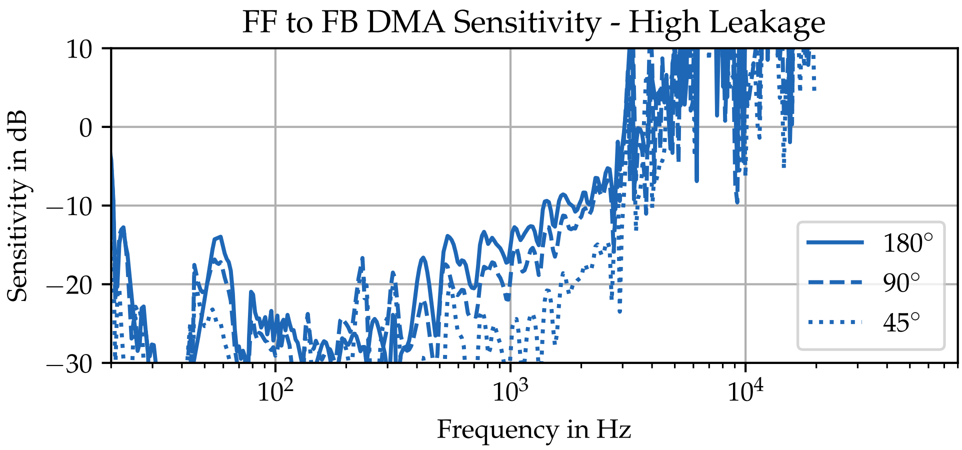

Figure 4.13.: Measured sensitivity of the differential microphone array consisting of the feed forward and the feedback microphone on the ams loose-fit earphone, measured at a high leakage wearing condition.

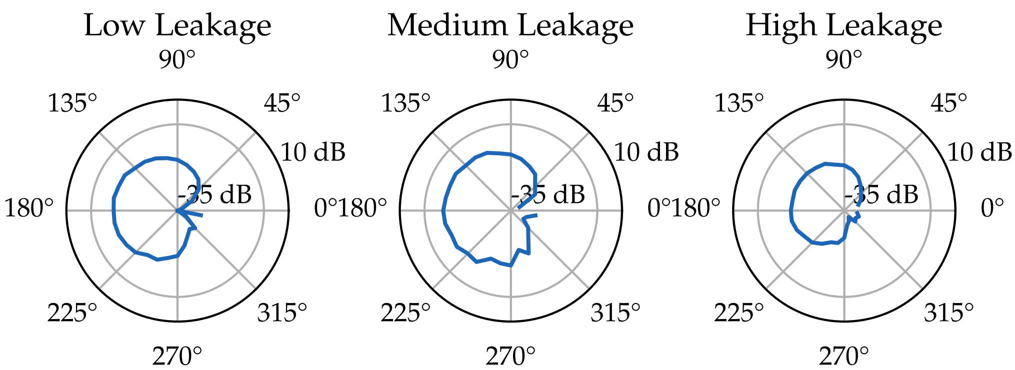

Figure 4.14.: Measured directivity pattern of the feed forward to feedback microphone DMA at a $1/9^{\text{th}}$ bandwidth around 1kHz, measured at three leakages. For each leak a matched $P(j\omega, \alpha_0)$ was used.

5. Algorithm Design

Introduction Based on the conclusions from the previous chapter regarding what the ideal DMA filter for the feed forward to feedback microphone array should look like for a given direction and leakage, the next step is to design an adaptive system that can approximate this ideal filter automatically.

Selected was an approach, which divides the system into parts corresponding to the variables to adapt to. The first part adapts to the leakage condition by using a LMS based cross-fading scheme and the second part takes care of the null steering by evaluating an averaged cross-correlation. A final and non-adaptive part compensates the DMA's high pass characteristic.

Design choices are mostly influenced by the target DSP hardware, namely the ams AS3460 noise-cancelling device.

Figure 5.1 shows the overall structure of the system.

Chapter Contents The overall structure of the adaptive system is described with detailed explanations on the three sections for adapting to leakage, adapting to the direction of incoming noise and compensating the DMA's high pass characteristic. In addition, the target hardware for the algorithm is introduced.

5.1. DSP Hardware

The algorithm was designed with respect to the ams AS3460 noise-cancelling chip. This chip contains two different DSP cores. The first, called „Aug-

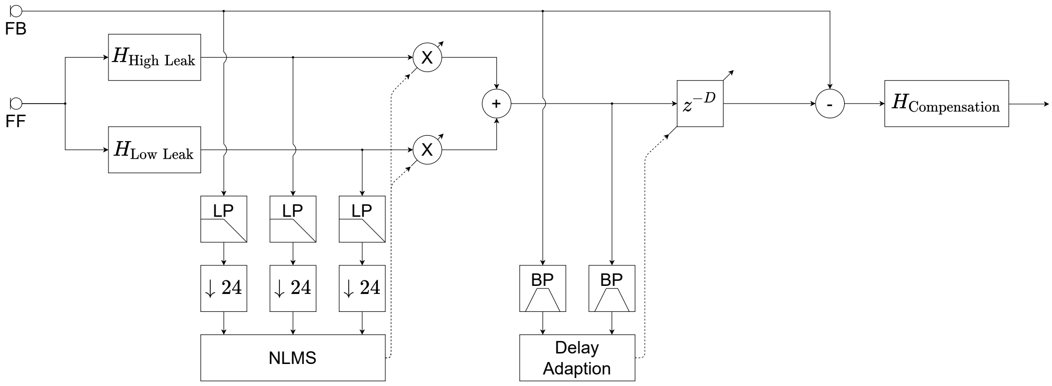

Figure 5.1.: Simplified block diagram of the adaptive system.

mented Hearing Engine"(AHE) is a custom DSP designed for energy efficient calculation of biquad filters at high sample rates (up to 348kHz) and the second is an ARM Cortex-M4 MCU used for running adaption logic at lower rates as well as system housekeeping tasks. Because of this structure, algorithms implemented on this system are split into filtering and adaption. The AHE provides the filtering and communicates down-sampled signals to the MCU where the adaption logic calculates and updates the AHE coefficients. The structure of the AS3460 can be seen in figure 5.2.

For the purpose of this thesis, the prototype implementation of the algorithm was written in python and is given in listing 3.

5.2. Leakage Compensation

The aim of the first section of the algorithm is to compensate the passive attenuation at the feedback microphone. Passive attenuation refers to the passive attenuation approximation as defined in eq. 4.5. Since this transfer function is relatively smooth, it can be approximated well by a series of five biquad filters. The biquad filters were implemented and adapted using the framework described in appendix A.

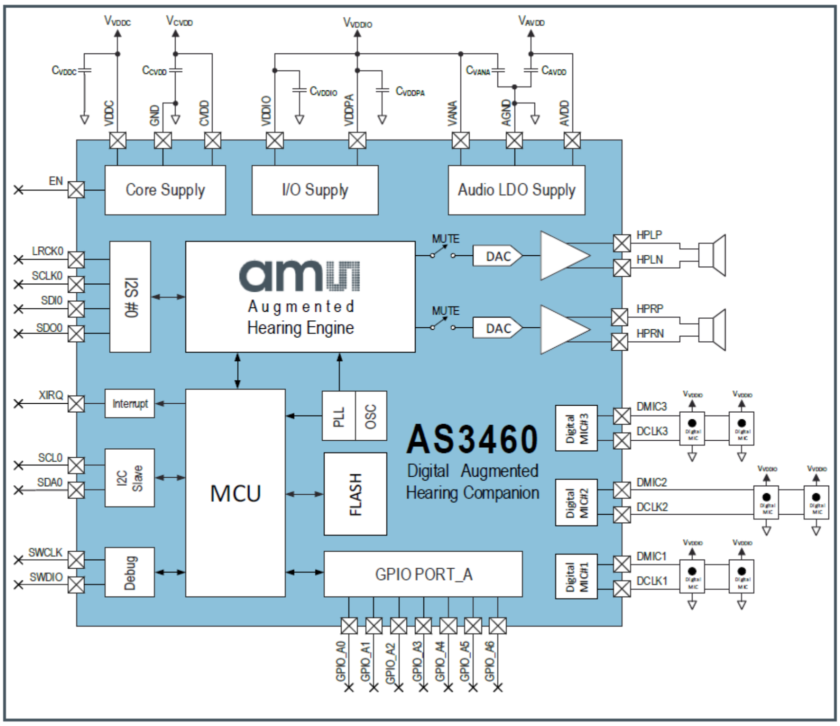

Figure 5.2.: Block diagram of the AS3460 noise-cancelling chip's internal structure.

Matching high and low leakage Figure 5.3 shows the transfer function of the measured passive attenuation approximation and a biquad filter array matched to it as well as the DMA's attenuation achieved with this match. The array consists of one gain and inverting[1] stage, five biquads and a simulated fractional delay. This means, that the transfer function $H_{\text{delay}}(j\omega) = e^{j\omega\tau}$ was implemented in frequency domain for the adaption and simulation. In the actual algorithm, a discrete delay will be used. Figure 5.4 shows the matching for a high leak case.

Matching in-between leakage Compensating leakage cases in between the two extremes is achieved by adaptively cross fading between the filter matched for low leak and the filter matched for high leak. Figure 5.5 illustrates how a mix of the filters shown in figure 5.3 and 5.4 match a medium leakage situation. In this result, the simulated fractional delay is already replaced by a discrete delay. This cross fading scheme is based on the technique used by ams for active noise-cancellation (see McCutcheon and Alcock, 2018).

Gain adaption The adaption of the two cross fading gains is implemented in a Least Mean Squares algorithm (LMS) like fashion. In a standard LMS, the learning rate depends on the scaling of the input signal, which is why for real systems, modified LMS algorithms are used, such as the Normalized Least Mean Squares algorithm (NLMS), which solves the problem by normalizing the step size with the power of the input signal (Haykin, 2002). The section of the input signal used to calculate the signal power is usually of the same length as the FIR filter to be adapted. For the application in this thesis this is not applicable, because the two gains are equivalent to two one-tap FIR filters and no meaningful normalizing can be done based on the signal power calculated with one sample. Therefore, the normalizing is implemented in a different way, namely by the IIR low pass-filtered absolute value of the input signal. This IIR filter runs on the AHE with the higher sampling rate, while the adaption itself runs on a lower sampling rate. The other signals, before going into the LMS calculation, are low pass filtered to

[1] Inverting the phase is necessary if one microphone has an inverted response relative to the other, as it is often the case when using one top and one bottom port MEMS-microphone.

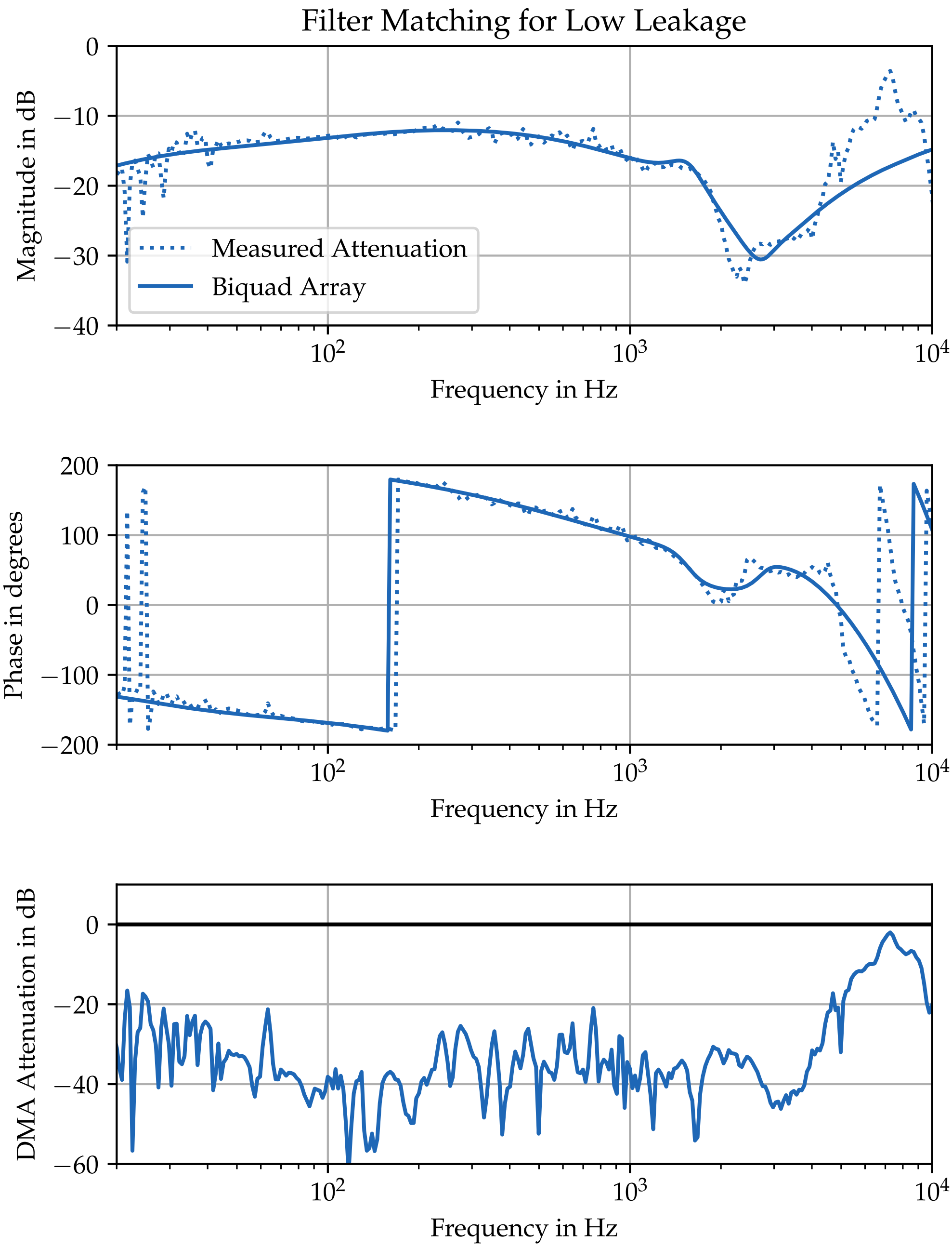

Figure 5.3.: Matching of the passive attenuation approximation by an array of biquad filters at low leak.

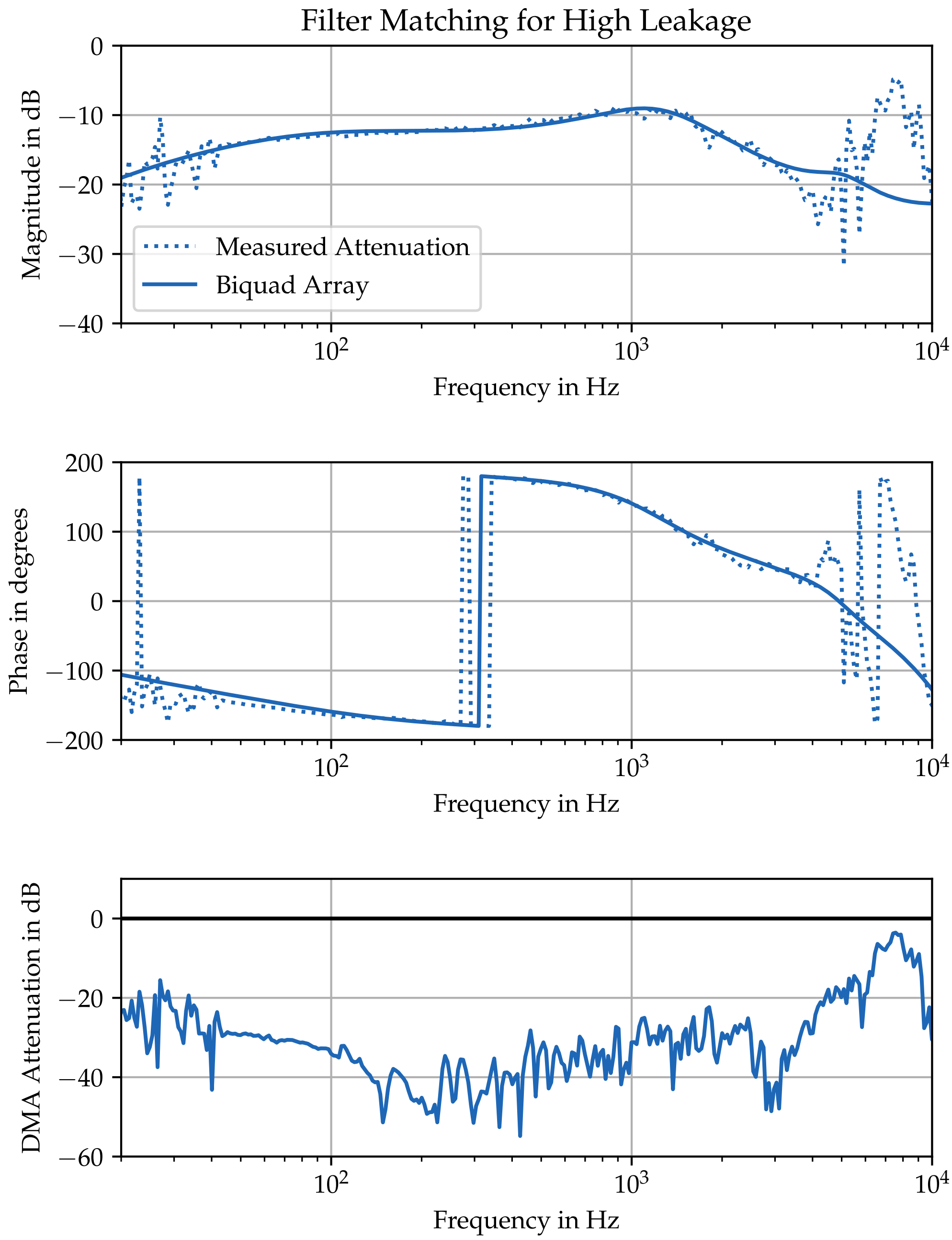

Figure 5.4.: Matching of the passive attenuation approximation by an array of biquad filters at high leak.

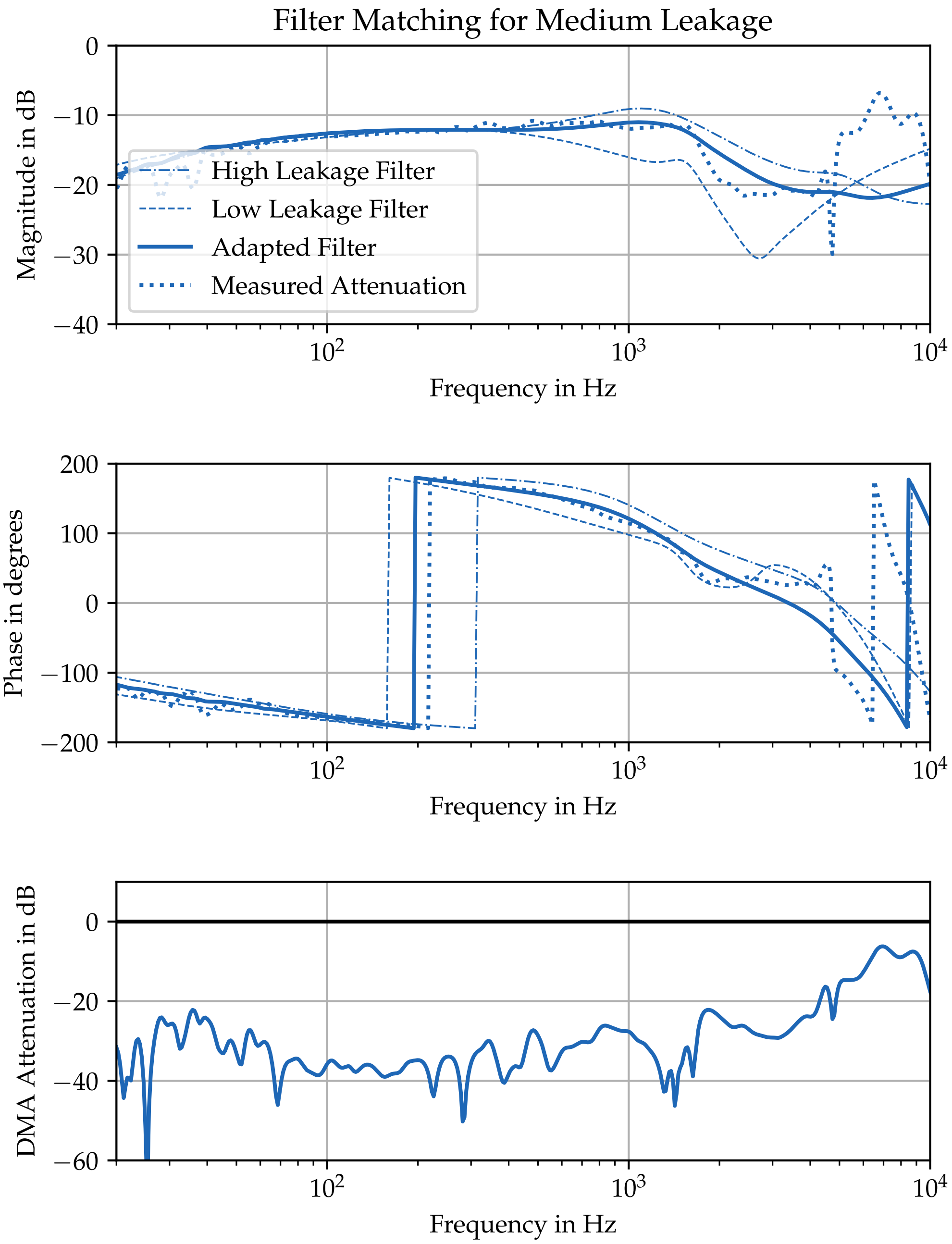

Figure 5.5.: Matching of a medium leakage passive attenuation approximation by mixing the filter matched for the high leakage case with one matched for the low leakage case. The gains are 0.36 for the low leakage and 0.65 for the high leakage filter. The filters are the same as in figure 5.3 and 5.4

avoid aliasing. The aliasing filter consists of two biquad low pass filters with a cutoff frequency of 500Hz. This provides an attenuation of approximately 47dB at the aliasing frequency, which is $\frac{f_s}{2}\frac{1}{24} = 2\text{kHz}$. The LMS already takes into account the adaptive delay.

Voice Activity The algorithm is meant to adapt only when the user is not speaking so it does not try to cancel the user's voice and is not confused by voice arriving at the feedback microphone via bone conduction. To pause the adaption in this case, the algorithm makes use of ams' voice activity detection, which already is implemented on the AS3460. It detects if the user is speaking by examining several features such as the energy ratio between feedback and feed forward microphone and tonality. The voice activity or enable adaption flag is provided manually in the python implementation. (see variable `vad` in listing 3)

5.3. Null Steering

At the second stage, after the outputs of the high- and low leakage filters were mixed together, is a discrete delay with an adaptive delay time. A discrete delay was chosen because the simulation results with ideal fractional delays in frequency domain showed no improvement over the discrete delay at a sampling frequency of 96kHz. Probably the leakage adaption, by mixing two filters with different phase responses, also works like a fractional delay to some extent.

The path from feedback microphone to the front vent is approx. 17mm long. At a speed of sound of $340\frac{\text{m}}{\text{s}}$ the time of flight is 50 µs or 4.8 samples at a sampling rate of $f_s = 96\text{kHz}$. At least at low leakage this path is static and does not change with the angle of incidence. The second path to consider, from feed forward microphone to front vent is 18mm long, with a time of flight of 53 µs or 5.1 samples. Therefore, the range to cover by the delay adaption is 4 to 10 samples. There are only six steering steps between the figure of eight pattern and endfire steering.

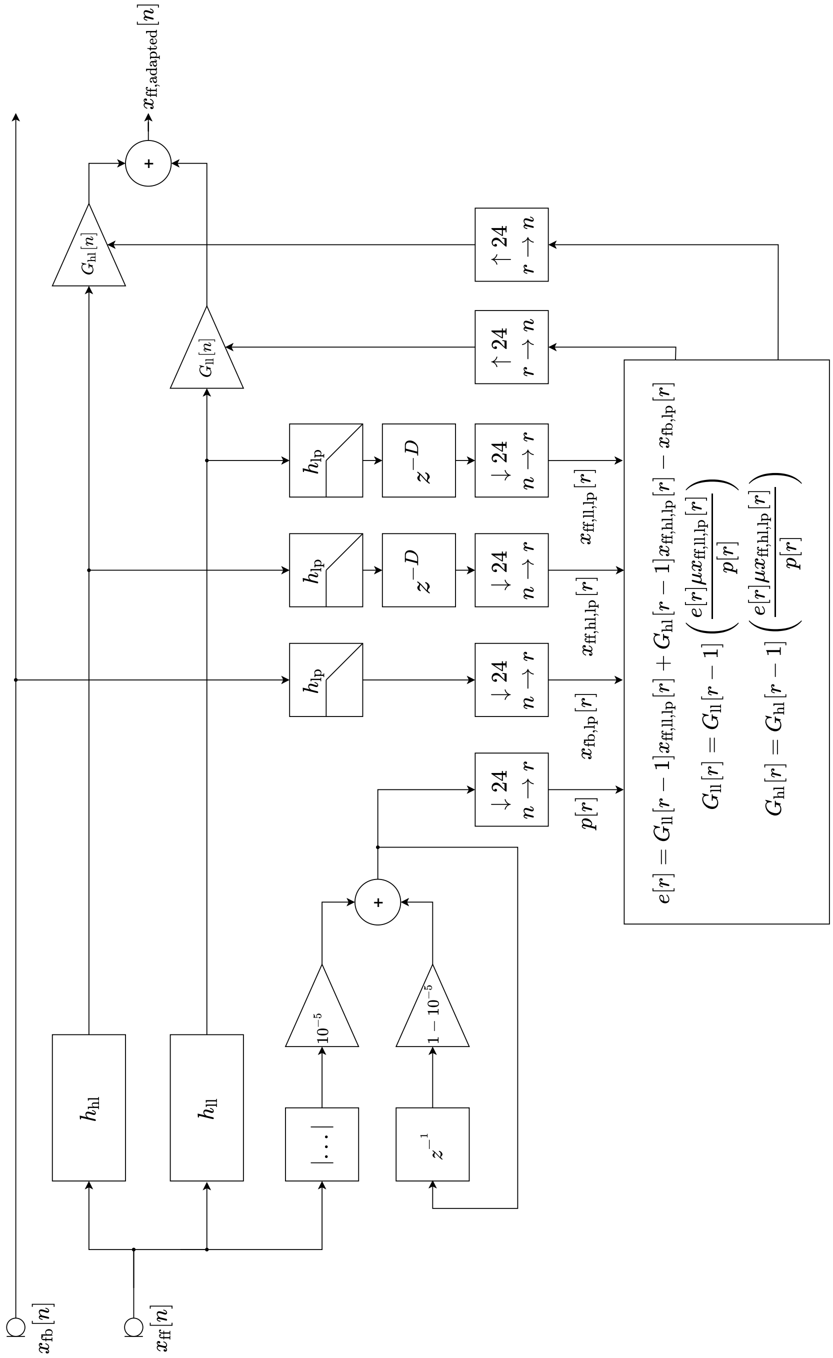

Figure 5.6.: First section of the algorithm for compensating the passive attenuation

Adaption of the delay time works by selecting the minimum of a slowly averaged cross-correlation. The output of the high- and low leakage mixing is band pass filtered and sent to a delay line. After the fixed delay of four samples the output after each of the six additional delay elements is subtracted from the also band pass filtered feedback signal. Then the power of each of the difference signals is tracked by the same kind of IIR structure already used in the first section. The optimal delay is selected by choosing the power-tracking IIR with the smallest output. The band pass filter's purpose is to focus the adaption on the frequency range where the delay mismatch has the biggest impact. A flowchart of this algorithm is in figure 5.7. As with the gain adaption, the delay adaption is also paused when the user is speaking.

5.4. High Pass Compensation

Figure 2.8 shows the high pass characteristic of an ideal DMA, which should be compensated at the DMA's output. In the case of the feedback to feed forward microphone DMA, the sensitivity or frequency response is more complicated, because of the varying mismatch between target and filter. For this reason, the compensation filter was based on measurements and simulations rather than the theory.

With the earphone sitting at low leakage in an artificial head's ear, the $\frac{A2FF}{A2FB}$ transfer function from the direction of the artificial head's mouth was measured. With this transfer function and the response of the low leakage compensation filter, the DMA's attenuation of a signal coming from the mouth direction could be simulated. The inverse of this attenuation is the target for equalizing or compensating the DMA output. The target was filtered with a high pass to limit sensor noise amplification. Figure 5.8 shows the match of the low leakage filter with the transfer function in mouth direction as well as the resulting signal attenuation at the DMA output. Figure 5.9 shows the filter compensating this attenuation.

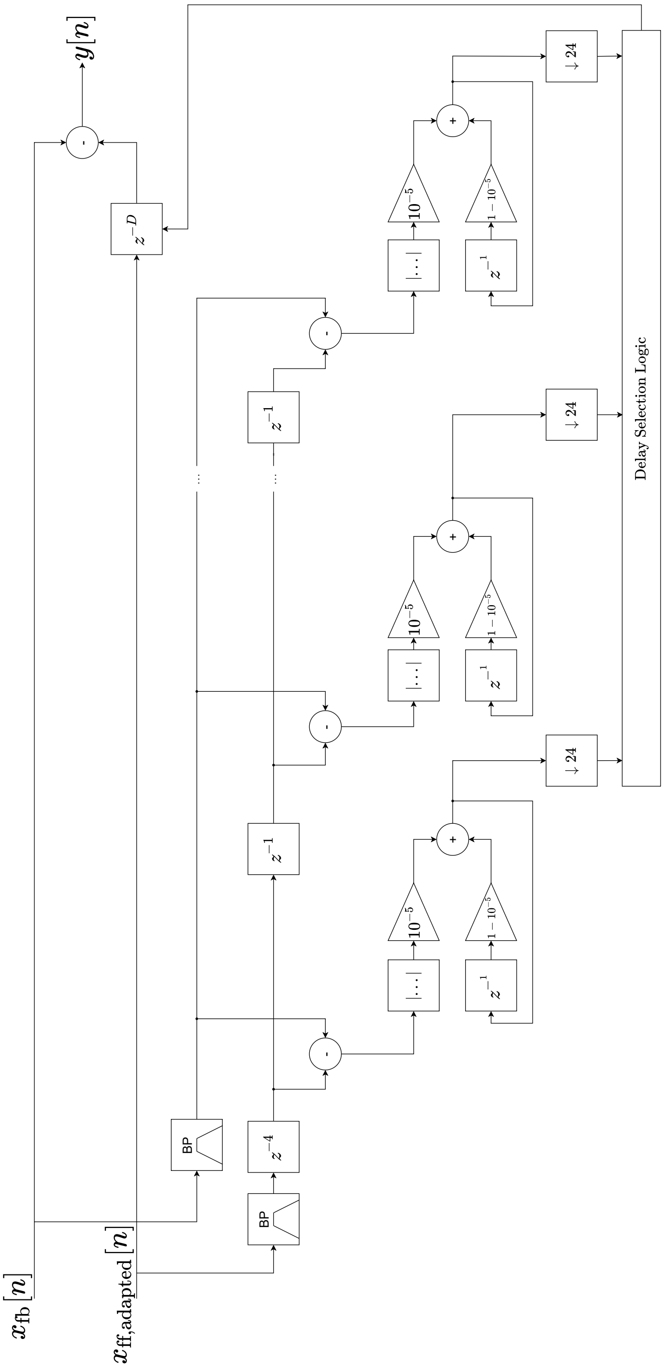

Figure 5.7.: Second section of the algorithm for steering the null or delay adaption

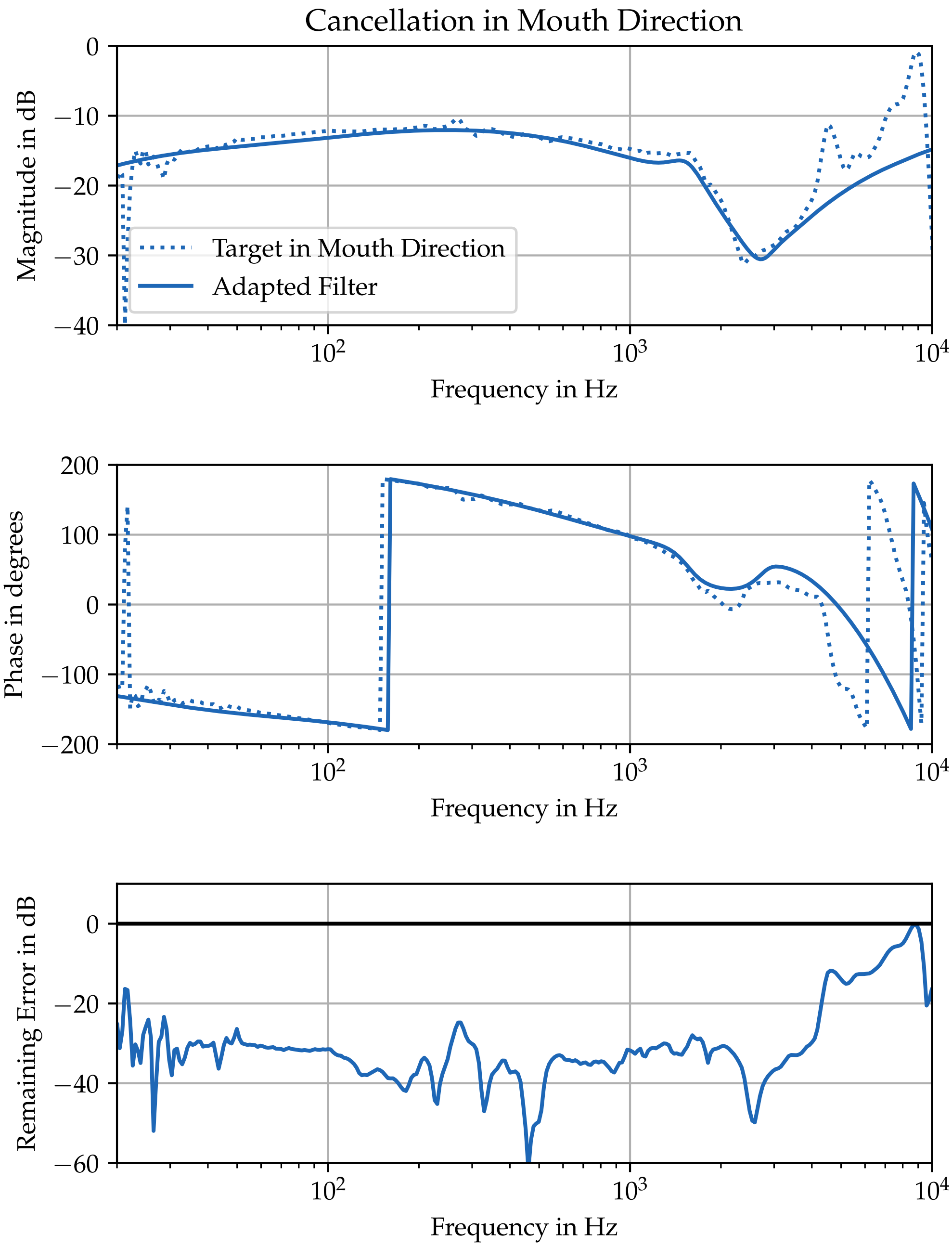

Figure 5.8.: Filter matching and signal attenuation of the DMA in the direction of the user's mouth.

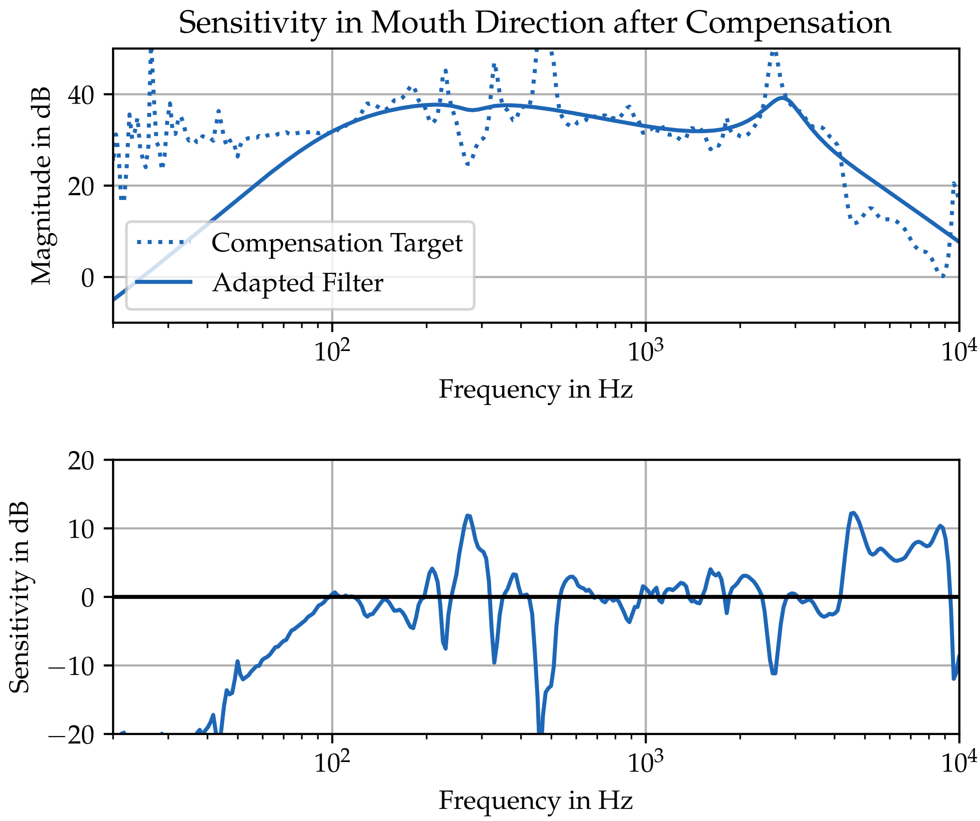

Figure 5.9.: Filter to compensate signal attenuation in mouth direction and resulting sensitivity.

6. Performance Evaluation

Introduction The final step in the development of the algorithm is to evaluate its function in a range of possible use cases and attain a reliable assessment of its performance. To do so, a measurement setup using an artificial head and a ring of loudspeakers was used to examine the signal improvement with noise from different directions as well as the performance of the algorithm in scenarios with omnidirectional noise.

It is evaluated by showing some selected frequency responses to illustrate the algorithm's workings, followed by a thorough evaluation of all test cases based on two metrics, the signal to noise ratio (SNR) and the psychoacoustically modeled audio quality measure PSM_t. These results are presented by showing the spatial distribution of the signal improvement as well as interpreting it statistically.

Chapter Contents This chapter describes the full algorithm evaluation, including the measurement setup, selected test cases and a spatial and statistical evaluation of all test cases.

6.1. Measurement Setup

The evaluation measurement was conducted in the ams audio laboratory's main room, which has a floor area of approximately $40m^2$. Although it is mainly used and equipped for working with electronics, it underwent some acoustical treatment and has a carpet floor and absorbers on the wall. Hanging from the ceiling is a circular arrangement of seven monitor speakers, two of which were used at their original position as the top speakers in the evaluation measurement, the other four were relocated on monitor stands. Figure 6.1 shows the speaker arrangement including the head simulator. The head simulator was rotated in four 45° steps to get to a total of 24 directions for the noise sources. This way the noise sources can cover roughly a spherical surface although only a vertical ring of speakers was available. The noise source positions with this rotation are illustrated in figure 6.3.

A reference measurement was conducted to check if different speaker responses might affect the results. A calibrated measurement microphone was placed 4cm above the head and the spectrum of each speaker playing back pink noise was recorded. 12 times 5s were recorded to obtain a reliable average. The measurement shows that the frequency response of the test speakers in the bandwidth of interest deviates by ±2.5dB (see figure 6.2). Since all of the presented results refer to relative values, i.e. the difference in signal quality or signal improvement, this small deviation of the absolute should not diminish the validity of the results.

Table 6.1.: Equipment list for evaluation measurements

Equipment	Model
Test speakers	6x Neumann K120A
Signal acquisition device	AS3460 and MiniDSP MCH Streamer
Head simulator	HEAD acoustics HMSII
Artificial ear	G.R.A.S. KB5011
Ear canal simulator	G.R.A.S. RA0045
Mouth speaker	B&O A1
Reference microphone	Crysound CRY372

6.2. Measurement Procedure

The measurement begins by inserting the earphone in one ear[1] and performing and evaluating a characterization measurement and thereby checking what leakage condition the earphone is in. This step is repeated after ten

[1]in all following measurements it was the left ear

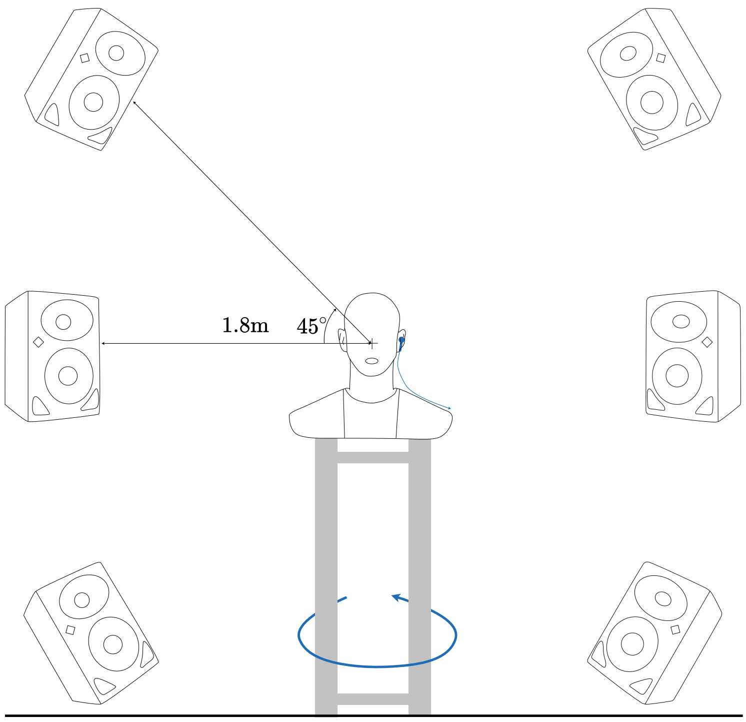

Figure 6.1.: Sketch of the evaluation measurement setup. The distance from the center of the head to the speakers was 1.8m for all directions.

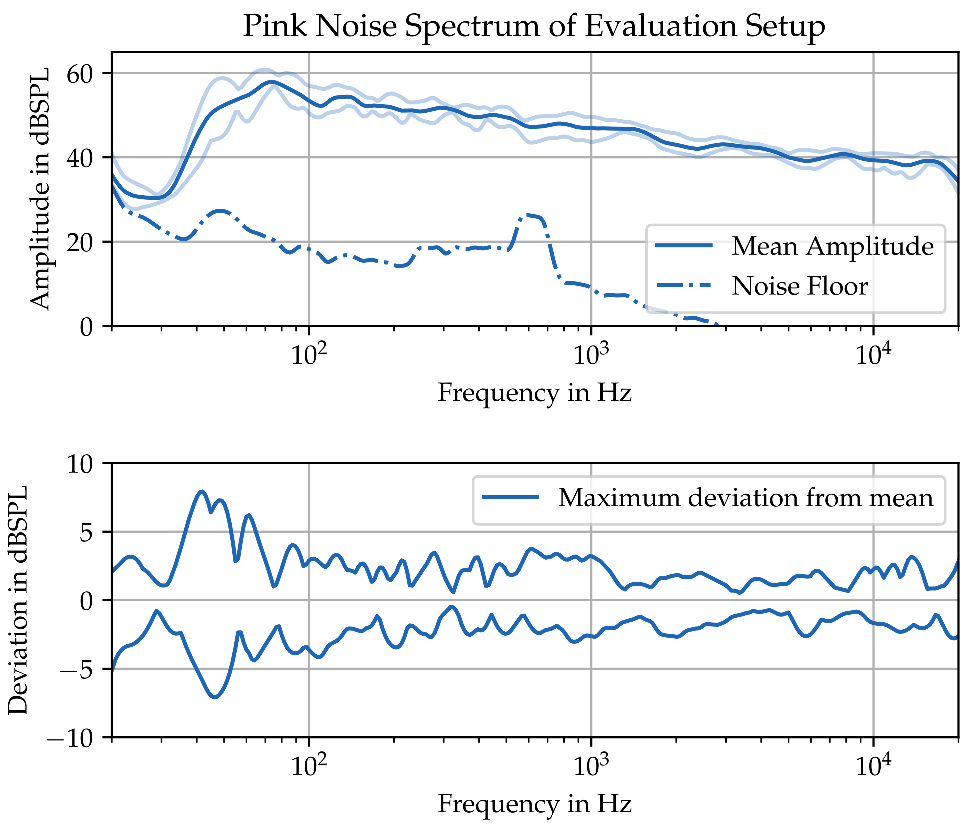

Figure 6.2.: Top: Noise floor measured in the room and the mean frequency response across the six channels. Indicated in light blue is the minimum and maximum. Bottom: the maximum deviation from the mean speaker response.

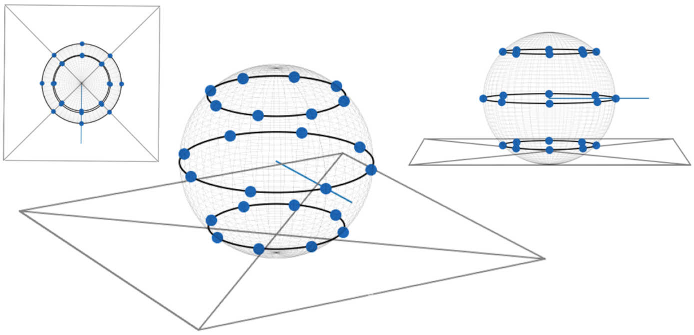

Figure 6.3.: Three views on the used noise source positions covering roughly a spherical surface around the artificial head. Indicated by the blue line is the direction normal to the face.

minutes, to check that the earphone did not move. Even a small movement can already constitute a significant leakage change and this cannot be checked visually.

Then a small speaker is positioned in front of the artificial head's mouth to playback the speech samples. The artificial head does have an internal mouth simulator speaker, but measurements with this were heavily distorted by structure-borne sound transmission inside the head. Distorted here means that the results with structure borne speech transmission were better than without. For the real human head, a similar effect is expected due to bone conduction, but it was intended to study the system without this effect in this measurement.

The samples are recorded through the earphone's microphones via the AS3460 and an I2S to USB audio interface. Followed by the speech recordings are the noise recordings. Approximately $30 - 60$s are recorded for each noise sample and for each channel. After recording the signals from six speakers, the artificial head is rotated and the process repeated. The actual test signals were produced afterwards by appending and mixing the noise and speech recordings. All result signals were generated using the algorithm in its development state recorded in listing 3 with the parameters of listing 4.

For the complete evaluation, this process was performed for a low, a medium and a high leakage condition, using a female and a male voice sample. Two

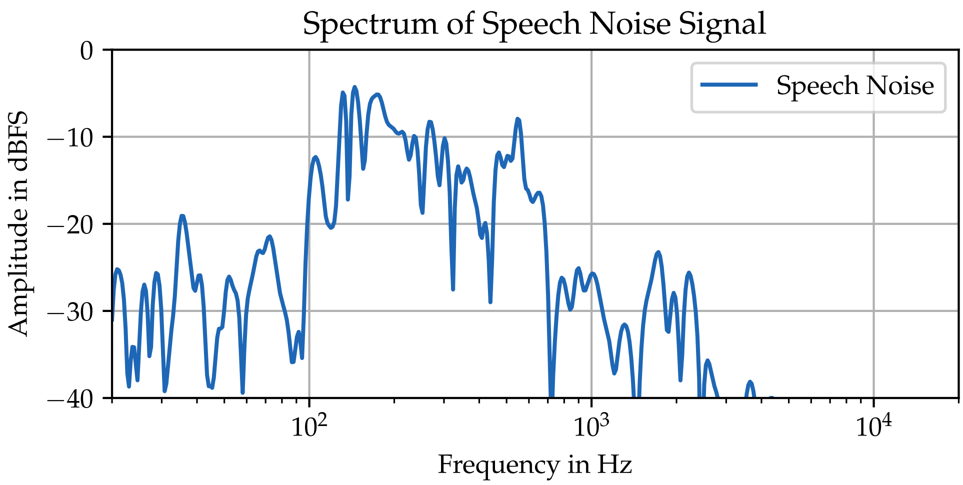

Figure 6.4.: 1/9th octave smoothed spectrum of the speech noise signal.

samples were used for the interfering noise: pink noise and „speech noise". „Speech noise" is a superposition of ten parliament speeches, constituting a noise like signal with a speech like spectrum (see figure 6.4).

6.3. Case Studies

Before evaluating the algorithm's performance based on statistics generated with above described measurement, the data will be used to illustrate the workings of the algorithm by showing a few selected examples.

Example 1: Normal use case

Noise	Speech	Leakage	Azimuth	Elevation
Pink	Female2	Low	180°	−45°

Pink noise from a source position low behind the head, i.e. an azimuth of 180° and an elevation of −45° relative to the head (see figure 6.5) and using a 20s speech sample of a female voice. The signal to noise ratio in this configuration is 7.1dB at the feed forward microphone and improves by 6.1dB at the output of the DMA. The algorithm was given 5 seconds to

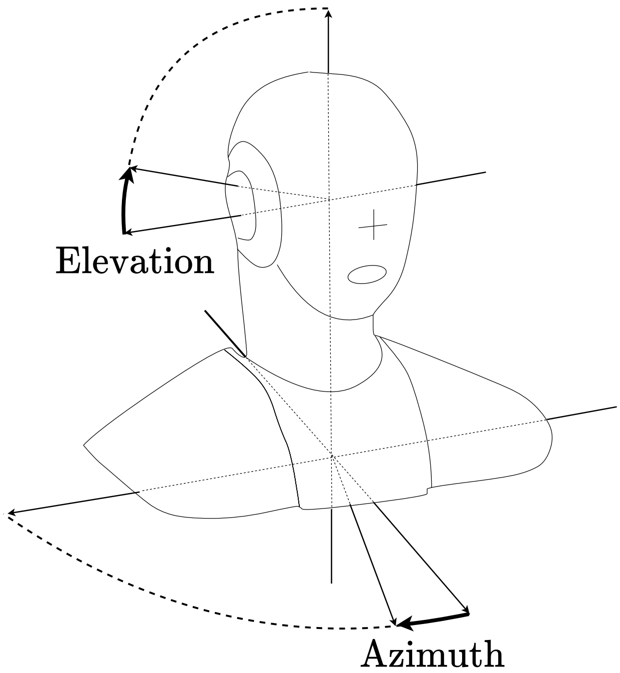

Figure 6.5.: Directions in the spherical coordinate system used for the measurement setup.

fully adapt, although the improvement after the first second of adaption is small.

There are two effects responsible for the SNR improvement. Firstly, the spatial separation in which the interfering noise is more attenuated by the system than the speech signal and secondly, the system's overall filtering characteristic (mostly a high pass characteristic), which improves the SNR by attenuating signal components outside of the speech bandwidth more.

Since the latter could also be achieved by a single microphone and a high pass filter, the two effects are shown separately in the following plots. To do so, the filtering that the noise is subject to is compared with the filtering that the speech signal is subject to. This is possible, because the algorithm can be paused using the voice activity flag. So one can let the system adapt, pause the adaption by enabling the voice activity flag and then send the noise and speech signal separately through the system. The filtering is calculated as the following transfer functions:

$$H_{\text{noise}}(j\omega) = \frac{Y_{\text{noise}}(j\omega)}{FF_{\text{noise}}(j\omega)} \tag{6.1}$$

$$H_{\text{speech}}(j\omega) = \frac{Y_{\text{speech}}(j\omega)}{FF_{\text{speech}}(j\omega)} \tag{6.2}$$

Where $Y_{\text{noise}}(j\omega)$ is the Fourier transform of the noise signal at the DMA output and $FF_{\text{noise}}(j\omega)$ is the noise signal at the feed forward microphone and $Y_{\text{speech}}(j\omega)$ is the Fourier transform of the speech signal at the DMA output and $FF_{\text{speech}}(j\omega)$ is the speech signal at the feed forward microphone. Figure 6.6 shows these transfer functions and their difference.

One can see that there is a strong spatial separation with up to 20dB difference, but unlike the theoretical DMA, the spatial separation is not as simple as in figure 2.8. The variable mismatch between the adapted filter and the actual attenuation leads to a more complicated directional sensitivity.

Example 2: No spatial separation

Noise	Speech	Leakage	Azimuth	Elevation
Pink	Female2	Low	0°	0°

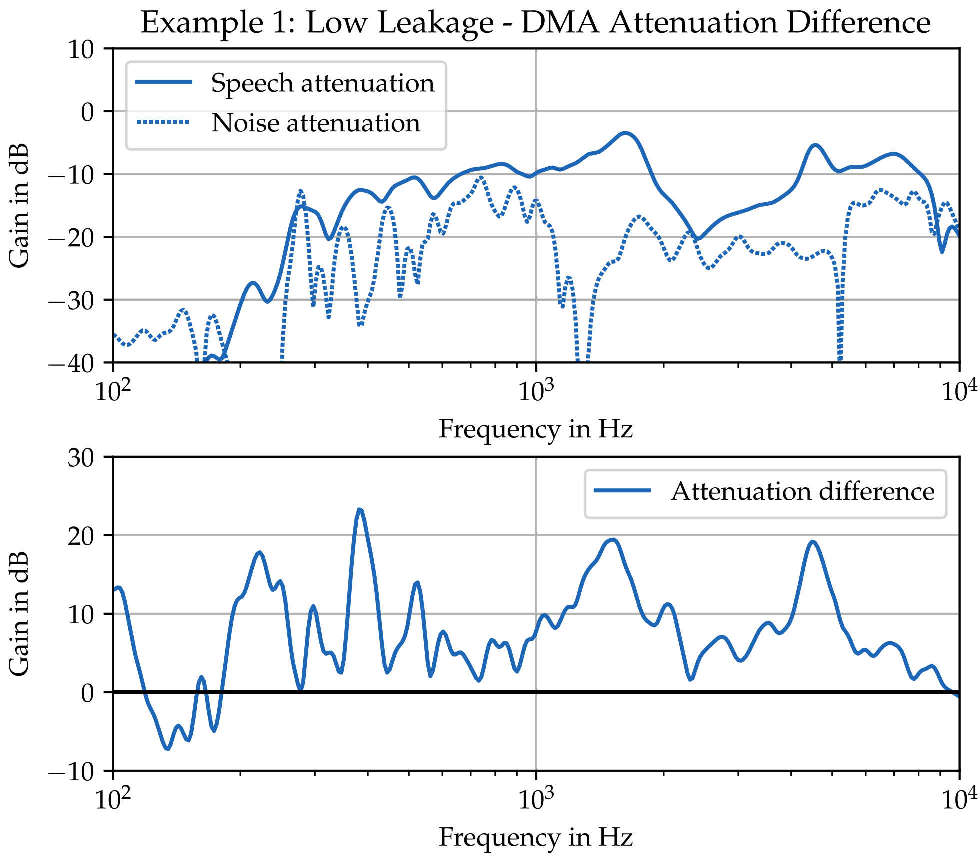

Figure 6.6.: Attenuation difference at low leakage. Top: Attenuation of noise and speech signal by the DMA as described in example 1. Bottom: the difference between the attenuations. A 1/12$^{\text{th}}$ octave smoothing was used.

The same speech sample as in example 1 and pink noise from a source position in front of the head, i.e. an azimuth of 0° and an elevation of 0°. This means that at the DMA the speech and noise signal arrive almost from the same direction. Here the SNR improvement from feed forward microphone to the DMA output is 0.3dB. The difference in attenuation can be seen in figure 6.7. It is much smaller than in example 1 and almost no spatial separation can be achieved.

Example 3: High leakage

Noise	Speech	Leakage	Azimuth	Elevation
Pink	Female2	High	0°	−45°

At high leakage, the SNR improves from 13.7dB by 5.2dB. The performance at high leakage is similar to the low leakage performance (see figure 6.8). Overall, the performance is similar to the low leak case, although here a different mismatch between filter and real attenuation produces a different spectrum of the attenuation difference.

Example 4: Medium leakage

Noise	Speech	Leakage	Azimuth	Elevation
Pink	Female2	Medium	0°	−45°

At medium leakage, the SNR improves from 8.4dB by 3.7dB. For a medium leakage condition, the performance is worse than for the high and low leak conditions. One explanation for this cold be the multi-path propagation occurring at medium leakage. As it was studied in the previous chapters, in the low leak condition the noise travels mostly through the front vent and in the high leak condition travels mostly through the leakage around the earphone. In the in-between condition, both paths are equally present. With this two-path propagation, the spatial separation is less reliable.

Example 5: Omnidirectional noise

Noise	Speech	Leakage	Azimuth	Elevation
Pink	Female2	Low	all positions	

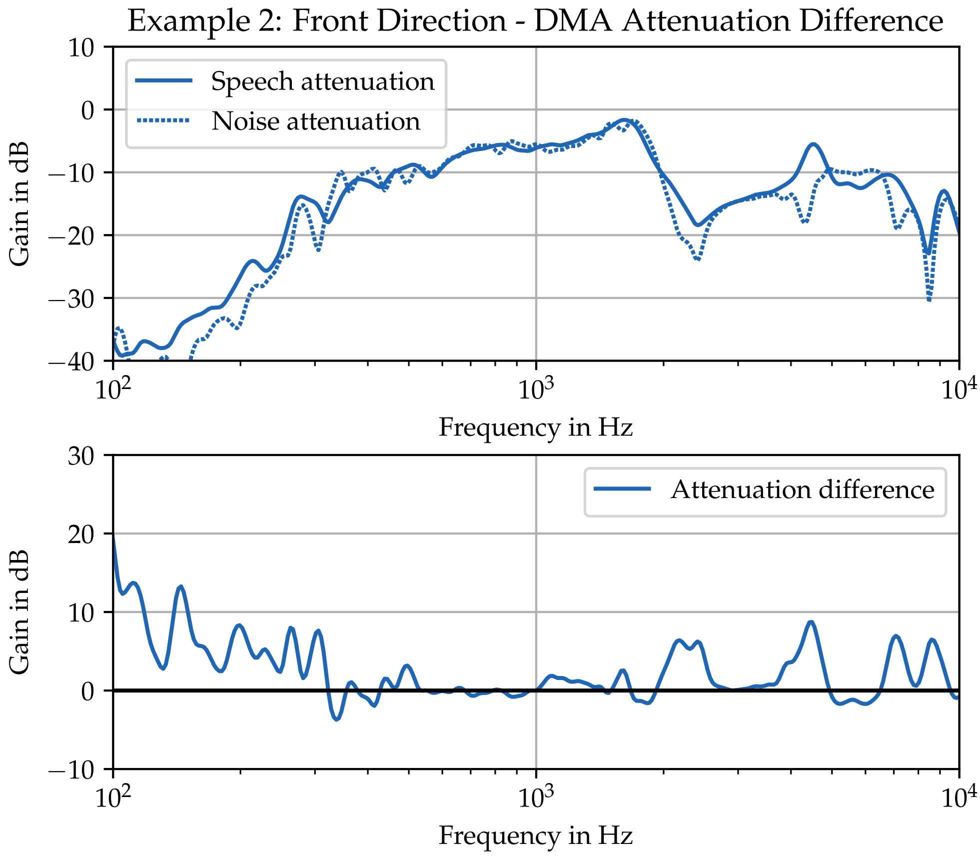

Figure 6.7.: Attenuation difference if noise and speech signal arrive from the same direction. Top: Attenuation of noise and speech signal by the DMA as described in example 2. Bottom: the difference between the attenuations. A 1/12$^{\text{th}}$ octave smoothing was used.

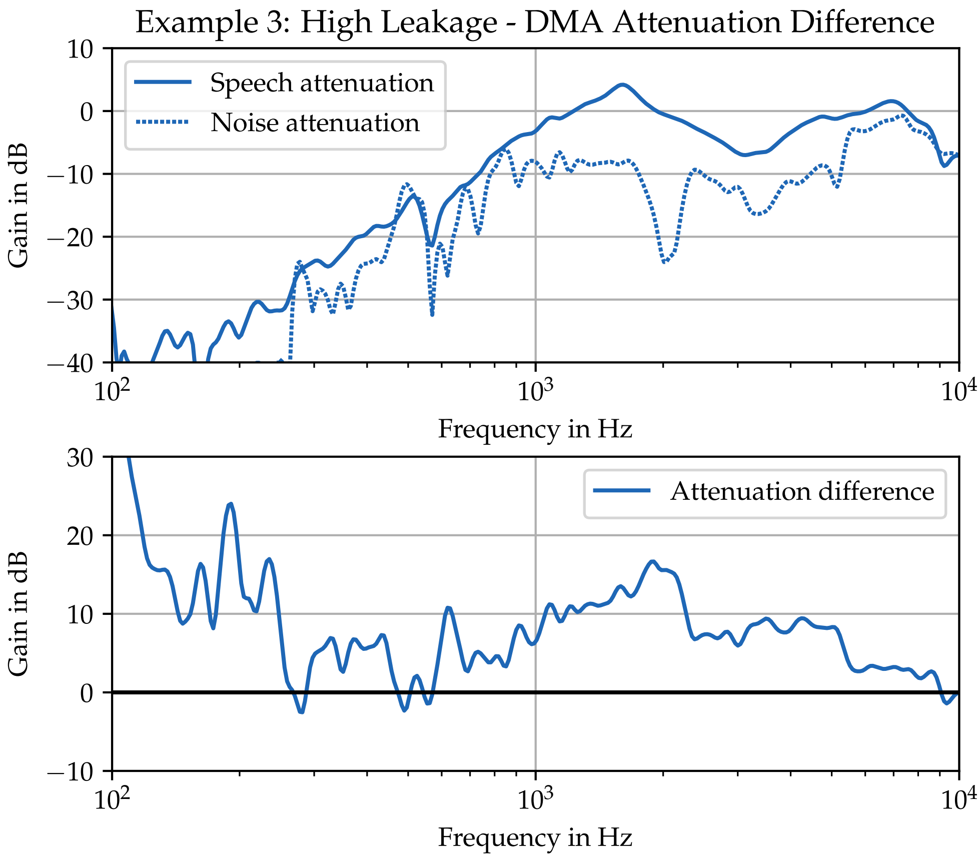

Figure 6.8.: Attenuation difference at high leakage. Top: Attenuation of noise and speech signal by the DMA as described in example 3. Bottom: the difference between the attenuations. A 1/12th octave smoothing was used.

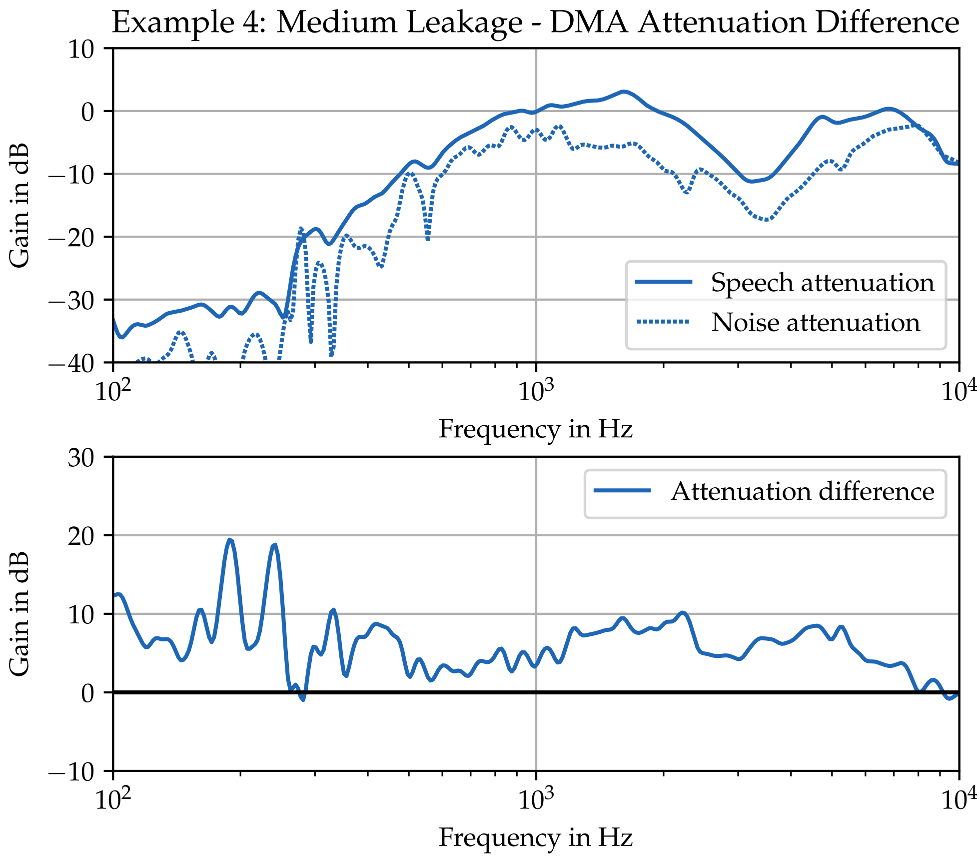

Figure 6.9.: Attenuation difference at medium leakage. Top: Attenuation of noise and speech signal by the DMA as described in example 4. Bottom: the difference between the attenuations. A 1/12$^{\text{th}}$ octave smoothing was used.

Due to the nature of the measurement setup in which not all speaker positions can be used at the same time, the omnidirectional noise had to be produced from the separate recordings, by superposition of the single channel recordings. Before summing the single channels, their gain was equalized using the measurements from figure 6.2 to avoid bias in any direction and they were time shifted using a random delay between 0.5s and 1.5s to avoid correlation between the identical pink noise samples. This resulted in an SNR of -7.5dB which the algorithm improved only by 0.5dB. Figure 6.10 shows the filter separation for this example. Here also no good spatial separation can be achieved.

6.4. Performance with Directional Noise

An overview of the algorithm's performance across all directions is given in two ways. There are polar plots, which visualize the spatial distribution of the performance and there are so called violin plots, which try to capture the performance by showing a histogram of the values. Two metrics are used for this evaluation, namely the SNR value, already used in the previous section and the PSM_t value, which is an audio quality measure based on a psychoacoustic model, introduced as a part of PEMO-Q by Huber and Kollmeier, 2006.

PSM_t The PSM_t value is calculated by comparing an undistorted sample with the distorted recording of it. Therefore, there was one value calculated to capture the signal quality loss from the clean to the noisy speech signal at the feed forward microphone and one for the loss from the clean speech signal at the feed forward microphone to the output of the DMA with speech and noise. The difference between the two represents a psychoacoustical metric for the signal improvement achieved by the DMA. The output values of the PSM_t calculation range from 0 to 1, with 1 being no quality loss at all.

Figure 6.11 shows the SNR improvement provided by the algorithm for each test speaker or noise source position. For this example pink noise and a female voice speech sample was used with the earphone being in a

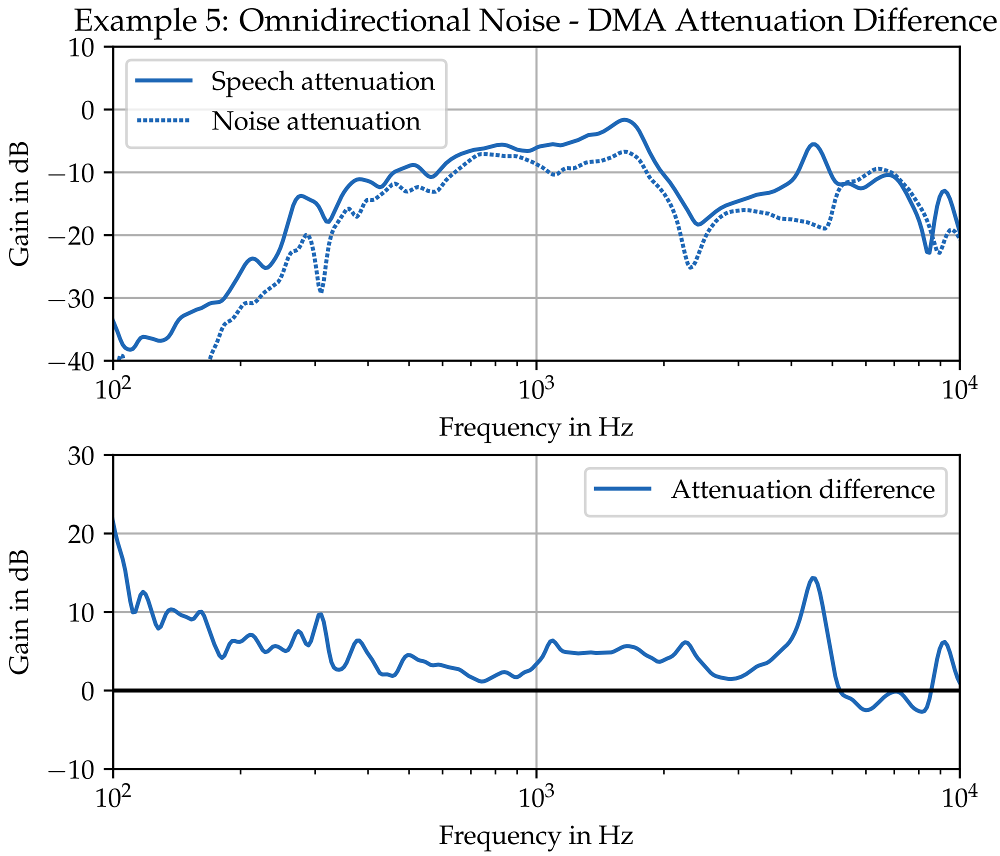

Figure 6.10.: Attenuation difference at low leakage with noise arriving from all available speaker positions. Top: Attenuation of noise and speech signal by the DMA as described in example 5. Bottom: the difference between the attenuations. A $1/12^{\text{th}}$ octave smoothing was used.

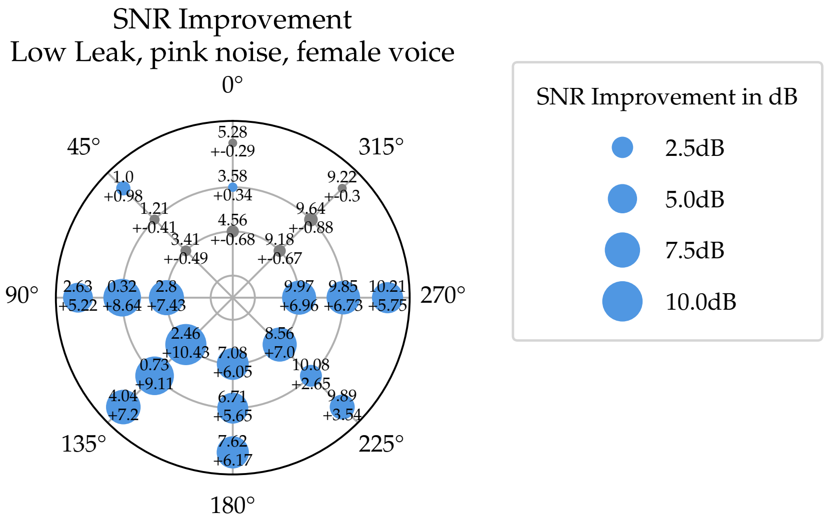

Figure 6.11.: Algorithm performance for noise directions. Here, showing the SNR value for the test case with pink noise, a female voice sample and a low leakage condition.

low leakage condition. The height of the test speaker or elevation angle is projected onto the radius. This means that the inner circle represents the lower speakers in the test setup and the outer circle those just below the ceiling. Azimuth is displayed as in the coordinate system from figure 6.5. Each dot is annotated with the SNR or PSM_t value before applying the algorithm and „+" the improvement. Grey dots indicate a negative value, where the algorithm actually decreased the signal quality. Figure 6.12 shows the PSM_t improvement for the same test case. All other test cases are listed in table D.1 in appendix D.

Statistical Results Some of the histograms in the violin plots (figure 6.13 and 6.14) show a bi modal distribution. This is because there is a set of directions for which the algorithm cannot achieve an improvement and there are directions for which the spatial separation works well and the disturbance can be partially canceled from the signal, as illustrated by the case studies above. The overall results, expressed by the median results for the twelve combinations of voice, noise and leakage conditions show that

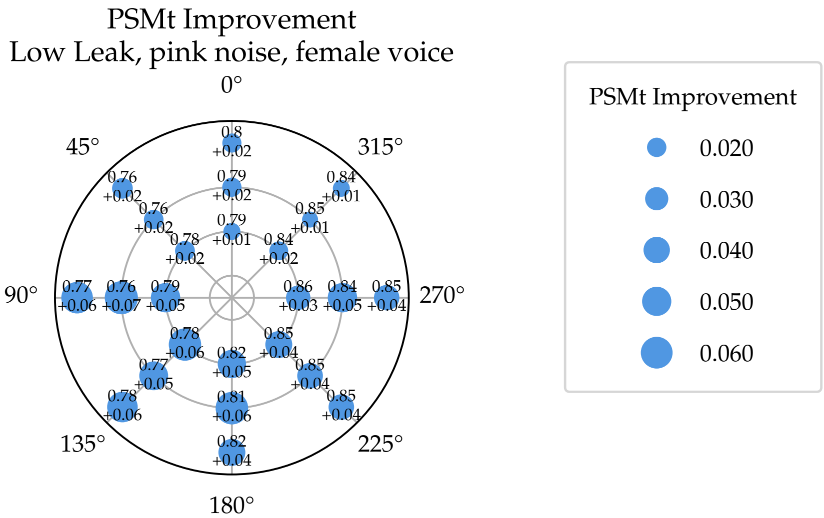

Figure 6.12.: Algorithm performance for noise directions. Here, showing the PSMt value for the test case with pink noise, a female voice sample and a low leakage condition.

the algorithm can always achieve a SNR improvement, although it does not perform equally well in all cases. Noticeable is the diminished performance for the medium leakage conditions, which is explained in the fourth case study above.

Comparison with FF to FF2 Array In the context of this statistical overview, a reference shall also be provided about how the performance is, compared to a regular microphone array. (see section 4.2) To calculate these results, the algorithm was provided with the FF2 microphone signal instead of the feedback microphone signal and stage one of the algorithm was removed, since there is no leakage to compensate in this configuration. Figure 6.15 shows the results. The regular microphone array performs more consistent, since it is not affected by the performance loss at medium leakage and performs better in the pink noise environment, but slightly worse in the speech noise environment. This indicates that the feed forward to feedback microphone array might achieve a better phase matching in the higher frequency regions. This might be due to the previously described effect, in

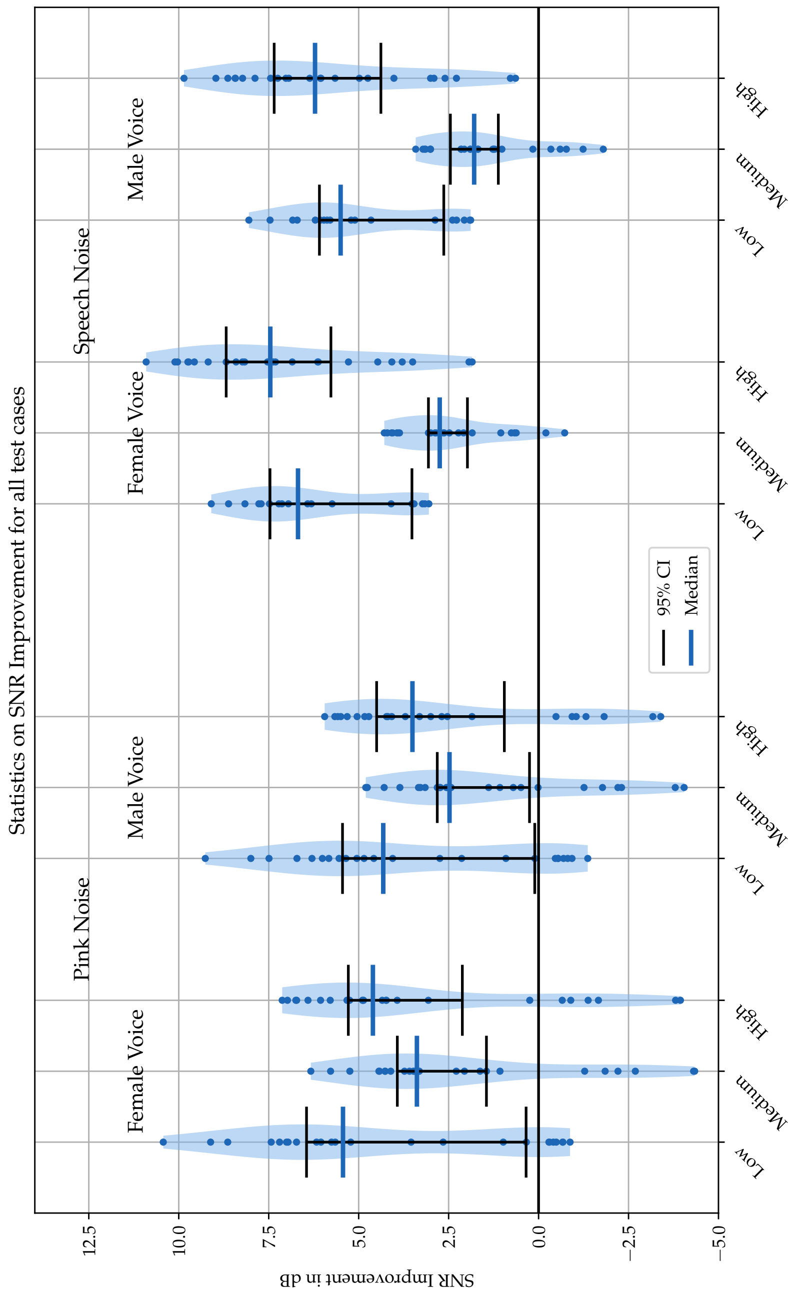

Figure 6.13.: Statistical evaluation of the SNR improvement for all test cases.

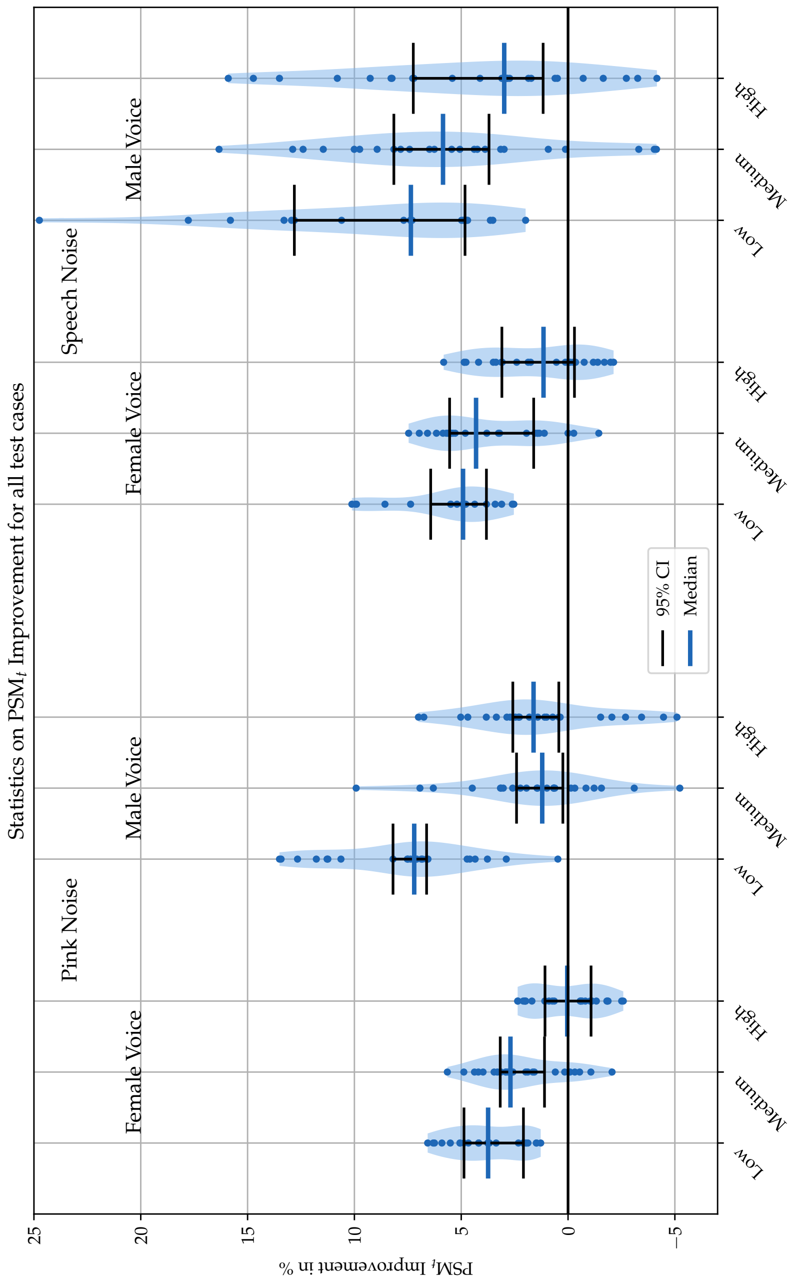

Figure 6.14.: Statistical evaluation of the PSM$_t$ improvement for all test cases.

which the leakage compensation algorithm can work like a fractional delay to some extent. This feature was not available for the outside microphone array, since the first stage of the algorithm was deactivated.

6.5. Performance with omnidirectional Noise

To estimate the algorithm's performance for scenarios in which noise does not arrive from a particular direction, but instead arrives from all directions, the procedure described in Example 5 in section 6.3 was applied to the same set of speech and noise samples and leakage conditions as for the evaluation with directional noise sources. The results are listed in table 6.2

In these omnidirectional noise scenarios, without a bias towards any direction, the null-steering part of the algorithm will not converge. Therefore, once the voice activity flag is enabled, the the alignment of the beam pattern will be fixed to a random direction. The random direction could point the null in mouth direction and worsen the SNR or it could point away and improve the SNR by suppressing at least some portion of the noise. A meaningful extension of the algorithm for this scenario could be to detect when the null steering does not converge and switch to a fixed steering.

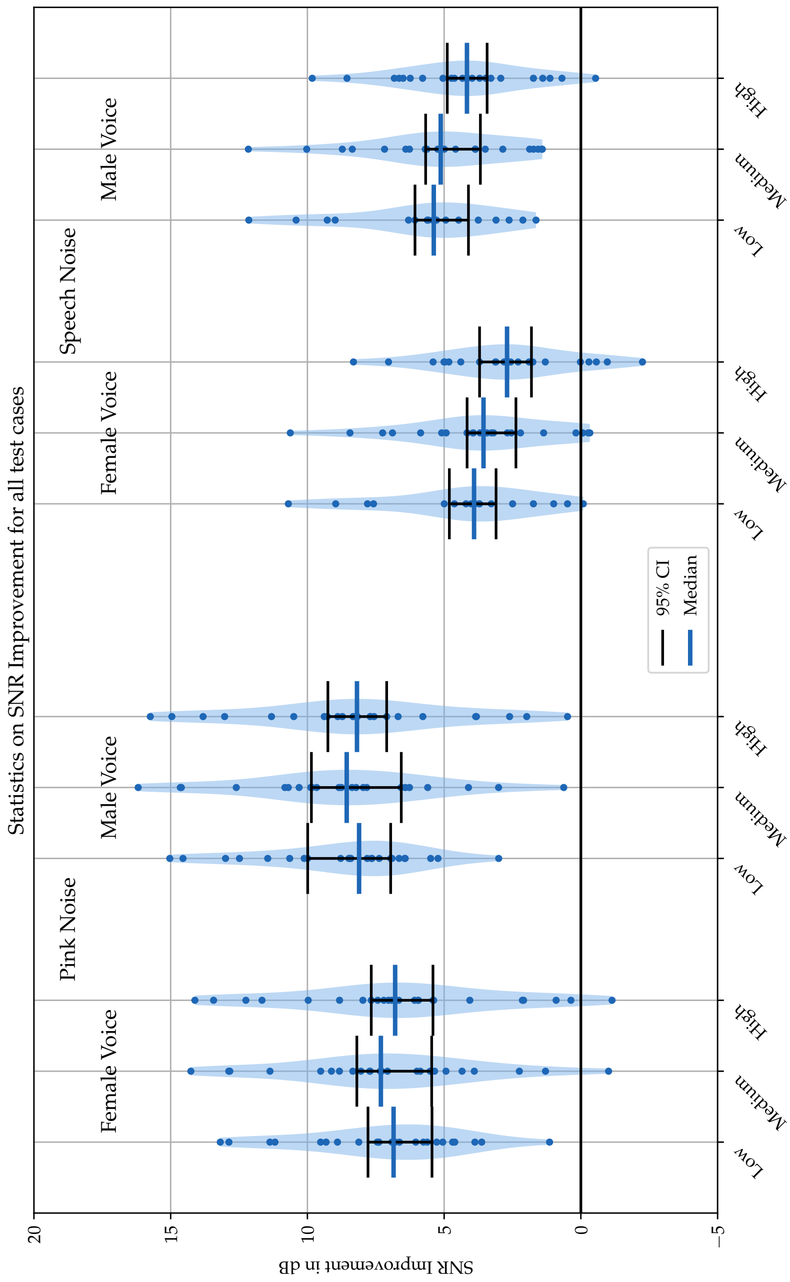

Figure 6.15.: Statistical evaluation of the SNR improvement for all test cases, when using the microphone array on the earphone's outside.

Table 6.2.: Algorithm Performance Results for omnidirectional noise

Noise	Speech	Leakage	SNR in dB and Improvement in dB	PSM_t in % and Improv. in dB
Pink	Female	Low	$-1.5 + 0.4$	$71.1 + 5.5$
		Medium	$-0.7 + 2.7$	$74.3 - 0.4$
		High	$5.0 - 0.3$	$78.7 - 0.1$
Pink	Male	Low	$-10.0 + 0.3$	$57.5 + 2.4$
		Medium	$-9.1 + 1.6$	$55.7 + 2.9$
		High	$-3.8 - 0.5$	$65.2 - 0.2$
Speech Noise	Female	Low[a]	$3.6 + 5.4$	$74.8 + 6.7$
		Medium	$3.5 + 2.0$	$73.2 + 4.8$
		High	$8.8 + 5.6$	$80.9 + 2.5$
Speech Noise	Male	Low [b]	$-4.8 + 4.2$	$50.6 + 12.7$
		Medium	$-5.0 + 0.6$	$54.3 - 1.3$
		High	$-0.0 + 4.7$	$70.3 + 1.9$

[a] A few recordings from this set were damaged and not used, which is why this result represents an omnidirectional noise source with some bias. The missing recordings can be seen in figure D.11.

[b] see footnote a.

7. Conclusion and Outlook

In this thesis an algorithm for enabling a differential microphone array with an in-ear microphone of a loose-fit in-ear headphone was developed. Theoretical aspects of microphone arrays and the acoustic properties of this particular kind of earphone were introduced and consequently studied by measurements.

The resulting algorithm was set up in a two-stage approach, in which the first stage compensates the passive attenuation that the in-ear microphone is subject to. It is adaptive to account for variable wearing conditions of the earphone. The second stage steers the beam pattern null to the direction of maximum interference.

An evaluation of the algorithm's performance was completed using a surround sound arrangement of speakers and an artificial head. It was found that the algorithm does not improve the signal quality when the noise and speech direction coincide, but does so for other directions. A smaller signal quality improvement is still possible if noise source is diffuse and sound arrives from all directions. The algorithm compensates for all wearing conditions, but shows the best performance in high and low leakage.

This thesis concludes with these findings, but some open questions for future research remain, a few of which shall be discussed in the following paragraphs.

The evaluation showed, that the algorithm performs worse in the medium leakage condition and it is not clear yet whether this is a shortcoming of the algorithm itself or a result of the acoustic situation at medium leakage in which multiple noise propagation paths prohibit a spatial separation.

Measurements throughout this thesis rely on artificial heads, which accurately model the acoustic properties of the human ear, but exclude one important phenomenon that occurs inside the head of a real speaking human, i.e. bone conducted speech. The bone conducted sound appears inside the ear canal as airborne sound, especially if the ear is occluded. Therefore, additional speech sound reaches the feedback microphone in a real human ear and would further improve the speech signal to noise ratio, but the effect of this on the algorithm is yet to be studied.

Of interest would be also to study the performance of the algorithm in another type of earphone. There are sealed fit headphones such as the „Apple AirPod Pro“, that use a large and acoustically transparent front vent and therefore behave similar like a loose-fit type earphone at low leakage. It should be possible to apply the algorithm here, since the only difference is a smaller leakage range to cover.

Bibliography

APx500 User's Manual (2018). Audio Precision (cit. on p. 30).

Benesty, Jacob and Jingdong Chen (Jan. 2012). *Study and Design of Differential Microphone Arrays* (cit. on p. 6).

Beranek, Leo L. (1954). *Acoustics*. 1996 edition. Acoustical Society of America. ISBN: 088318494X (cit. on p. 17).

Bristow-Johnson, Robert (2021). *Cookbook formulae for audio EQ biquad filter coefficients*. URL: https://webaudio.github.io/Audio-EQ-Cookbook/Audio-EQ-Cookbook.txt (visited on 02/02/2021) (cit. on p. 92).

Buck, Markus (2002). "Aspects of first-order differential microphone arrays in the presence of sensor imperfections." In: *European Transactions on Telecommunications* 13.2, pp. 115–122. DOI: 10.1002/ett.4460130206. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.4460130206. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4460130206 (cit. on pp. 9, 40).

DEGA-Empfehlung 101 Akustische Wellen und Felder (Mar. 2006). Deutsche Gesellschaft für Akustik e.V. URL: http://www.ak.tu-berlin.de/fileadmin/a0135/Unterrichtsmaterial/Skripte/DEGA_Empfehlung_101.pdf (cit. on p. 31).

Elko, G. W. and Anh-Tho Nguyen Pong (1995). "A simple adaptive first-order differential microphone." In: *Proceedings of 1995 Workshop on Applications of Signal Processing to Audio and Accoustics*, pp. 169–172 (cit. on p. 9).

Haykin, Simon (2002). *Adaptive filter theory*. 4th. Upper Saddle River, NJ: Prentice Hall (cit. on p. 50).

Huber, Rainer and Birger Kollmeier (2006). "PEMO-Q—A New Method for Objective Audio Quality Assessment Using a Model of Auditory Perception." In: *IEEE Transactions on Audio, Speech, and Language Processing* 14.6, pp. 1902–1911. DOI: 10.1109/TASL.2006.883259 (cit. on p. 74).

Jønsson, Søren et al. (Mar. 2003). “Simulation of Couplers.” In: *Audio Engineering Society Conference: Workshop 7*. URL: https://www.aes.org/technical/documentDownloads.cfm?docID=177 (cit. on p. 19).

McCutcheon, Peter and Robert Alcock (Aug. 2018). “Noise Cancellation System, Noise Cancellation Headphone And Noise Cancellation Method.” URL: http://uspto.report/patent/app/20200265826 (cit. on p. 50).

Tikander, Miikka (Mar. 2007). “Modeling the Attenuation of a Loosely-Fit Insert Headphone for Augmented Reality Audio.” In: *Audio Engineering Society Conference: 30th International Conference: Intelligent Audio Environments*. URL: http://www.aes.org/e-lib/browse.cfm?elib=13926 (cit. on p. 19).

Zölzer, Udo et al. (2011). *DAFX: Digital Audio Effects*. John Wiley and Sons, Ltd. ISBN: 9781119991298. DOI: https://doi.org/10.1002/9781119991298.ch2 (cit. on p. 92).

Appendix

Appendix A.

Python Filtering Framework

Several Python classes were developed for implementing, simulating and adapting of biquad filters.

A.1. Usage

Filters can be defined using a Python dictionary or JSON syntax, as shown in listing 1. To instantiate a filter, the filter definition and a sampling rate must be provided. Then the method `Filter.transferFunction(f)` returns the complex transfer function of this filter at the frequency points specified by `f`. This is demonstrated in listing 1, figure A.1 shows the result of its execution.

The class has two interfaces through which signals can be processed. There is **y** `= Filter.applySTFT(`**x**`)` for applying the filter by means of a fast convolution of a vector **x** with the transfer function and there is `y = Filter.h(x)` for applying the filter's difference equation for a single sample. A call to `Filter.h` changes the objects internal state.

A.2. Implementation

In the background each type of filter stage is implemented in a separate class and an object of the class `Filter` will create the filter stage objects as

```python
from Filter import Filter

filterdefinition = {
    "Highpass": {
        "Type": "Highpass",
        "Frequency": 100,
        "Q": 0.7
    },
    "Notch": {
        "Type": "Peak",
        "Frequency": 1000,
        "Gain": -20,
        "Q": 3
    }
}

fs = 96000
filterObject = Filter(filterdefinition, fs=fs)

from PlotTools import bode
import numpy as np

f = 2 * np.logspace(1, 4, 500)
H = filterObject.transferFunction(f)

axM, axP = bode(f, H, label="Filter Response")
axM.set_ylim((-40,10))
```

Listing 1: Example of a filter definition with the framework.

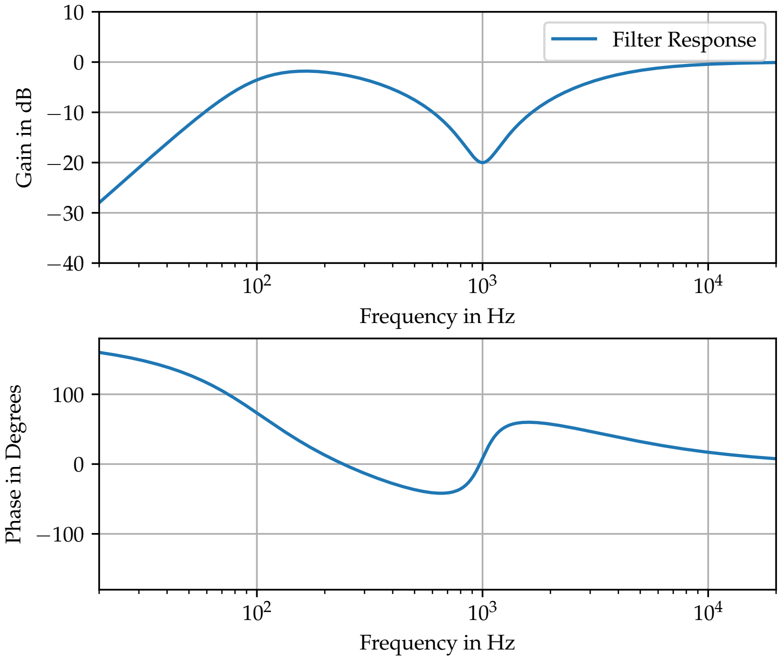

Figure A.1.: Result of executing listing 1. `PlotTools.bode()` simply produces a formatted Bode-plot from the complex response using matplotlib.

required by the filter definition. Also it multiplies the transfer function of all stages to return one total transfer function and passes the samples through each stage when using the `Filter.h` interface.

There is one biquad filter stage base class, which implements a biquad filter in direct form 1. Various definitions exist for peak, low- or highpass biquad filters. Here the following were used:

Low- and Highpass Low- and highpass are defined by a cut-off frequency f, and a quality factor Q. With the relations from Bristow-Johnson, 2021 and f_s being the sampling frequency their coefficients are calculated as stated in table A.1 and A.2 respectively.

Table A.1.: Coefficients for a lowpass biquad filter.

$\alpha = \frac{\sin(\omega_0)}{2Q}$	$\omega_0 = \frac{2\pi f}{f_s}$	$a'_0 = 1 + \alpha$
$a_0 = 1$	$a_1 = \frac{-2\cos(\omega_0)}{a'_0}$	$a_2 = \frac{1-\alpha}{a'_0}$
$b_0 = \frac{1-\cos(\omega_0)}{2a'_0}$	$b_1 = \frac{1-\cos(\omega_0)}{a'_0}$	$b_2 = \frac{1-\cos(\omega_0)}{2a'_0}$

Table A.2.: Coefficients for a highpass biquad filter.

$a_0 = 1$	$a_1 = \frac{-2\cos(\omega_0)}{a'_0}$	$a_2 = \frac{1-\alpha}{a'_0}$
$b_0 = \frac{1+\cos(\omega_0)}{2a'_0}$	$b_1 = \frac{-1-\cos(\omega_0)}{a'_0}$	$b_2 = \frac{1+\cos(\omega_0)}{2a'_0}$

Peak or Notch The peak or notch filter is defined by the center frequency f, the quality factor Q and the gain G (in dB), which is positive for the peak or negative for the notch filter. With f_s being the sampling frequency and following the definition from Zölzer et al., 2011 the peak filter's coefficients are calculated as in table A.3 and the notch filter's coefficients as in table A.4.

Table A.3.: Coefficients for a peak biquad filter.

$K = \tan\left(\frac{\pi f}{f_s}\right)$	$a_0' = 1 + \frac{K}{Q} + K^2$	$V_0 = 10^{\frac{G}{20}}$
$a_0 = 1$	$a_1 = \frac{K^2-1}{2a_0'}$	$a_2 = \frac{1-\frac{K}{Q}+K^2}{a_0'}$
$b_0 = \frac{1+\frac{V_0K}{Q}+K^2}{a_0'}$	$b_1 = a_1$	$b_2 = \frac{1-\frac{V_0K}{Q}+K^2}{a_0'}$

Table A.4.: Coefficients for a notch biquad filter.

$K = \tan\left(\frac{\pi f}{f_s}\right)$	$a_0' = 1 + \frac{V_0K}{Q} + K^2$	$V_0 = 10^{-\frac{G}{20}}$
$a_0 = 1$	$a_1 = \frac{K^2-1}{2a_0'}$	$a_2 = \frac{1-\frac{V_0K}{Q}+K^2}{a_0'}$
$b_0 = \frac{1+\frac{K}{Q}+K^2}{a_0'}$	$b_1 = a_1$	$b_2 = \frac{1-\frac{K}{Q}+K^2}{a_0'}$

A.3. Adaption

An offline adaption scheme was included in the framework to ease the tedious task of manually matching filters to the number of different target transfer functions. The adaption approach is similar to that of a particle filter. A set of N parameters of a filter array are treated like a N-dimensional vector. A random guess for each parameter is made within the range or constraint defined by the user. This guess is called a particle. An error function determines how good the guess matches the desired target transfer function. Next, a random offset is generated and added to the guess-vector, then the error function is evaluated again to check if this movement in a random direction resulted in a reduction of the error. If so, the movement will be repeated. If not, a new random direction will be attempted. After a fixed number of iterations, this process is stopped and the next particle will be treated the same way. After moving all particles to a possible minimum in that manner, the one yielding the lowest error value is selected as the best match. Selecting a sufficient amount of particles, iteration steps and a small enough step-size to scale the random direction reliably returns a good matching filter.

Example The example in listing 2 shows how to apply the filter adaption. By extending syntax already introduced in listing 1 the user can define ranges, which constrain the parameters during adaption. Using `Measurements.FleX`, an utility class for handling characterization measurements in Excel format and calculating derived transfer functions such as $\frac{A2FB}{A2FF}$, the target transfer function is loaded. The next step is to initialize the filter optimizer and call the `BiquadOptimiser.optimise` method, with specifying the bandwidth of interest, the number of particles, iterations and step size. By default, the error function is the difference of the target and the filter transfer function with an implicit weighting according to the frequency axis provided. For other applications, one might generate a subclass of `BiquadOptimiser` and implement a different error function. Figure A.2 shows the result of executing listing 2.

```python
import matplotlib.pyplot as plt
import numpy as np

filterdefinition = {
    "Gain": {
        "Type": "Gain",
        "Gain": 0, "Min Gain": -15, "Max Gain": 0,
        "Invert": True
    },
    "Highpass": {
        "Type": "Highpass",
        "Frequency": 20, "Min Frequency": 1, "Max Frequency": 100,
        "Q": 0.7, "Min Q": 0.1, "Max Q": 1.5
    },
    "Peak": {
        "Type": "Peak",
        "Frequency": 900, "Min Frequency": 500, "Max Frequency": 2200,
        "Gain": 0, "Min Gain": 0, "Max Gain": 30,
        "Q": 0.7, "Min Q": 0.1, "Max Q": 4
    },
    "Notch": {
        "Type": "Peak",
        "Frequency": 1000, "Min Frequency": 1500, "Max Frequency": 5000,
        "Gain": -20, "Min Gain": -40, "Max Gain": 0,
        "Q": 3, "Min Q": 0.1, "Max Q": 4
    }
}

from Measurements import FleX
M = FleX("Characterisation")
f, Target = M.getTransferFunction("A2FB/A2FF")

from BiquadOptimiser import BiquadOptimiser
opt = BiquadOptimiser(filterdefinition, f, fs=96000, target=Target)
opt.optimise(fmin=20, fmax=4e3, Nparticles=100, Niterations=1000, mu=0.1)
opt.plot()
```

Listing 2: Example of a matching a filter with the framework.

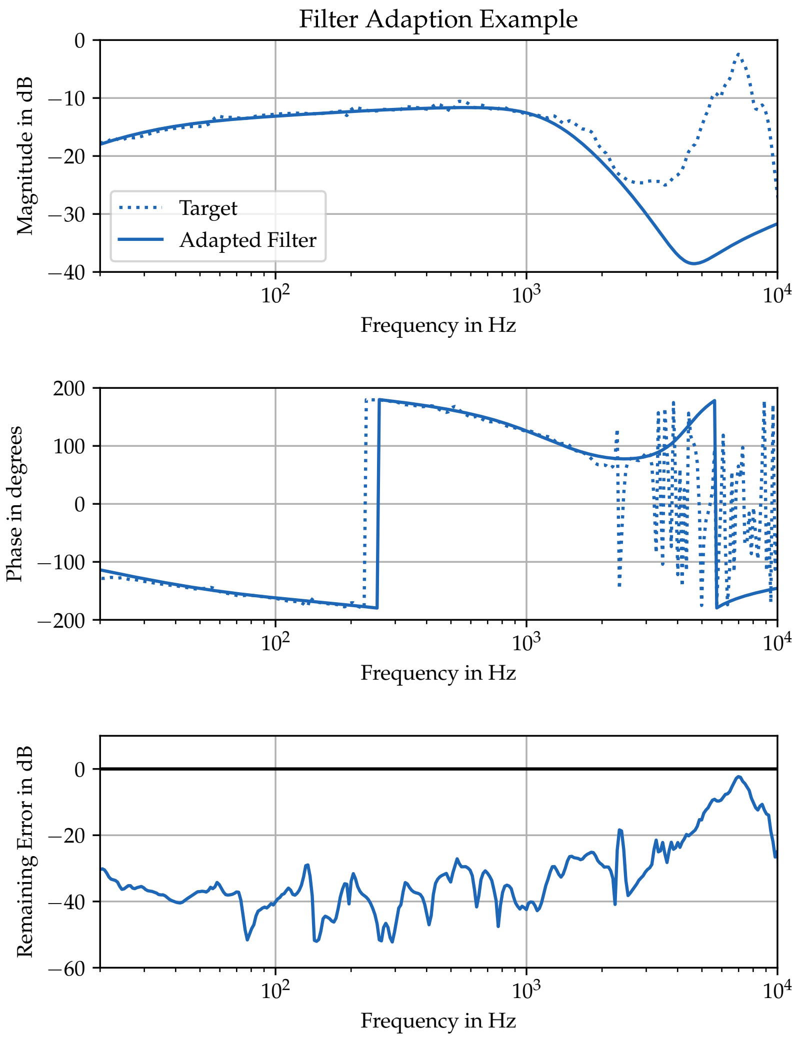

Figure A.2.: Result of executing listing 2. The plot shows a bode diagram of the target and the adapted filter's transfer function and in the bottom diagram the remaining error or in this application the resulting noise cancellation

Appendix B.

Verification of Lumped Parameter Model

Additionally to the measurements shown in figure 3.9 and 3.10 in chapter 3.2, further measurements were conducted to verify the lumped parameter model and show that the measurements are reproducible. Each measurement was repeated three times and the spread between minimum and maximum in the measurements is indicated by the colored area in the charts (see figure B.1 - B.4), while the „Measured“ line is the mean.

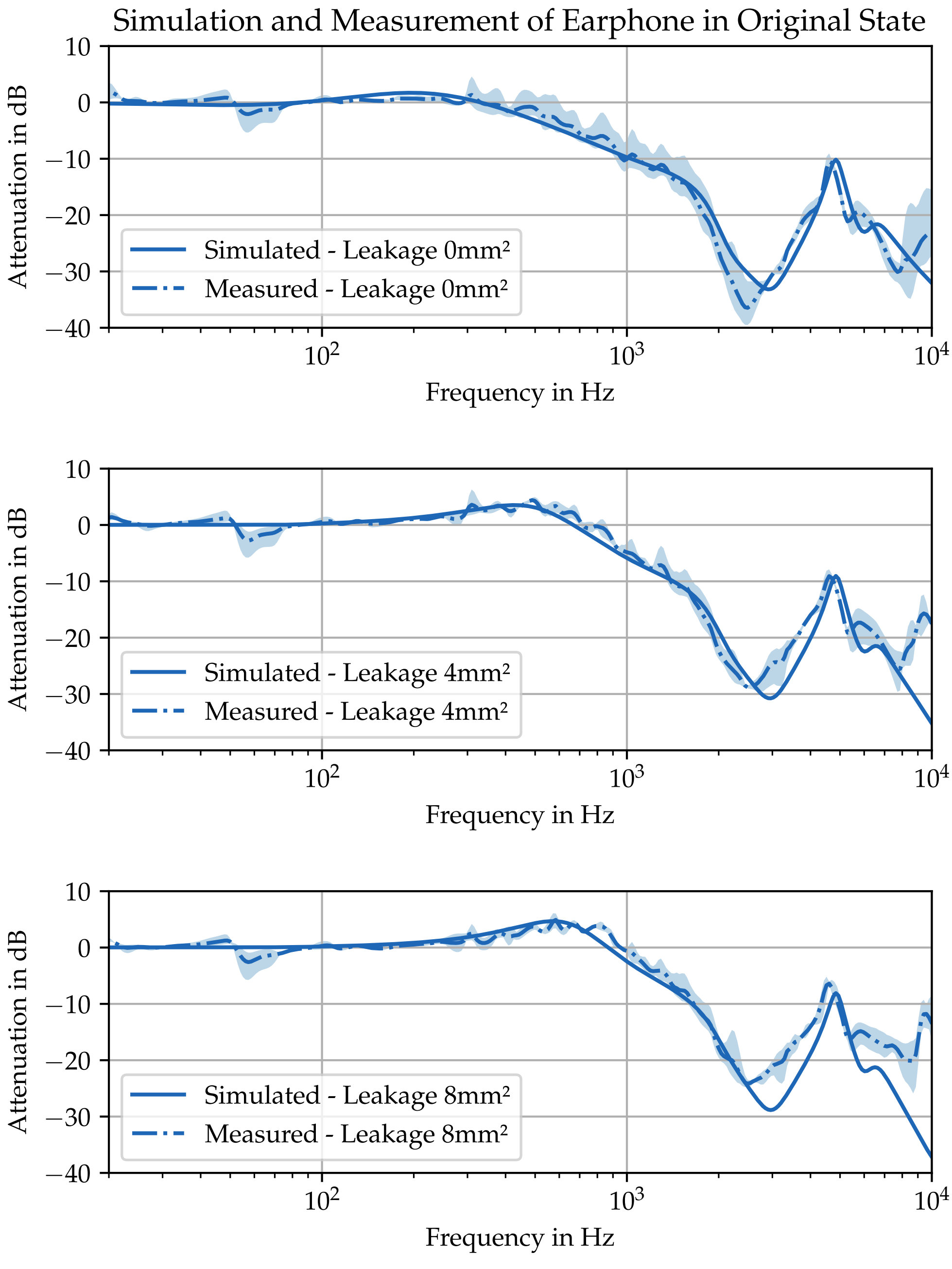

Figure B.1.: Comparison of simulated and measured passive attenuation of earphone in its original state at three different leakages. The blue area indicates the spread or reproducibility of the measurement by including the lower and upper limit out of three measurements, while the „Measured" line is the mean out of these.

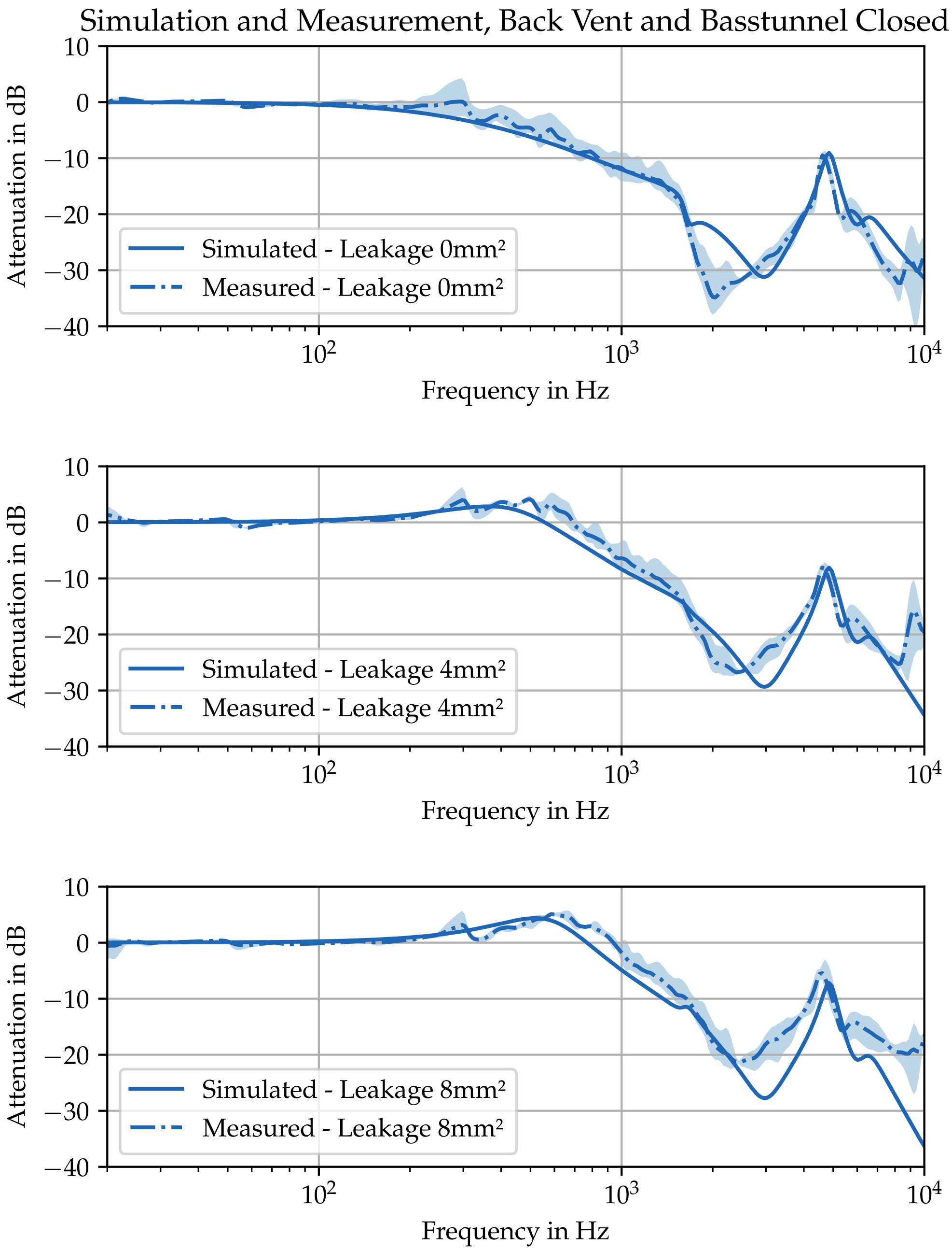

Figure B.2.: Comparison of simulated and measured passive attenuation of earphone with the back vent and bass tunnel closed.

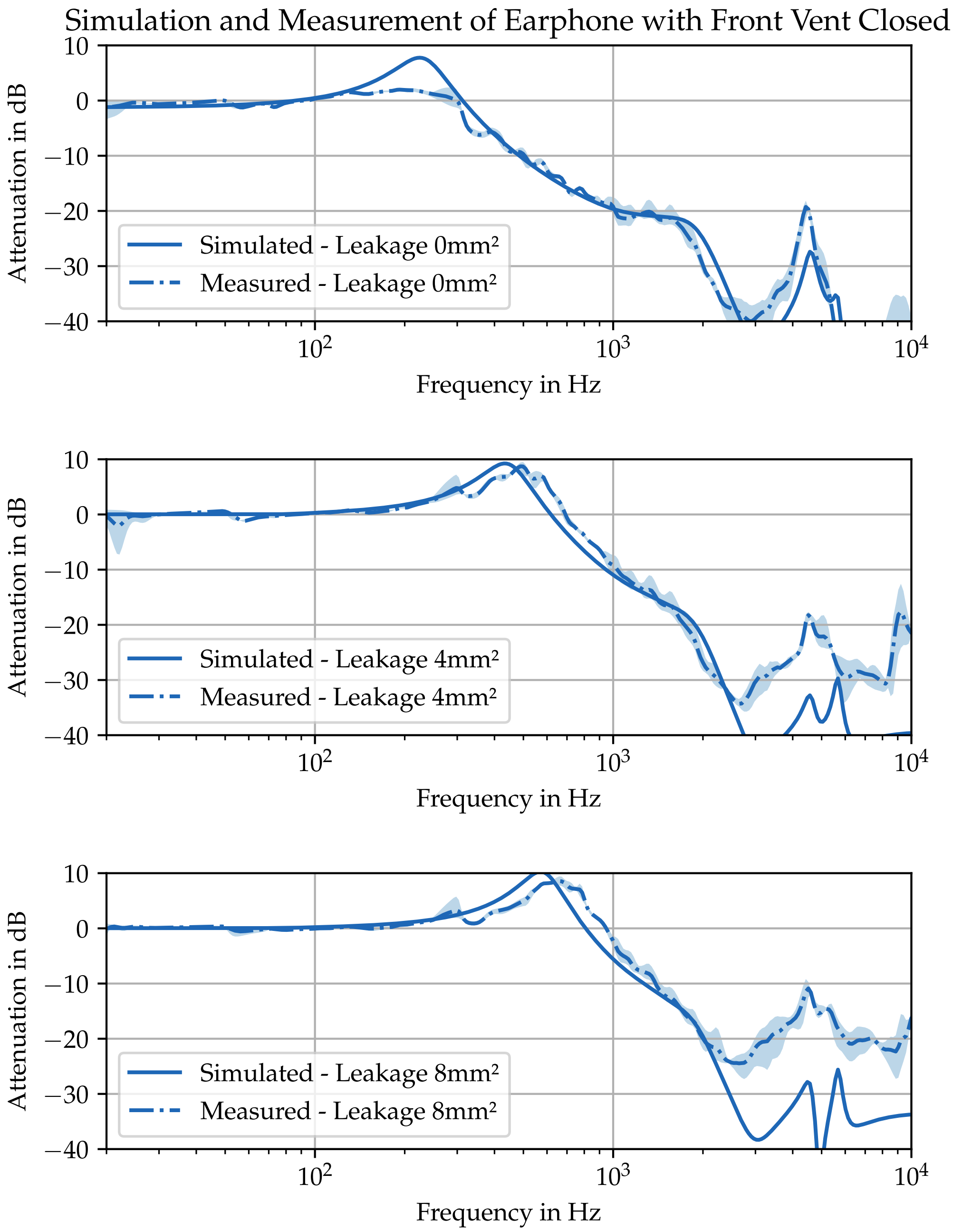

Figure B.3.: Comparison of simulated and measured passive attenuation of earphone with the front vent closed.

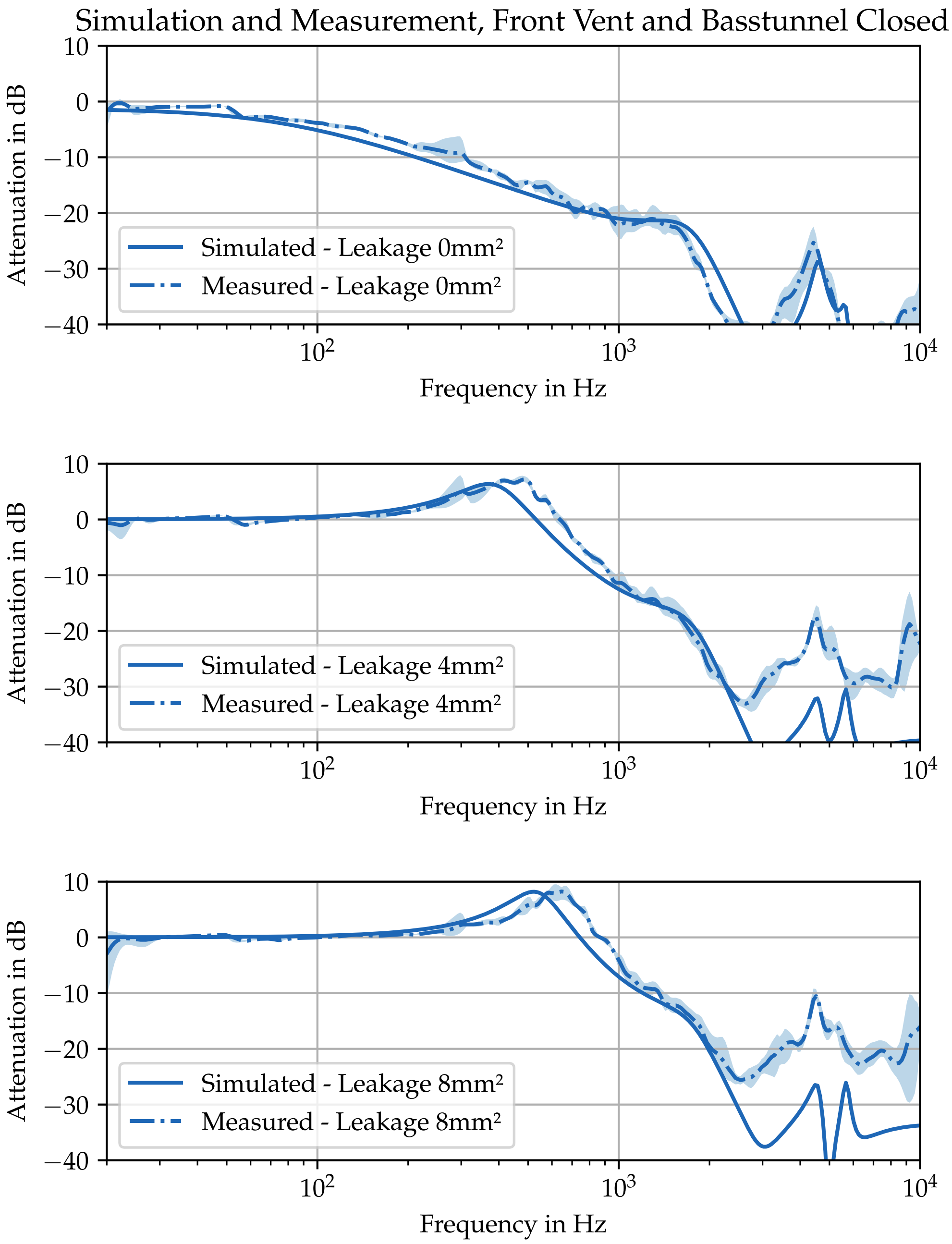

Figure B.4.: Comparison of simulated and measured passive attenuation of earphone with the front vent and bass tunnel closed.

Appendix C.

Implementation and Default Parameters of Algorithm

To give the full reference for the algorithm described in chapter 5, the complete python implementation is given in the listing 3 below. The set of parameters that was used for the evaluation in chapter 6 is in listing 4.

Implementation

```
import numpy as np
from Filter import Filter
import collections

class Algorithm():

    def __init__(self, parameters):
        """
        Setting up all algorithm parameters and ringbuffers
        :param parameters: dict containing the parameters
        """
        self.parameters = parameters
        self.fs = self.parameters["fs"]
        self.decimateBy = self.parameters["Decimate By"]
        self.mu = self.parameters["LMS Step Size"]
        self.minDelay = self.parameters["Minimum Delay"]
```

```
        self.maxDelay = self.parameters["Maximum Delay"]
        self.Ndelay = self.maxDelay - self.minDelay

        # Setup the filters
        self.highLeakFilter = \
            Filter(self.parameters["High Leak Filter"], fs=self.fs)

        self.lowLeakFilter = \
            Filter(self.parameters["Low Leak Filter"], fs=self.fs)

        self.lmsLowpassLowLeak = \
            Filter(self.parameters["LMS Lowpass"], fs=self.fs)

        self.lmsLowpassHighLeak = \
            Filter(self.parameters["LMS Lowpass"], fs=self.fs)

        self.lmsLowpassFeedback = \
            Filter(self.parameters["LMS Lowpass"], fs=self.fs)

        self.delayAdaptionBandpassFeedForward = \
            Filter(self.parameters["Delay Adaption Bandpass"], fs=self.fs)

        self.delayAdaptionBandpassFeedback = \
            Filter(self.parameters["Delay Adaption Bandpass"], fs=self.fs)

        self.compensationFilter = \
            Filter(self.parameters["Compensation Filter"], fs=self.fs)

        # Initialise the ringbuffers
        self.feedForwardMixedBuffer = \
            collections.deque(np.zeros(self.maxDelay), maxlen=self.maxDelay)

        self.feedForwardHighLeakLowpassedBuffer = \
            collections.deque(np.zeros(self.maxDelay), maxlen=self.maxDelay)

        self.feedForwardLowLeakLowpassedBuffer = \
            collections.deque(np.zeros(self.maxDelay), maxlen=self.maxDelay)
```

```python

        self.feedForwardMixedBandpassedBuffer = \
            collections.deque(np.zeros(self.maxDelay), maxlen=self.maxDelay)


        # Initialise single buffers
        self.llOut = 0
        self.hlOut = 0
        self.mixedOut = 0
        self.feedbackLowpassed = 0
        self.feedbackBandpassed = 0

        # Initialise the energy trackers
        self.inputEnergy = 0.0
        self.delayEnergy = 0.001 * np.ones(self.maxDelay-self.minDelay)

        # Initialise other variables
        self.gainLL = 1
        self.gainHL = 0.01
        self.adaptedDelay = self.minDelay

    def process(self, ff, fb, vad):
        """
        Apply the algorithm to the signals ff and fb
        :param ff: feed forward signal
        :param fb: feedback signal
        :param vad: On/Off State of Voice Activity Detection
        :return: DMA output signal
        """
        N = len(ff)
        y = np.zeros(N)

        # Filter FF Signal with High and Low Leak Filter


        for i in range(N):
            #Sample By Sample Filtering
```

```
            self.llOut = self.lowLeakFilter.h( ff[i] )

            self.feedForwardLowLeakLowpassedBuffer.append(
                self.lmsLowpassLowLeak.h( self.llOut ))

            self.hlOut = self.highLeakFilter.h( ff[i] )

            self.feedForwardHighLeakLowpassedBuffer.append(
                self.lmsLowpassHighLeak.h( self.hlOut))

            self.mixedOut = \
                self.llOut*self.gainLL + self.hlOut*self.gainHL

            self.feedForwardMixedBuffer.append( self.mixedOut )

            self.feedForwardMixedBandpassedBuffer.append(
                self.delayAdaptionBandpassFeedForward.h(
                    self.mixedOut ))

            self.feedbackLowpassed = self.lmsLowpassFeedback.h( fb[i] )

            self.feedbackBandpassed = \
                self.delayAdaptionBandpassFeedback.h( fb[i] )

            y[i] = self.compensationFilter.h(
                self.feedForwardMixedBuffer[-self.adaptedDelay] - fb[i] )


            # IIR Energy Trackers
            if vad[i] == 0:
                self.inputEnergy = 0.99999 * self.inputEnergy + 0.00001 \
                                   * np.abs(ff[i])

                for z in range(self.Ndelay):
                    self.delayEnergy[z] = 0.99999 * self.delayEnergy[z] \
                                          + 0.00001 * np.abs(
                        self.feedForwardMixedBandpassedBuffer[
```

```
                                -self.minDelay-z]
                            - self.feedbackBandpassed )

                if not i % self.decimateBy:
                    # Adaption Logic on lower rate,
                    # only if there is no voice activity
                    self.adaptedDelay = self.minDelay + \
                                        np.argmin(self.delayEnergy)

                    self.e = self.feedForwardLowLeakLowpassedBuffer[
                                 -self.adaptedDelay]\
                             *self.gainLL \
                             + self.feedForwardHighLeakLowpassedBuffer[
                                 -self.adaptedDelay]\
                             *self.gainHL \
                             - self.feedbackLowpassed

                    self.e = self.e / self.inputEnergy

                    self.gainLL = self.gainLL - \
                                  (self.mu * self.e *
                                   self.feedForwardLowLeakLowpassedBuffer[
                                       -self.adaptedDelay])

                    self.gainHL = self.gainHL - \
                                  (self.mu * self.e *
                                   self.feedForwardHighLeakLowpassedBuffer[
                                       -self.adaptedDelay])

                    if self.gainLL > 1.4:
                        self.gainLL = 1.4
                    elif self.gainLL < 0:
                        self.gainLL = 0
                    if self.gainHL > 1.4:
                        self.gainHL = 1.4
                    elif self.gainHL < 0:
                        self.gainHL = 0
```

```
        return y
```

Listing 3: Python implementation of algorithm

Parameters


```
{
  "fs": 96000,
  "Decimate By": 24,
  "Minimum Delay": 4,
  "Maximum Delay": 12,
  "LMS Step Size": 3,
  "LMS Lowpass": {
    "LP1": {
      "Type": "Lowpass",
      "Frequency": 500,
      "Q": 1
    },
    "LP2": {
      "Type": "Lowpass",
      "Frequency": 500,
      "Q": 1
    }
  },
  "Delay Adaption Bandpass": {
    "Peak": {
      "Type": "Peak",
      "Frequency": 1500,
      "Gain": 30,
      "Q": 1
    }
  },
  "Low Leak Filter": {
    "Highpass": {
      "Type": "Highpass",
      "Frequency": 10.041479818872666,
```

```
      "Min Frequency": 1,
      "Max Frequency": 30,
      "Q": 0.28433375933478156,
      "Min Q": 0.1,
      "Max Q": 0.5
    },
    "Peak 1": {
      "Type": "Peak",
      "Gain": 11.381072035200184,
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 7370.346567777666,
      "Min Frequency": 20,
      "Max Frequency": 10000,
      "Q": 3.1001445572962987,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Peak 2": {
      "Type": "Peak",
      "Gain": -19.076091925627,
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 2434.3328255298884,
      "Min Frequency": 20,
      "Max Frequency": 10000,
      "Q": 4.5674106332266575,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Peak 3": {
      "Type": "Peak",
      "Gain": 3.2461719387220502,
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 6112.953048821244,
      "Min Frequency": 20,
```

```
      "Max Frequency": 10000,
      "Q": 3.1710929425055734,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Peak 4": {
      "Type": "Peak",
      "Gain": 5.818897161223295,
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 1809.1382252660583,
      "Min Frequency": 20,
      "Max Frequency": 10000,
      "Q": 4.314203842986419,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Peak 5": {
      "Type": "Peak",
      "Gain": -11.355103776240963,
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 4375.970798359587,
      "Min Frequency": 20,
      "Max Frequency": 10000,
      "Q": 1.982030777808798,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Input Gain": {
      "Type": "Gain",
      "Gain": -12.30815635227781,
      "Min Gain": -30,
      "Max Gain": 0,
      "Invert": 1
    }
  },
```

```
    "High Leak Filter": {
      "Highpass": {
        "Type": "Highpass",
        "Frequency": 10.5464395879225,
        "Min Frequency": 1,
        "Max Frequency": 30,
        "Q": 0.3075733015343025,
        "Min Q": 0.1,
        "Max Q": 0.5
      },
      "Peak 1": {
        "Type": "Peak",
        "Gain": 5.970587961326679,
        "Min Gain": -30,
        "Max Gain": 30,
        "Frequency": 9991.983644404128,
        "Min Frequency": 20,
        "Max Frequency": 10000,
        "Q": 1.0871979275747774,
        "Min Q": 0.1,
        "Max Q": 5
      },
      "Peak 2": {
        "Type": "Peak",
        "Gain": 2.926959184800265,
        "Min Gain": -30,
        "Max Gain": 30,
        "Frequency": 6791.805481607518,
        "Min Frequency": 20,
        "Max Frequency": 10000,
        "Q": 3.9308865553581143,
        "Min Q": 0.1,
        "Max Q": 5
      },
      "Peak 3": {
        "Type": "Peak",
        "Gain": 18.361436597455835,
```

```
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 7991.4629899970105,
      "Min Frequency": 20,
      "Max Frequency": 10000,
      "Q": 0.509492469553807,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Peak 4": {
      "Type": "Peak",
      "Gain": -23.688649626850967,
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 3659.985548810906,
      "Min Frequency": 20,
      "Max Frequency": 10000,
      "Q": 3.463169577993926,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Peak 5": {
      "Type": "Peak",
      "Gain": 0.6727065916404765,
      "Min Gain": -30,
      "Max Gain": 30,
      "Frequency": 5095.426705443508,
      "Min Frequency": 20,
      "Max Frequency": 10000,
      "Q": 3.6104738435006665,
      "Min Q": 0.1,
      "Max Q": 5
    },
    "Input Gain": {
      "Type": "Gain",
      "Gain": -12.625624452118085,
      "Min Gain": -30,
```

```
        "Max Gain": 0,
        "Invert": 1
      }
    },
    "Compensation Filter": {
      "Bandpass Gain": {
        "Type": "Gain",
        "Gain": -20,
        "Invert": 1
      },
      "Bandpass Highpass": {
        "Type": "Highpass",
        "Frequency": 300,
        "Q": 0.77
      },
      "Bandpass Lowpass": {
        "Type": "Lowpass",
        "Frequency": 5000,
        "Q": 0.77
      },
      "Comp Peak1": {
        "Type": "Peak",
        "Frequency": 140.54998875692732,
        "Min Frequency": 50,
        "Max Frequency": 8000,
        "Gain": 7.2538388713331745,
        "Min Gain": -20,
        "Max Gain": 20,
        "Q": 0.10679643475184247,
        "Min Q": 0.1,
        "Max Q": 4
      },
      "Comp Peak2": {
        "Type": "Peak",
        "Frequency": 247.54916539587703,
        "Min Frequency": 50,
        "Max Frequency": 8000,
```

```
      "Gain": -12.664004279867621,
      "Min Gain": -20,
      "Max Gain": 20,
      "Q": 1.1963441312772995,
      "Min Q": 0.1,
      "Max Q": 4
    },
    "Comp Peak3": {
      "Type": "Peak",
      "Frequency": 125.92089466219456,
      "Min Frequency": 50,
      "Max Frequency": 8000,
      "Gain": 19.284422669048695,
      "Min Gain": -20,
      "Max Gain": 20,
      "Q": 0.1030680464677476,
      "Min Q": 0.1,
      "Max Q": 4
    },
    "Comp Gain": {
      "Type": "Gain",
      "Max Gain": 20,
      "Min Gain": -20,
      "Gain": 17.66942618860697,
      "Invert": 1
    }
  }
}
```

Listing 4: Default parameters of algorithm

Appendix D.

Algorithm Evaluation Plots

Below is the set of plots showing the evaluation of all measured test cases. An overview is given in table D.1.

Table D.1.: List of all test cases

Noise	Speech	Leakage	SNR	PSM$_t$
Pink	Female	Low	6.11	6.12
		Medium	D.1	D.2
		High	D.3	D.4
Pink	Male	Low	D.5	D.6
		Medium	D.7	D.8
		High	D.9	D.10
Speech Noise	Female	Low	D.11	D.12
		Medium	D.13	D.14
		High	D.15	D.16
Speech Noise	Male	Low	D.17	D.18
		Medium	D.19	D.20
		High	D.21	D.22

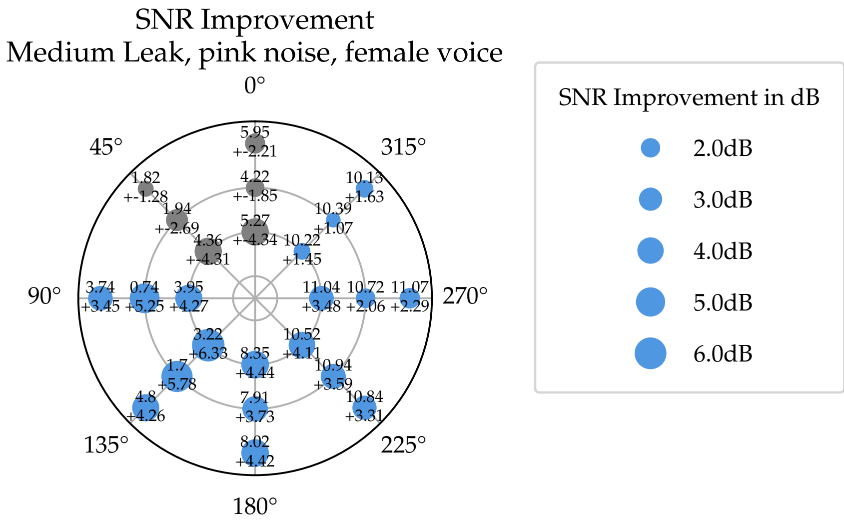

Figure D.1.: Algorithm performance for noise directions. Here showing the SNR value for the test case with pink noise, a female voice sample and a medium leakage condition.

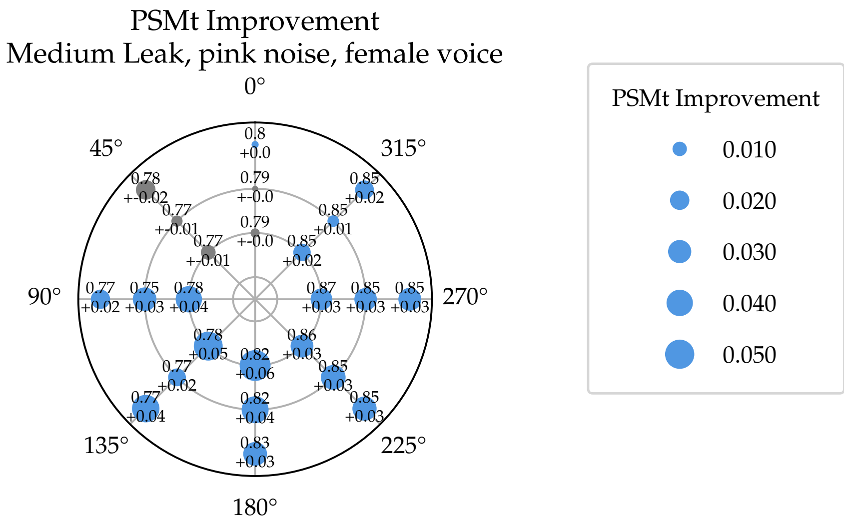

Figure D.2.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with pink noise, a female voice sample and a medium leakage condition.

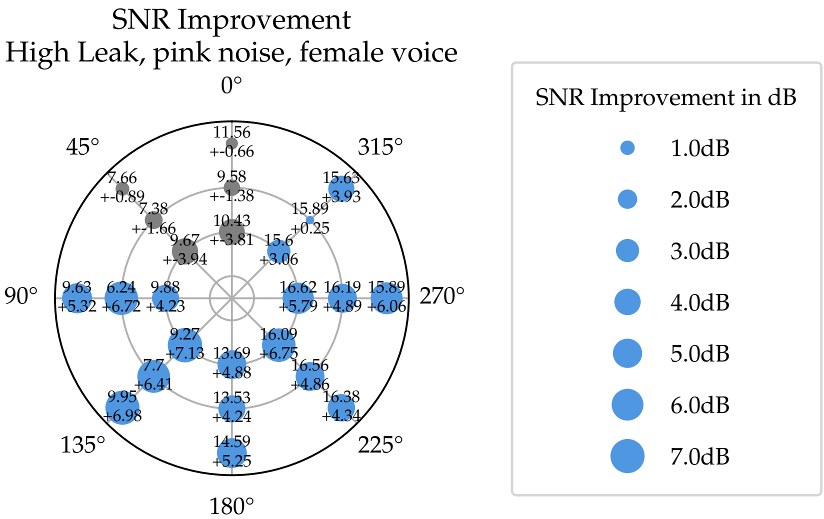

Figure D.3.: Algorithm performance for noise directions. Here showing the SNR value for the test case with pink noise, a female voice sample and a high leakage condition.

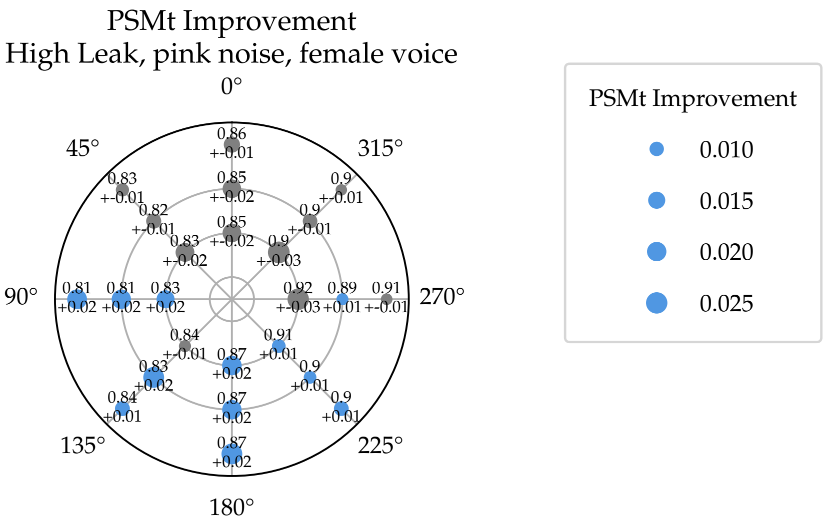

Figure D.4.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with pink noise, a female voice sample and a high leakage condition.

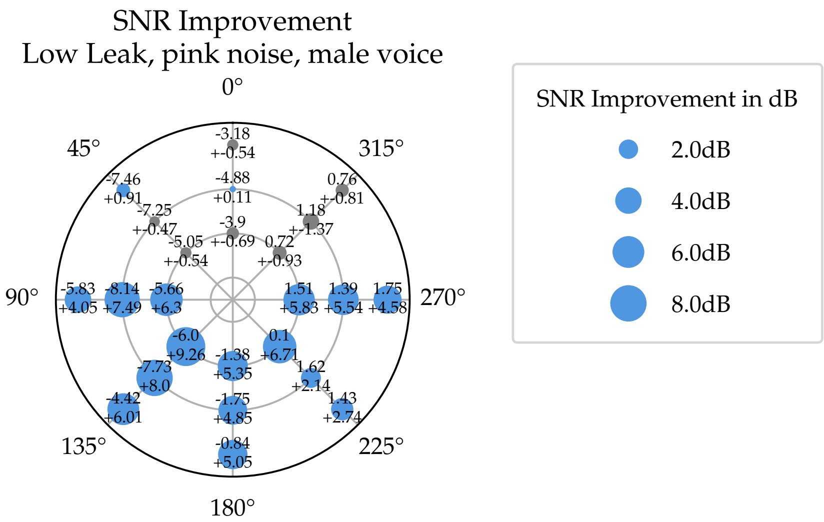

Figure D.5.: Algorithm performance for noise directions. Here showing the SNR value for the test case with pink noise, a male voice sample and a low leakage condition.

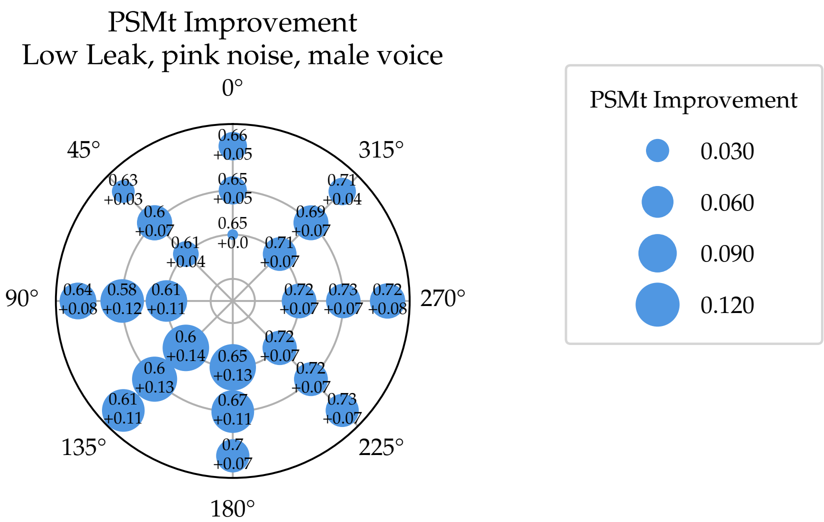

Figure D.6.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with pink noise, a male voice sample and a low leakage condition.

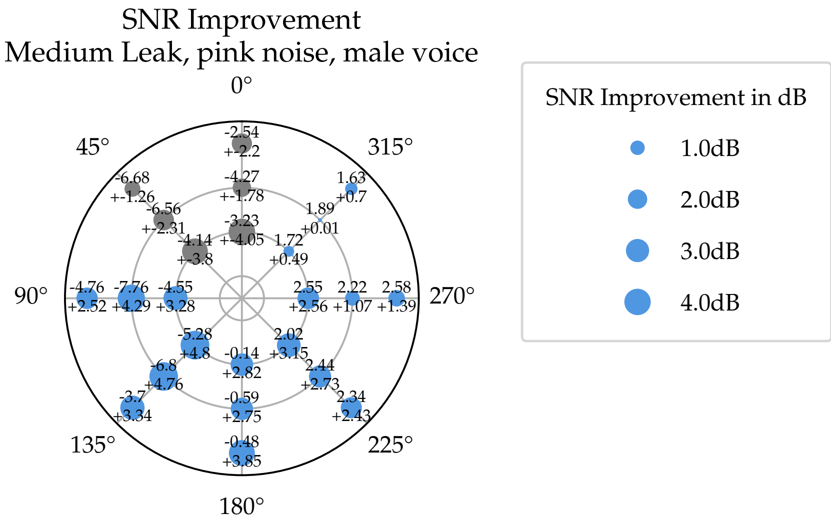

Figure D.7.: Algorithm performance for noise directions. Here showing the SNR value for the test case with pink noise, a male voice sample and a medium leakage condition.

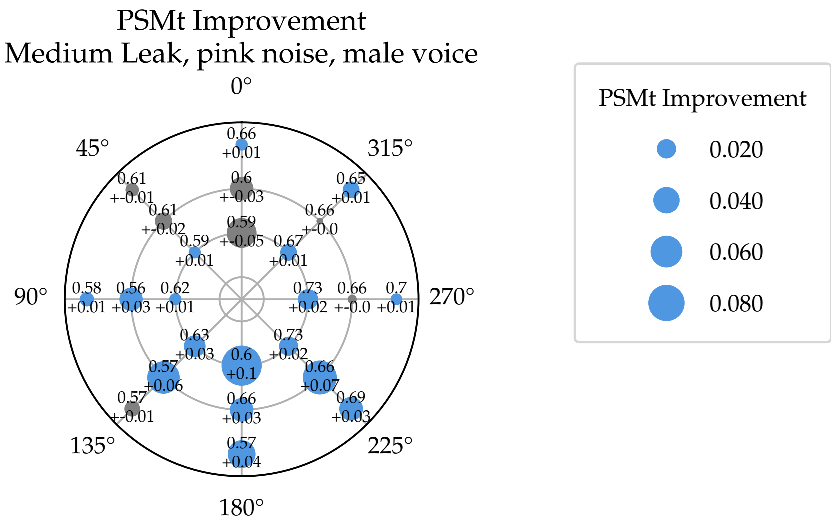

Figure D.8.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with pink noise, a male voice sample and a medium leakage condition.

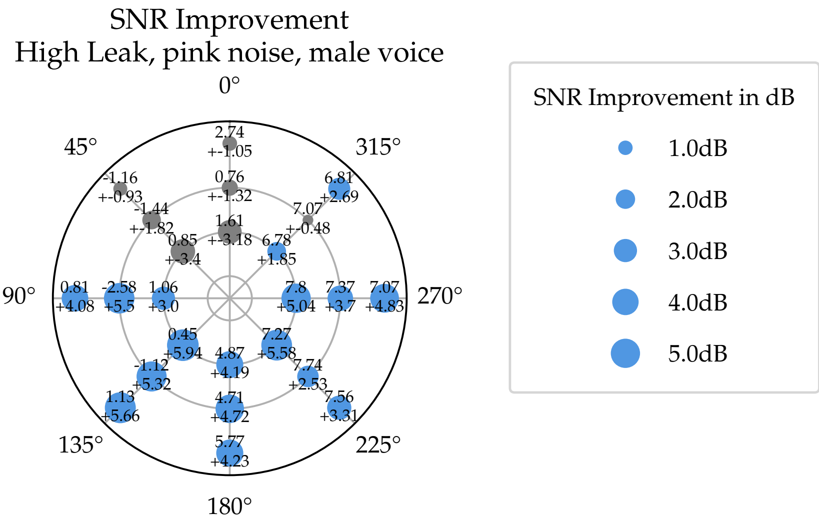

Figure D.9.: Algorithm performance for noise directions. Here showing the SNR value for the test case with pink noise, a male voice sample and a high leakage condition.

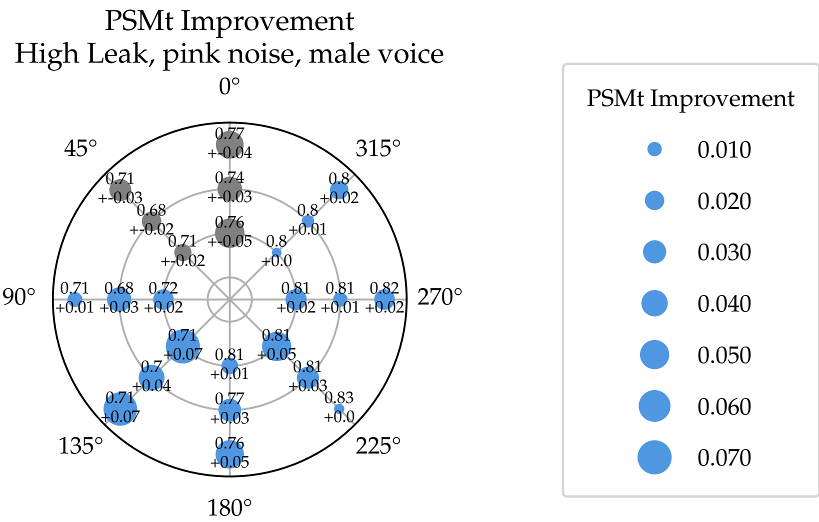

Figure D.10.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with pink noise, a male voice sample and a high leakage condition.

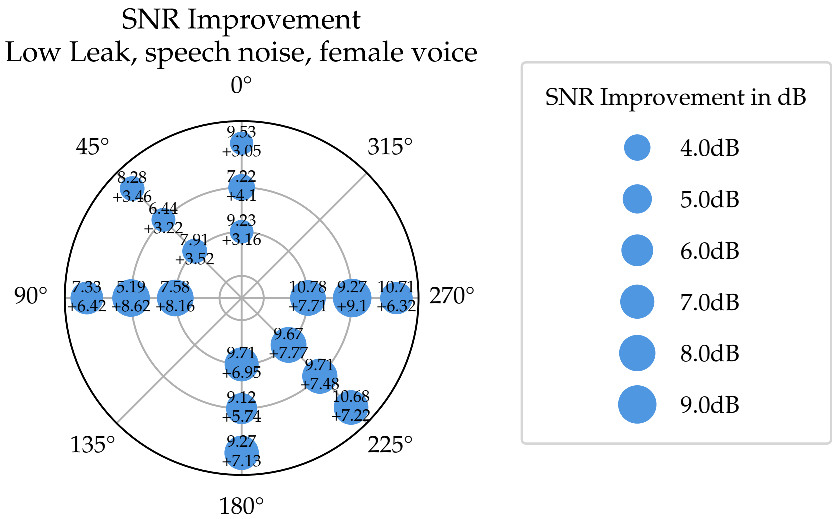

Figure D.11.: Algorithm performance for noise directions. Here showing the SNR value for the test case with speech noise, a female voice sample and a low leakage condition.

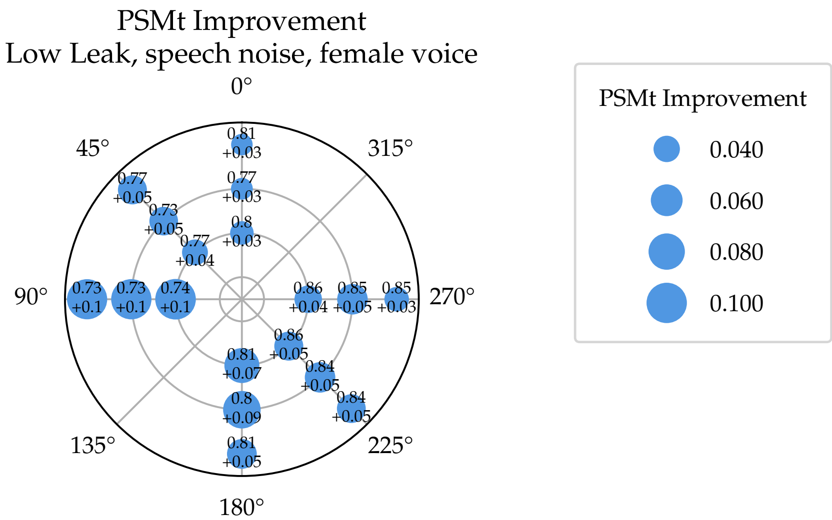

Figure D.12.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with speech noise, a female voice sample and a low leakage condition.

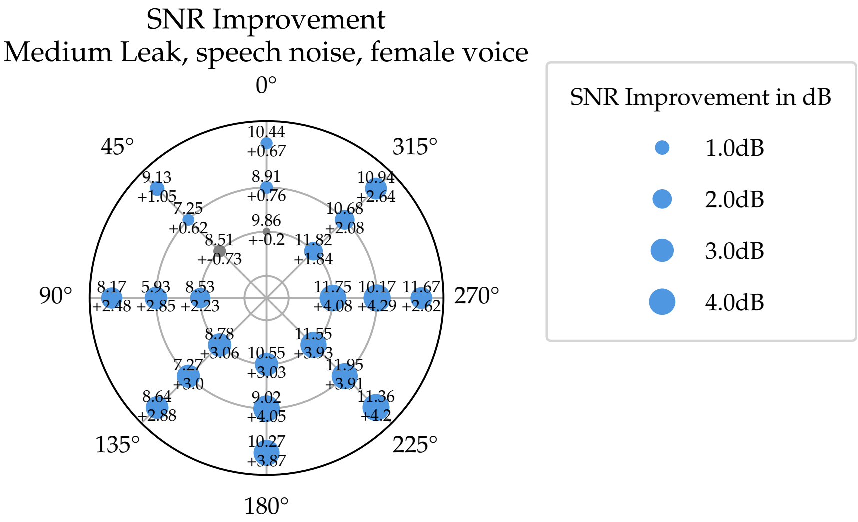

Figure D.13.: Algorithm performance for noise directions. Here showing the SNR value for the test case with speech noise, a female voice sample and a medium leakage condition.

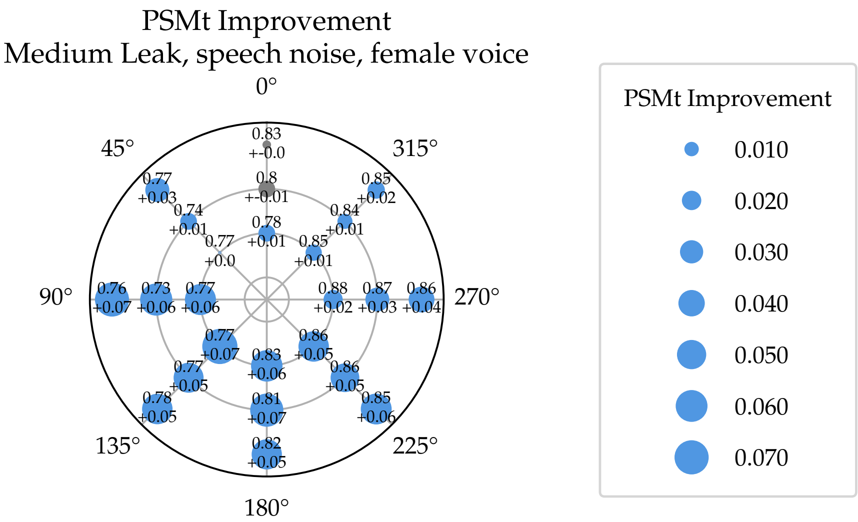

Figure D.14.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with speech noise, a female voice sample and a medium leakage condition.

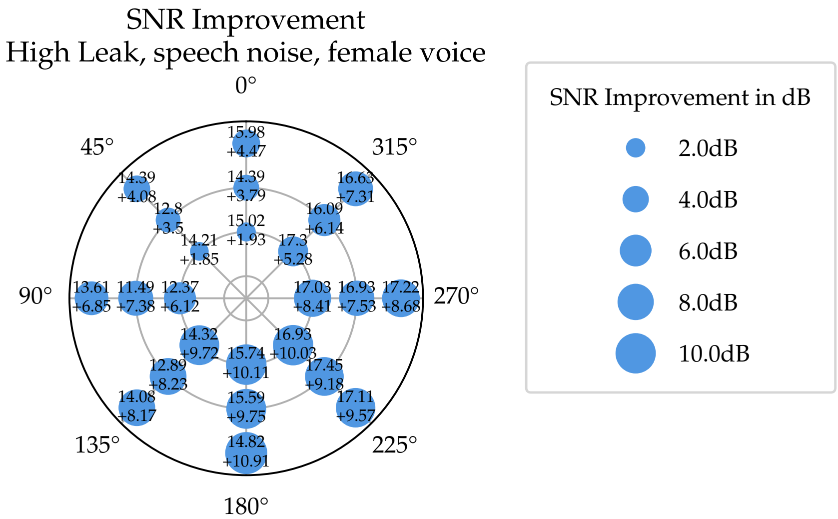

Figure D.15.: Algorithm performance for noise directions. Here showing the SNR value for the test case with speech noise, a female voice sample and a high leakage condition.

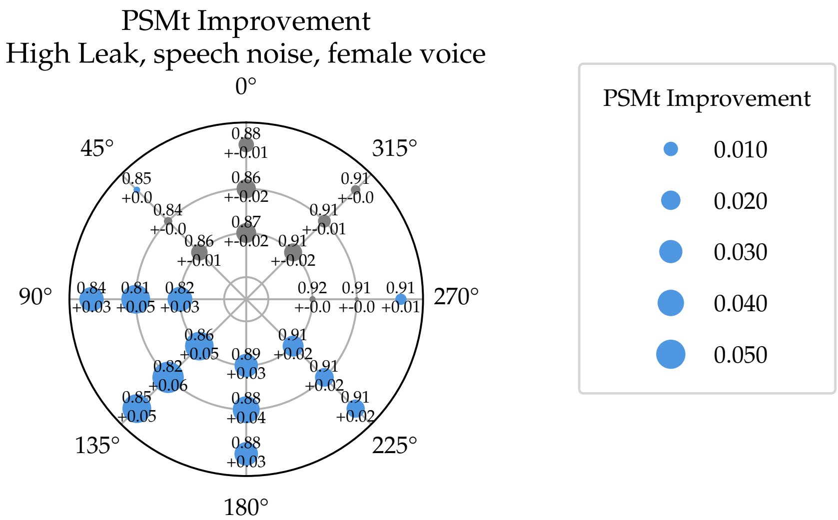

Figure D.16.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with speech noise, a female voice sample and a high leakage condition.

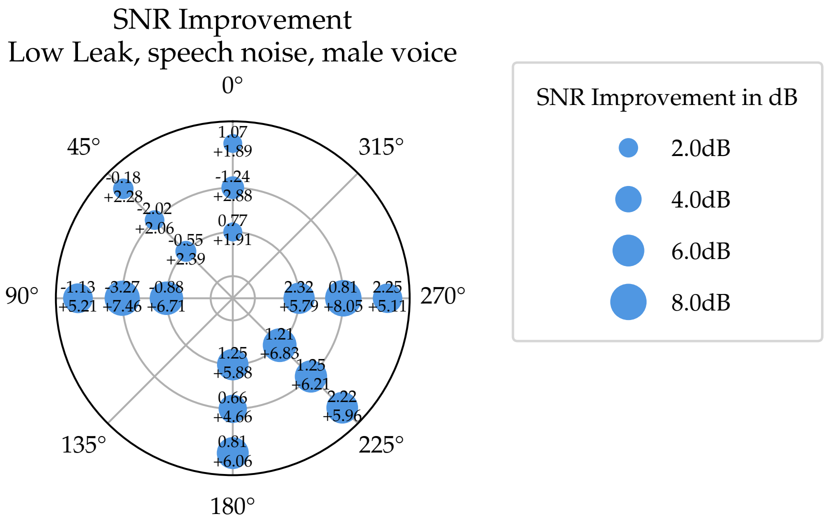

Figure D.17.: Algorithm performance for noise directions. Here showing the SNR value for the test case with speech noise, a male voice sample and a low leakage condition.

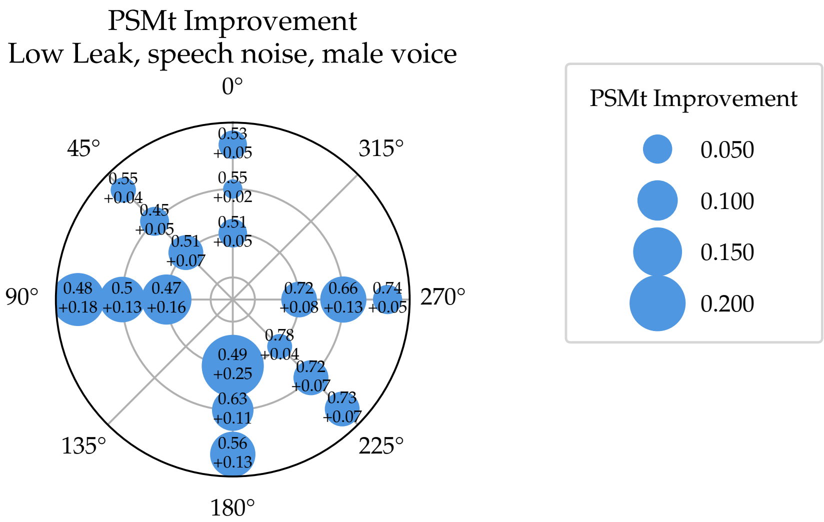

Figure D.18.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with speech noise, a male voice sample and a low leakage condition.

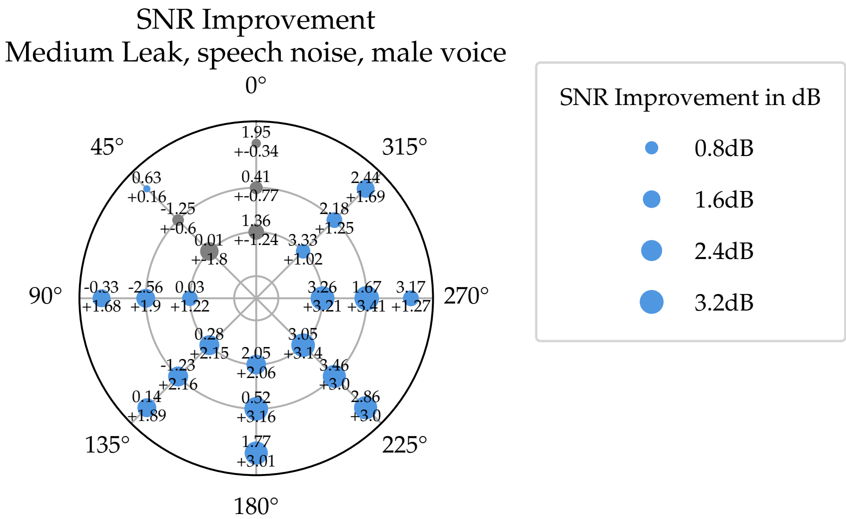

Figure D.19.: Algorithm performance for noise directions. Here showing the SNR value for the test case with speech noise, a male voice sample and a medium leakage condition.

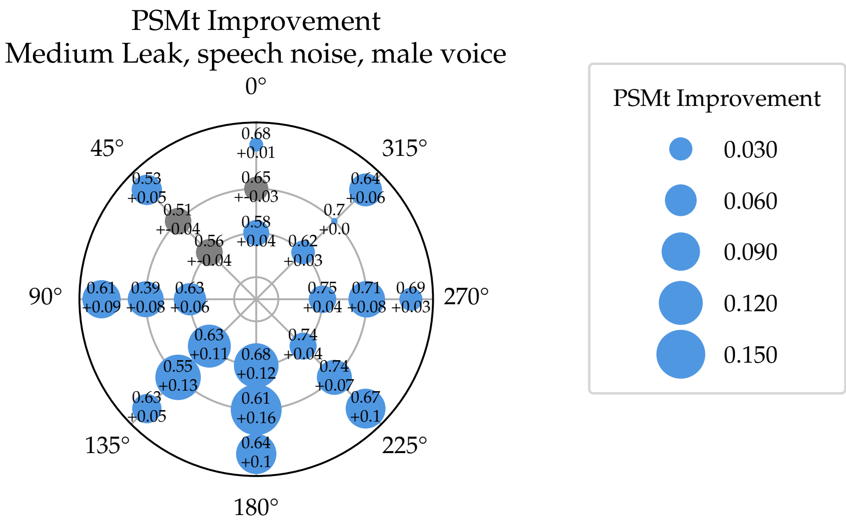

Figure D.20.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with speech noise, a male voice sample and a medium leakage condition.

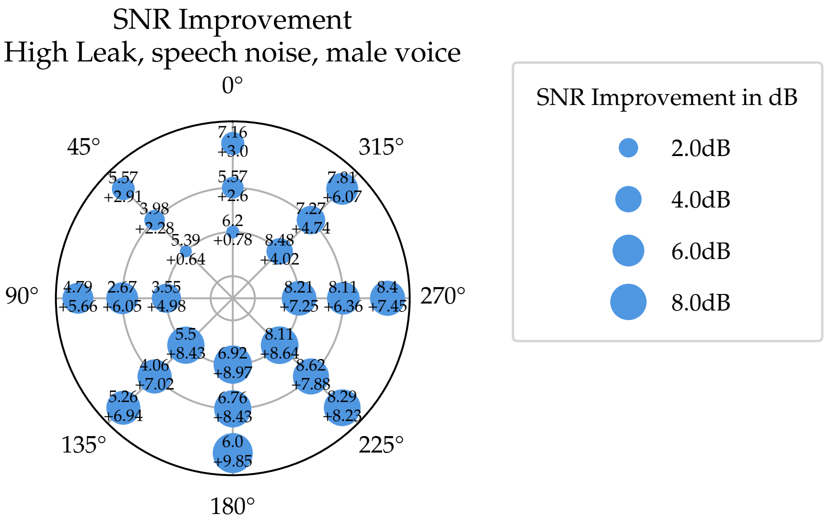

Figure D.21.: Algorithm performance for noise directions. Here showing the SNR value for the test case with speech noise, a male voice sample and a high leakage condition.

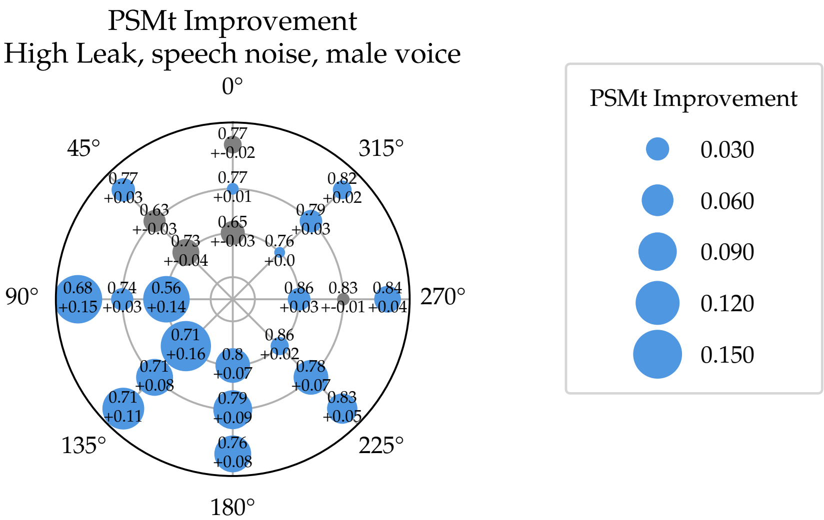

Figure D.22.: Algorithm performance for noise directions. Here showing the PSMt value for the test case with speech noise, a male voice sample and a high leakage condition.