Speech Dereverberation and Speaker Separation Using Microphone Arrays in Realistic Environments

Teun F. Krikke

Submitted for the degree of
Doctor of Philosophy

Heriot-Watt University

School of Engineering and Physical Sciences.

Awarded jointly with The University of Edinburgh

June 17, 2021

ACADEMIC REGISTRY
Research Thesis Submission

Name:	Teun F. Krikke		
School:	School of Engineering and Physical Sciences		
Version:	Final	Degree Sought:	Degree of Doctor of Philosophy

Declaration:

In accordance with the appropriate regulations I hereby submit my thesis and I declare that:

1. the thesis embodies the results of my own work and has been composed by myself
2. where appropriate, I have made acknowledgement of the work of others and have made reference to work carried out in collaboration with other persons
3. the thesis is the correct version of the thesis for submission and is the same version as any electronic versions submitted*.
4. my thesis for the award referred to, deposited in the Heriot-Watt University Library, should be made available for loan or photocopying and be available via the Institutional Repository, subject to such conditions as the Librarian may require
5. I understand that as a student of the University I am required to abide by the Regulations of the University and to conform to its discipline.
6. I confirm that the thesis has been verified against plagiarism via an approved plagiarism detection application e.g. Turnitin.

* Please note that it is the responsibility of the candidate to ensure that the correct version of the thesis is submitted.

Signature of Candidate:	Teun F. Krikke	Date:	17-06-2021

Submission:

Submitted by:	TEUN KRIKKE
Signature of Individual Submitting:	Teun F. Krikke
Date submitted:	17-06-2021

For Completion in the Student Service Centre (SSC)

Received in the SSC by (name in capitals):			
Method of Submission (Handed in to SSC; posted through internal/external mail):			
E-thesis Submitted **(mandatory for final theses)**:			
Signature:		Date:	__-__-____

Please note this form should be bound into the submitted thesis.

Abstract

This thesis concentrates on comparing novel and existing dereverberation and speaker separation techniques using multiple corpora, including a new corpus collected using a microphone array. Many corpora currently used for these techniques are recorded using head-mounted microphones in anechoic chambers. This novel corpus contains recordings with noise and reverberation made in office and workshop environments. Novel algorithms present a different way of approximating the reverberation, producing results that are competitive with existing algorithms.

Dereverberation is evaluated using seven correlation-based algorithms and applied to two different corpora. Three of these are novel algorithms (H_s NTF, Cauchy WPE and Cauchy MIMO WPE). Both non-learning and learning algorithms are tested, with the learning algorithms performing better.

For single and multi-channel speaker separation, unsupervised non-negative matrix factorization (NMF) algorithms are compared using three cost functions combined with sparsity, convolution and direction of arrival. The results show that the choice of cost function is important for improving the separation result. Furthermore, six different supervised deep learning algorithms are applied to single channel speaker separation. Historic information improves the result. When comparing NMF to deep learning, NMF is able to converge faster to a solution and provides a better result for the corpora used in this thesis.

Acknowledgements

First and foremost I want to thank my supervisor Dr. Frank Broz, for his patience, insights and support during my PhD. He was my third first supervisor and the best I could wish for. His insights and ideas on my work helped to shape this thesis.

Secondly, I want to thank my examiners Dr. Katrin Lohan and Dr. Renan Moioli for their interesting questions and discussion during the examination.

I want to thank the Centre for Doctoral Training in Robotics and Autonomous Systems for offering me the position and funding for doing this PhD. I want to thank Dr. Kartic Subr who was my second first supervisor for planting the seed that eventually grew out into this work. My thanks go to Anne Murphy for her help with everything surrounding a PhD and its travel to conferences.

My thanks go out to Stefan Hogendoorn for his comments on my work, ability to simplify the explanation of my work and for giving the me push to do a PhD. I want to thank my parents for supporting me during the process of education that eventually after many years let to doing a PhD. Finally, next to having a good supervisor it is equally important to have a supportive partner, therefore I want to thank my partner Hannah Jones for her love, care and support especially when things were not going the way they should. She has been a fantastic second reader. Lastly, I want to thank you the reader for taking the time to read this.

Contents

List of Tables

List of Figures

List of Publications

[1] Teun F Krikke, Frank Broz, and David Lane, "Who said that? a comparative study of non-negative matrix factorization techniques.," in *INTERSPEECH*, 2018, pp. 1234–1238.

[2] Teun F Krikke, Frank Broz, and David Lane, "Who said that? a comparative study of non-negative matrix factorisation and deep learning techniques," in *2017 AAAI Fall Symposium Series*, 2017.

Symbols

$^{-1}$	inverse of a matrix
^{T}	transpose of a matrix
^{H}	Hermitian transpose of a matrix
F	The Frobenius norm of a matrix
v	the speed of sound (340 m/s)
F	the number of STFT frequency bins
N	the number of STFT timesteps
X	original mixture
$\widetilde{V}$	approximation of the original mixture
V	$\lvert X\rvert^2$
W	feature matrix in NMF
H	activation matrix in NMF
K	number of components
S	the number of sources
M	the number of microphones
O	the number of look directions
A	spatial covariance matrix
D	direction of arrival coefficients matrix
P	Pressure
f	frequency
f_s	sample frequency
ω	$2\pi f$
t	time
i	imaginary number
L_s	location of the speaker

L_m	location of the microphone
R	$\lvert L_s - L_m \rvert$

Notation

G_{XY}	auto correlation between matrices X and Y^H
$\overset{\rightarrow \mathrm{t}}{H}$	H shifted by t timesteps

Chapter 1

Introduction

Speech recognition has increased in popularity with more devices using the technology and therefore, the demands of the technology are also increasing. The technology has improved to a standard whereby people who are almost fluent in one of the major languages can be understood by a speech recogniser providing the environment is suitable. With these advances, the usage has expanded and the expectations of the technology continue to increase. However, the environment in which a speech recogniser is trained has not changed. The user expects that a speech recogniser can understand them in any environment such as walking along a busy street, working in a factory or cooking a three course meal with friends in the kitchen. However, current and past research largely concentrates on making recordings of people wearing a headset and standing in an anechoic chamber, not of people in an office environment away from the microphone.

The history of speech recognition starts with Bell Labs in the 1950s where it was first used for spoken digit recognition. In this system a single voice speaking digits aloud could be recognised. Twenty years later, Carnegie Mellon introduced a system that could recognise 1000 words. In the same decade Bell Labs introduced a system that could understand multiple voices. With the start of the new millennium and the introduction of Google Voice, the accuracy of speech recognition rose to 80% in a lab environment. Google Voice distributes the processing load over various data centres instead of using the user's computer, the former approach has more processing power which allows more complicated models to be run.

By the start of 2010, speech recognition had improved again with the intro-

duction of deep learning. By making use of the graphics cards in computers, the running time of complex models decreased to almost real-time. The combination of using more accurate models with distributed computing allows speech recognition to run on devices that do not have enough computing power to run these complex models required for adequate results. Instead, it will listen for pre-configured words (so-called wake words) and will send the recording of the user through to a more powerful computer. This has raised the expectations of speech recognition to a level where people expect to be recognised in crowded or empty environments. These expectations give rise to a new set of challenges.

When addressing a robot, the speaker needs to be understood to enable the robot to execute the tasks the speaker demands from it. In general, robots are able to understand the speaker when tested in a lab setting with no other speakers around, but in real-world environments, noise, reverberation and the presence of multiple speakers make the speech recognition task more difficult. This is not only the case for robots but also for smart speakers (e.g. Amazon Alexa, Google Home, Microsoft Cortana and Apple Siri). These smart speakers work in a home or office environment where there is noise coming from various things like the kitchen, television, printer or multiple background speakers. However, robots also need to work in industrial environments where there is noise from heavy machinery. The noise makes it difficult for the speech recogniser to understand the speaker which does not happen in the ideal, clean lab, scenario.

To deal with the distorted speech obtained in varied environments, the signal needs to be pre-processed before it can be given to a speech recogniser. Pre-processing removes the reverberation, noise and other speakers to give the speech recogniser a signal that is as clean as possible. These are three challenges that are currently worked on, two of which (reverberation and other speakers) are addressed in this thesis.

There are a number of different ways of removing the noise, reverberation and de-mixing speakers. The easiest is to know how the signals are mixed and use this as a function for de-mixing the signals. This requires information about the positioning of the speakers and noise sources in the environment, the number of speakers and the influence of the environment. The latter describes the influence of the reverberation

on both the noise sources and the speech. Reverberation makes it more difficult to define where the noise or speech stops because the signal dies out slowly. There are many cases where we lack some of the parameters or do not know anything about the mixture apart from it containing speech. This latter case is called blind source separation (or blind dereverberation).

1.1 Scope

This thesis is concerned with the dereverberation of speech signals and the separation of speakers. Two categories of algorithms are used for both problems; one assumes that there is no ground truth speech signal presented to the algorithm (unsupervised learning), the other assumes there is a ground truth speech signal presented (supervised learning). For the first category, the algorithms only use the input signal to recreate the speech signals. Whereas with the second category, the algorithms are presented with the input signal and a ground truth signal to learn to recreate the transition from input to ground truth.

To test the algorithms, a corpus has been created using a microphone array (called the Acoustic Camera or AC corpus). This AC corpus contains speech in reverberant office and workshop environments. These environments create realistic scenarios in which robots and artificial assistants have to work. Apart from this corpus, three existing corpora (TIMIT, vocalization and MapTask corpora) are being used. For these corpora, a room is simulated with the exact dimension of the room used for the recordings of the AC corpus.

In the case of dereverberation, the seven algorithms (H_1, H_2, H_s, H_1 NTF, H_s NTF, WPE and MIMO WPE) require recordings from more than one microphone. These algorithms determine the correlation between microphones in order to measure the reverberation of the room. For simplicity, microphone arrays with two microphones are used in the simulated environments. The algorithms used for this problem are applied to the TIMIT corpus in order to generate a comparison with other techniques.

For the workings of the separation algorithms, it is important to know how many speakers there are in the mixture, this thesis concentrates on mixtures containing

two speakers. The separation algorithms are run in a single channel version where there is only information present from one microphone and in a multichannel version where there is information present from two microphones in a stereo configuration. The algorithms used for the single and multichannel separation are based on non-negative matrix factorisation and non-negative tensor factorisation. Deep learning is only used for the single channel separation task. As input to the algorithms three corpora (vocalization, MapTask and AC corpora) are used. In the case of the multichannel separation, the recordings of two corpora (vocalization and MapTask) are run in a simulated environment of the same dimensions as used in the AC corpus.

For measuring the performance of the algorithms, a unified framework is used to measure the difference in:

- distortion
- artefacts
- interference

between the original signal and the outcome of the algorithms as it was presented in Vincent et al. [1]. This framework allows for the comparison of the algorithms used in this thesis with the ones described in the literature. In addition to these three measurements perceptual evaluation of speech quality (PESQ) [2] is specifically used for the dereverberation algorithms in order to compare them with existing work.

Contributions

- A corpus recorded in realistic environments (office and workshop) with a high quality 72 microphone array.
- A new non-negative matrix factorisation based algorithm for speaker dereverberation.
- New modifications to multiple-input-multiple-output weighted prediction error and weighted prediction error.
- Analysis on the performance of non-negative matrix factorisation and non-negative tensor factorisation in realistic environments.
- Analysis of the importance of cost function choice with using non-negative matrix factorisation for speaker separation.
- Analysis of the importance of directionality for multichannel non-negative ten-

sor factorisation.

- Analysis on the importance of extra information in the form of convolution, sparseness or directionality in non-negative matrix factorisation to deal with noise and reverberation.

Out of scope

The microphone array is used as a finished product and does not have to be built. For the simulation of a room, a library is used as a finished product. For the separation of speakers, the assumption is that no information is available about the speakers (e.g. gender and age) and that all speakers in a mixture are of equal importance. Target speaker separation is not considered. The measurements that are used for assessing the accuracy of the algorithms are not developed by us but are the standard in the field of blind source separation with no further development considered.

1.2 Research questions

- How does the performance of algorithms compare between a simulated and a real environment with noise and reverberation and varying distance between the microphones and speaker?
  - What is the influence of the distance between the microphone and speaker?
  - What is the influence of noise and reverberation in the recordings on the separation of speakers?
- What is the performance gain of a multichannel algorithm over that of a single channel and how does the cost function influence this?
  - What is the performance gain of a multichannel algorithm over that of a single channel algorithm?
  - What is the influence of the cost function on the performance of the algorithm?
- What is the performance gain of a learning algorithm over that of a non-learning algorithm?

1.3 Structure

- Chapter 2 focuses on the workings of deep learning, non-negative matrix factorisation and weighted error prediction algorithms and makes a comparison between supervised and unsupervised learning techniques. In this chapter non-negative matrix factorisation and weighted prediction error are being explained. This chapter also focuses on beamforming algorithms and the workings of deep learning algorithms as recurrent neural networks, autoencoders, convolution neural networks and deep neural networks. The commonalities between different cost functions (Kullback-Leibler, Itakura-Saito, Euclidean and Cauchy) are described. Six different window functions used for the short-time Fourier transform are introduced. Finally, there is an overview of the measurements used for tracking the performance of the different algorithms that are used in this thesis.
- Chapter 3 presents a selection of corpora used for speech recognition research. This chapter introduces the corpora that are used for dereverberation and speaker separation in this thesis and compares these against the other corpora currently used in similar situations. An overview of the current work on dereverberation is given. This is presented with the situations in which the different techniques are tested, their performance and the corpora used for training and testing. Current work on speaker separation is presented along with the performance of the different techniques and the corpora used for training and testing.
- Chapter 4 presents the recordings made with a microphone array. The chapter starts with a description of the workings of the microphone array. This is followed by an overview of the recordings made with this device and their use cases. The chapter also describes the recording environment, instructions to the speakers and gives general details about the speakers. Post-processing of the files in order to create four different datasets within the corpus is also described.
- Chapter 5 describes the application of correlation algorithms to the dereverberation problem. The chapter starts with three correlation non-learning based methods. Two of these are expanded to work with non-negative tensor factori-

sation to create correlation learning based methods. This is followed by the application of weighted prediction error and multiple-input-multiple-output weighted prediction error applied to dereverberation. These two methods have been modified with three modifications inspired by the Cauchy distribution. The chapter concentrates on applying the different techniques to the TIMIT [3] corpus in a simulated room.

- Chapter 6 presents the work of applying non-negative matrix factorisation (NMF) and deep learning to the problem of single channel source separation. This chapter presents an overview of the different modifications applied to NMF and measures their performance. These modifications are three different cost functions (Kullback-Leibler, Itakura-Saito and Euclidean) and three different additions to the NMF algorithm (convolution, directionality and sparseness). The performance of these algorithms is tested on three different corpora (vocalization, MapTask and Acoustic Camera corpora). Next to the NMF algorithms an ideal binary filter is applied to each of the corpora. This creates a baseline to compare the algorithms against. Deep learning is applied to the vocalization corpus in the form of six algorithms. The result of the ideal binary filter and deep learning algorithms is compared with NMF.
- Chapter 7 describes the application of multi-channel non-negative matrix factorisation (NMF) and non-negative tensor factorisation (NTF) to speaker separation using microphone array recordings. This chapter uses four algorithms, three are based on NTF (time-difference of arrival NTF, covariance NTF and direction of arrival NTF) and one is based on NMF (direction of arrival NMF). The performance of these algorithms is tested on three different corpora (vocalization, MapTask and Acoustic Camera corpora). For two of these corpora (vocalization and MapTask corpora), a simulated environment and microphone array are used because these corpora have been recorded with one microphone. The simulated environment has the same dimensions as the environment used in the recordings of the third corpus (Acoustic Camera corpus).
- Chapter 8 concludes this thesis. The conclusions obtained from this research are presented. The limitations of the used techniques are being described and future work towards addressing said limitations is suggested in this chapter.

Chapter 2

Background

2.1 Introduction

For dereverberation and speech separation, corpora are used to test the different algorithms. For the collection of data, this chapter concentrates on different types of microphone arrays, from arrays filling a room to smaller sized ones implemented in robots. One of the key usages of microphone arrays is beamforming. This technique can be used for speaker localisation in the time domain or in the frequency domain. Additionally, the properties of the different corpora, such as near field speech, are explained.

The chapter concentrates on the underlying methods of the algorithms used for dereverberation and speaker separation described in this thesis. For example, the time-difference of arrival non-negative tensor factorisation algorithm (used for speaker separation) is based on beamforming to determine the time difference of arrival of speech between microphones and non-negative tensor factorisation to cluster the speech signal into different speakers. To describe the performance of these algorithms, the workings of a unified performance framework is also described.

To remove the reverberation from the speech signal, it is important to know what the room impulse response is and how this is calculated. This is described in this chapter along with the image method for simulating reverberation and the techniques used for dereverberation (weighted prediction error and multiple input multiple output weighted prediction error).

The algorithms used in this thesis for speaker separation work in the frequency

domain which means that it is important to choose a window function for the short-time Fourier transform. These window functions have different characteristics which are described in this chapter. The chapter also concentrates on the underlying principles of non-negative matrix factorisation and deep learning. For non-negative matrix factorisation and deep learning, cost functions are used to determine the convergence of an algorithm. The cost functions used in this thesis are based on the β-divergence and the symmetric α-stable distribution.

2.2 Fundamental concepts

2.2.1 Sound

Sound comes from many sources, from cars passing to trees falling and fireworks exploding, e.g. when a speaker speaks they introduce movement into the air that surrounds them. In this thesis, the main concentration is on sound as speech created by one or more people. The movement of speech through the air is picked up by our ears or microphones and translated into information that we can understand. The movements of the air are called sound waves because of the wave like pattern they have. These waves can be modelled using sine and cosine functions (see Equation 2.1) [4] travelling from the speakers to a listener (being a microphone or a person's ear). The period (T) of these waves is the time it takes for one cycle to complete (see Equation 2.2). This measure can be changed to calculate the sample rate (see Equation 2.3) which is the number of samples taken per second. In order to measure a wave accurately, there needs to be at least two samples in each cycle; one for the positive part of the wave and one for the negative. The more samples per cycle the higher the amplitude accuracy. This is important for reconstructing the original wave and extracting features from it. When there are less than two samples per cycle, the frequency of the sound wave cannot always be determined.

$$y = A \times sin(2\pi ft) \tag{2.1}$$

$$T = \frac{1}{f} \tag{2.2}$$

$$f_s = \frac{1}{T} \tag{2.3}$$

Propagation

When sound travels in an empty flat field it does not encounter any obstructions thus not creating reflections. However, when this field is changed into a valley, with high mountains on either side, the sound encounters surfaces that reflect the waves back to the original speaker who will hear this as an echo [5]. Furthermore, when the speaker is in a large factory hall with a roof, then the sound waves bounce off these surfaces creating reverberation as well as echo. With a metal roof that is heated by the sun, noise is created by the expansion of the metal plates in the roof. These surfaces absorb and reflect frequencies differently. Similar to when sound travels in air, it can also travel in water or along a metal pipe. The material in which the sound propagates determines how far the sound can travel, this is dependent on four properties: the elasticity of the material, the density of the material, the density of the air and the temperature of the air. In case of the elasticity of the material, it means that a more rigid material, for example, iron plates deform less and will reflect and propagate more of the sound than a rubber floor tile. Water has a higher density than air, therefore it is easier for sound to travel a greater distance.

Human speech

Human speech has a frequency range from approximately 85 Hz to 255 Hz (see Jurafsky et al [6]). Most of the information that is present in human speech can be found in frequencies below 10 kHz, therefore the maximum sample rate to capture all information is 20 kHz. When a corpus is specifically designed for telephone speech, the sampling rate is often lower (8 kHz). This is because the telephone calls are routed through a switchboard that filters the speech down to 4 kHz.

When a speech signal is passed through the short-time Fourier transform it switches to a complex signal containing the magnitude as the real part and the phase information as the imaginary part of the complex signal. The phase information can be used to determine where the signal is coming from (see [7]).

The magnitude, also described as the amplitude, shows which frequencies are present in the signal and contains features such as the pitch and is used for the ex-

traction of others (e.g. the Mel-frequency cepstral coefficients [8], linear prediction cepstral coefficients [9] and perceptual linear prediction [10, 11]). Magnitude can be changed to the power spectrum of the signal which is used for speaker tracking, separation and extraction. Pitch is a term that is related to the fundamental frequency of the sound which is the frequency of the vocal cord vibration. When the sound has a higher fundamental frequency, it is often perceived as having a higher pitch. The fundamental frequency or pitch can be plotted in a pitch track and gives information about tonal languages to a speech recogniser [12].

2.2.2 Supervised and unsupervised learning

The algorithms used for dereverberation and speaker separation are used as either supervised or unsupervised learning algorithms (see Alpaydin [13]). The main difference is the presence of labelled data. With supervised learning, labelled data is given to the algorithm during the training phase. The algorithm uses this information to actively adapt itself to the training data and finds information in the training data that best describes the labels.

Unsupervised learning is when unlabelled data is passed to the algorithm and the algorithm has to learn how to represent the input. The outcome of this process can be a label or other representations that are used to describe the input data.

Classification and regression are examples of supervised learning. During training, the algorithm is given training data and training labels. The algorithm learns the transformation from input data to labels. When a new unseen item is given, the algorithm then tries to match it against what it has learned from the training data and returns a label. For example, in its training phase, a recurrent neural network (RNN) is given a mixture of two speakers and a label per timestep that shows which speaker is speaking. During the test phase, the algorithm is only given a mixture of two speakers and it has to produce the labels for this mixture. This is an example of a classification algorithm. Regression, on the other hand, works with a continuous number as a label (for example predicting a temperature). Looking at the reverberation problem, an RNN is given a recording with reverberation and the same recording without reverberation. Now the network has to learn the transformation from the reverberant file back to a non-reverberant file. This transformation should

converge to the room impulse response function for that specific environment.

Clustering is often given as an unsupervised learning example. In this case, alongside the training data, the algorithm is told to divide the data into N clusters. The algorithm itself will fill in the labels. Non-negative matrix factorisation is an example of a clustering algorithm where the input (in this case a mixture of two speakers) is given to an algorithm to separate it into N clusters where N is the number of speakers.

2.2.3 Short-time Fourier transform

The short-time Fourier transform (STFT) [14, 15] transforms a signal from the time domain to the frequency domain (see Equation 2.4). By definition, it assumes that the signal is continuous. However, it cuts the signal up in overlapping blocks of a specific length (m, also called the window size). These blocks are first multiplied with a window ($w(m)$) which is, in essence, a filter. The outcome of this multiplication is passed through the discrete-time Fourier transform [15] to create a complex signal. This changes the signal from amplitude over time to frequency over time where at each timestep the signal is divided into a number of frequency bins. These frequency bins determine the number of steps in the sampling frequency, e.g. if the sample frequency is 16 kHz and there are 1024 frequency bins then there are 1024 steps between 0 and 16,000, therefore the width of each bin is between 15 and 16 Hz.

$$X = STFT(x) = DTFT(x(n-m)w(m)) \tag{2.4}$$

The advantage of the STFT is that it is invertible, meaning that it is possible to go from the frequency domain back to the time domain using the inverse STFT (iSTFT). This advantage can be exploited by using the STFT for noise reduction. In this case, the signal is passed through the STFT after which a threshold function is applied to the frequency bins thus removing the noise. The last step is to pass the signal through the iSTFT. However, this method is dependent on the threshold and can create artefacts when the wrong threshold is chosen.

Window Function

Window functions are used within the STFT to act as a filter, thus creating a windowed sequence. The window functions each have their own characteristics which makes them suited to different situations. The Hann (or Hanning) window (see Figure 2.1) [16] function is often used with speech because it deals better with the boundary conditions and reduces the influence of artefacts. However, the Hann window does introduce some smearing of frequencies. The Hamming window (see Figure 2.2) [16] is very similar to the Hann window with a difference in the ends not going down to zero. This results in a slight discontinuity in the signal. On the other hand, the Hamming window has a bigger difference between the main and side lobe in the frequency domain. This results in a better cancellation of the nearest side lobe but the window does a worse job at cancelling the others. Looking at the the difference in side lobes we see that the Barlett, Bartlett-Hann and triangular windows (see Figures 2.3, 2.4 and 2.5) [16] are very similar to that of the Hann window. The Blackman window (see Figure 2.6) has a bigger difference between the main and side lobes meaning that it deals better with the smearing of frequencies.

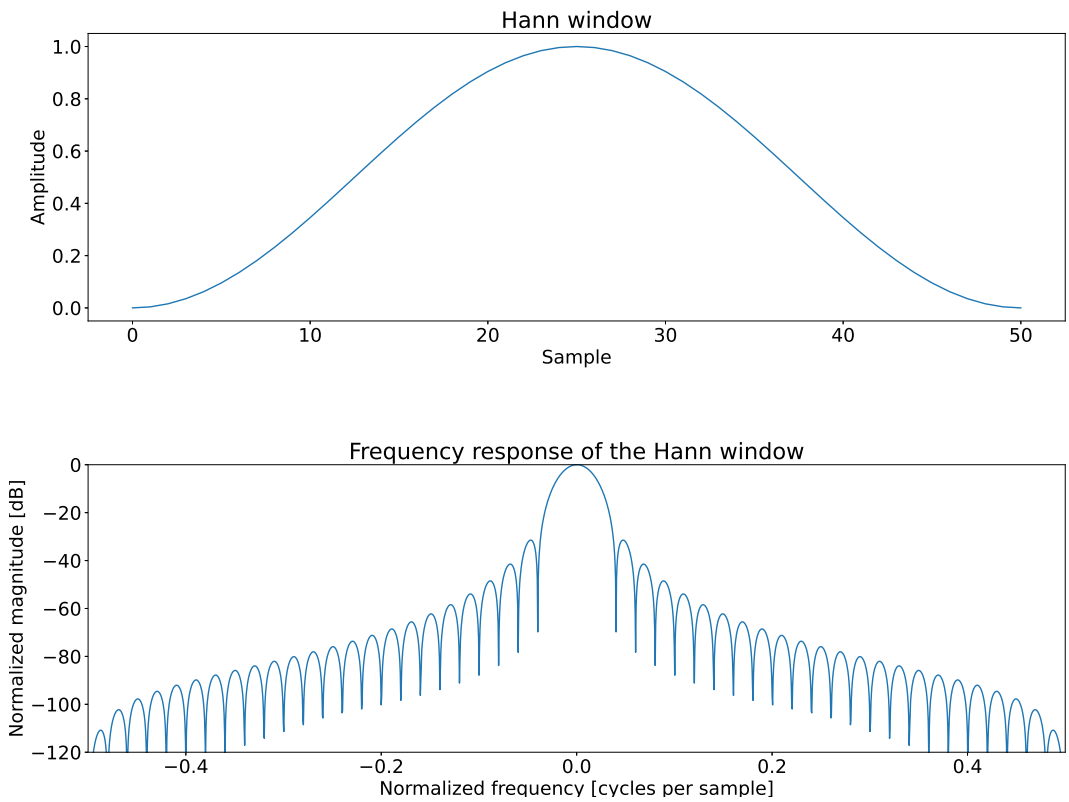


Figure 2.1: Hann window and its frequency response

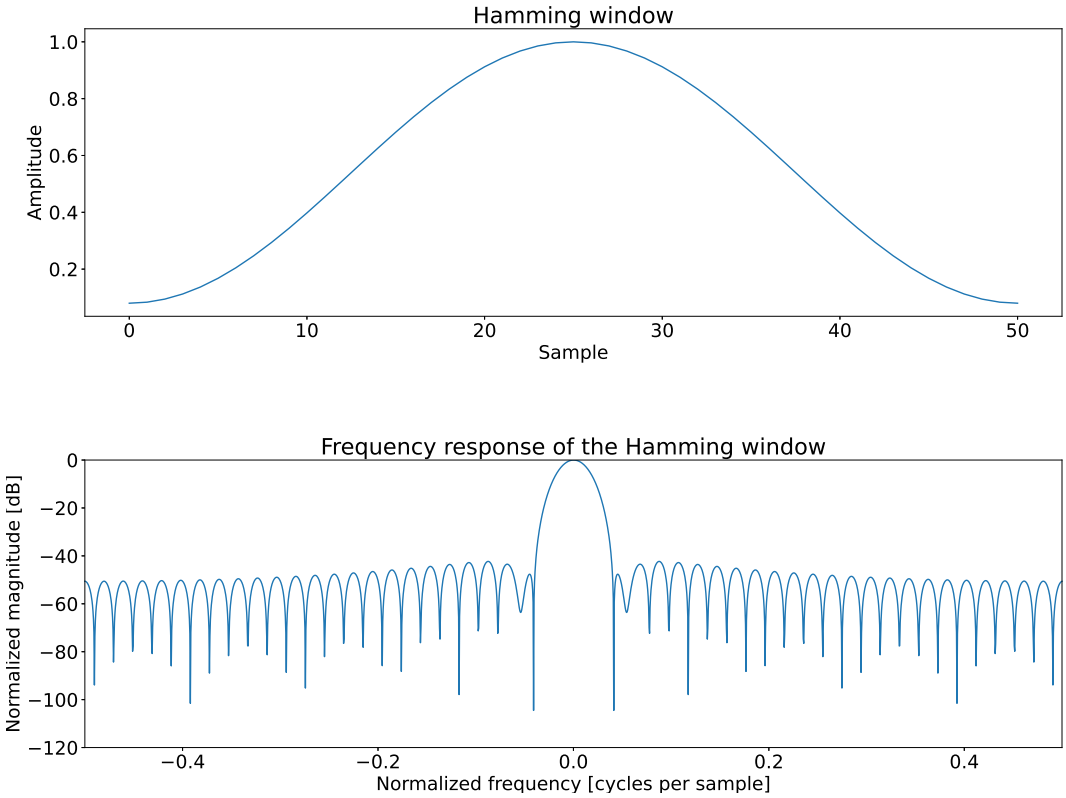


Figure 2.2: Hamming window and its frequency response

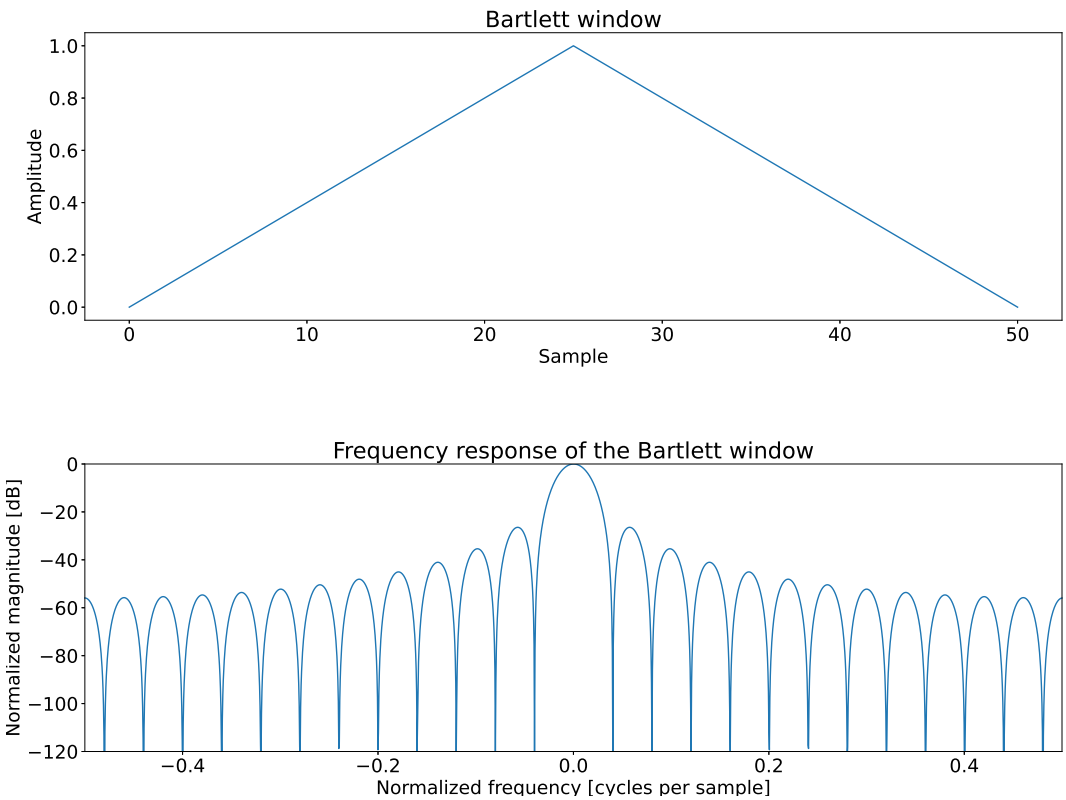


Figure 2.3: Bartlett window and its frequency response

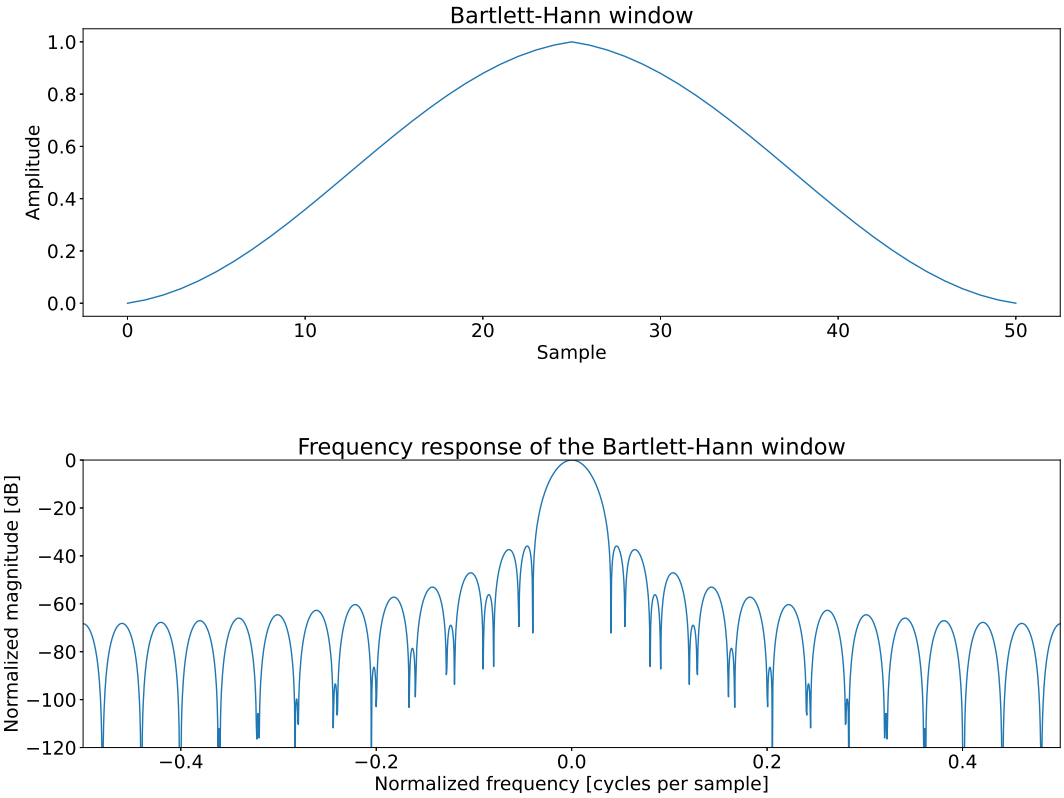


Figure 2.4: Bartlett-Hann window and its frequency response

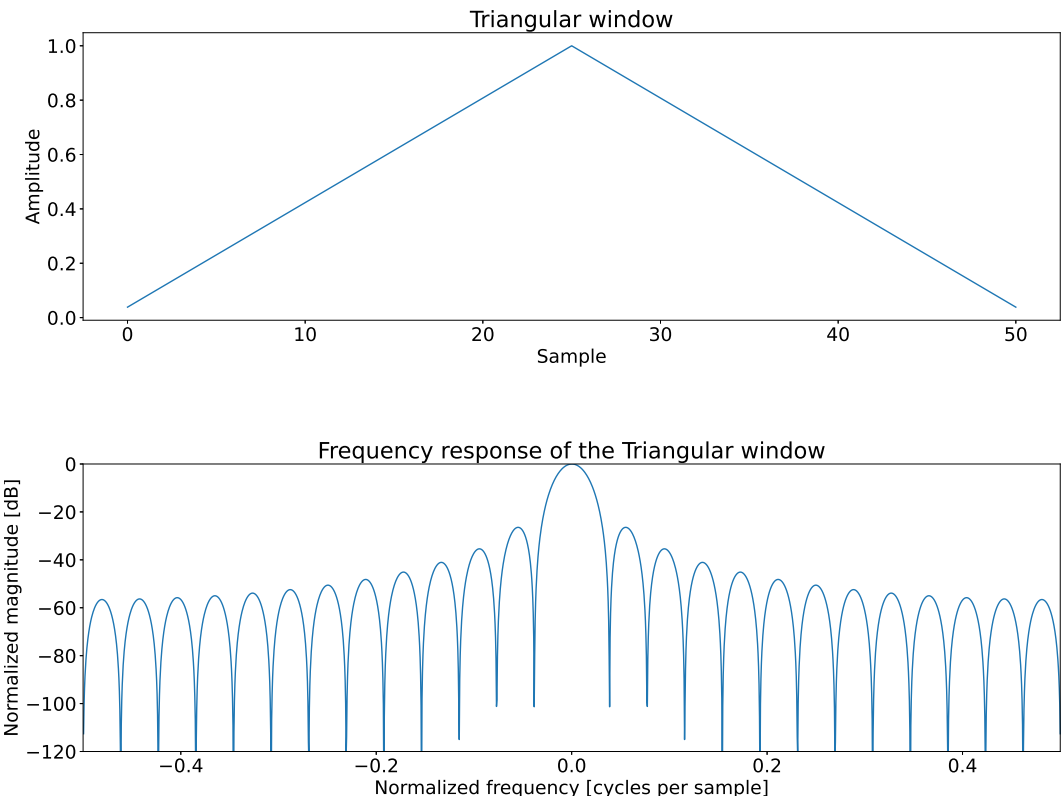


Figure 2.5: Triangular window and its frequency response

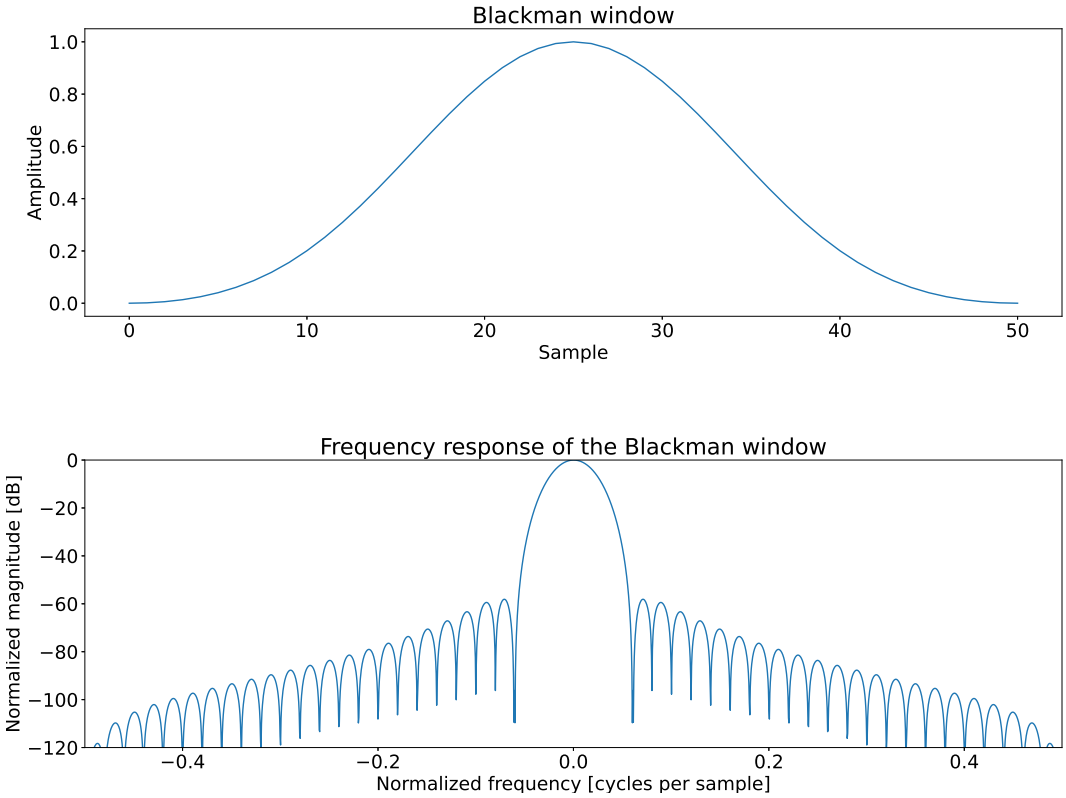


Figure 2.6: Blackman window and its frequency response

2.2.4 Cost function

Cost functions are convex (or concave) functions [17] (see Figure 2.7) that describe the difference between a true distribution (A) and an approximated version (B) of the true distribution ($D(A||B)$). The lowest point in the convex function corresponds to both distributions being the same. Cost functions can be divided into two categories, divergences and distances. Divergences (for example the Kullback-Leibler divergence) are described as pseudo-distances because they are not symmetric meaning that $D(A||B)$ is not the same as $D(B||A)$ and it does not satisfy the triangle inequality. On the other hand, distance measurements (for example Wasserstein distance) are symmetric and satisfy the triangle inequality. The triangle inequality [18] states that the sum of the lengths of any two sides is greater than the length of the remaining side. In the case of a cost function measuring the difference between a prior and posterior distribution, the triangle inequality states that the difference needs to be less than the sum of the individual updates. For example, the cost of a process that starts with a prior distribution $p(x)$ and goes to a posterior distribu-

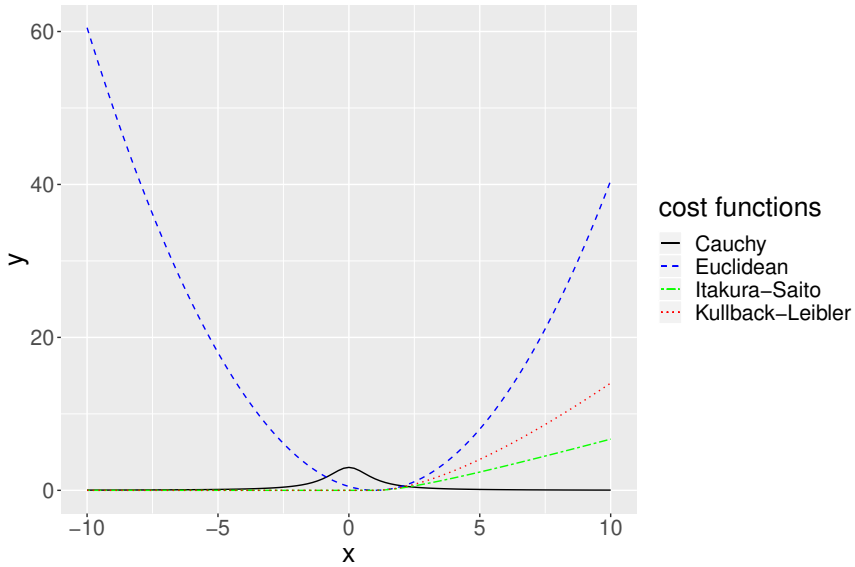


Figure 2.7: The second derivative of the Cauchy distance ($d(y|x)$, see Equation 2.16) plotted with the Euclidean, Kullback-Leibler and Itakura-Saito divergences ($d(y|x)$, see Equations 2.8 to 2.11). All measurements assume that y=1.

tion $p(x|I)$ is measured. When the posterior distribution is simultaneously updated with two parts of new information y_1 and y_2, creating the new posterior distribution $p(x|y_1, y_2, I)$ then the difference between the old posterior and the new one needs to be less than the sum of the individual updates $D(p(x|y_1, I)||p(x|I)$ and $D(p(x|y_2, y_1, I)||p(x|y_1, I))$ to satisfy the inequality (see Equation 2.5).

$$D(p(x|y_1, y_2, I)||p(x|I)) < D(p(x|y_2, y_1, I)||p(x|y_1, I)) + D(p(x|y_1, I)||p(x|I) \quad (2.5)$$

This section concentrates on the β-divergence which is a special class of divergences containing the squared Euclidean, Kullback-Leibler [19] and Itakura-Saito [20] divergences which are used for non-negative matrix factorisation. Note that the squared Euclidean distance differs from the Euclidean distance in that the latter is symmetric and the former is considered a divergence because of the triangle inequality. Another cost function that is described is the Cauchy divergence which is used for weighted error prediction.

β-divergence

One of the, if not the main, reasons for using the β-divergence [21, 22] is its robustness to outliers. The β-divergence (see Equations 2.6 and 2.8) is a subset

of the Bregman distances [23, 24] and includes the following divergence measurements: the squared Euclidean distance (see Equation 2.9), Kullback-Leibler (KL, see Equation 2.10) divergence and the Itakura-Saito (IS, see Equation 2.11) divergence. Where in Equations 2.6 and 2.8 $p(x)$ is the prior distribution, $q(x)$ is the posterior distribution and β is a real number. Equation 2.6 is said to be continuous for $\beta = 0$ and $\beta = 1$ (see Equation 2.7). Therefore, Equation 2.8 includes the special cases for $\beta = 0$ and $\beta = 1$ which are the Itakura-Saito and the Kullback-Leibler divergences respectively.

$$D_B^{\beta}(P||Q) = \int \left(p(x) \frac{p(x)^{\beta-1} - q(x)^{\beta-1}}{\beta - 1} - \frac{p(x)^{\beta} - q(x)^{\beta}}{\beta} \right) d\mu(x) \tag{2.6}$$

$$\begin{aligned} &\forall \beta_0 \in \mathbb{R} \\ &\forall p, q \in \mathbb{R}_+ \\ &D_B^{\beta_0}(P||Q) = \left[\lim_{\beta_0 \to \beta} D_B^{\beta_0}(P||Q) \right] \end{aligned} \tag{2.7}$$

$$D_B^{\beta}(P||Q) = \begin{cases} \frac{1}{\beta(\beta-1)} \int \left(p^{\beta}(x) + (\beta - 1) q(x)^{\beta} - \beta p(x) q(x)^{\beta-1} \right) d\mu(x) \beta \neq 0, 1 \\ \int \left(p(x) log \frac{p(x)}{q(x)} - p(x) + q(x) \right) d\mu(x) \beta = 1 \\ \int \left(log \frac{q(x)}{p(x)} + \frac{p(x)}{q(x)} - 1 \right) d\mu(x) \beta = 0 \end{cases} \tag{2.8}$$

$$d_{EUC}(x|y) = \frac{1}{2}(x - y)^2 \tag{2.9}$$

$$d_{KL}(x|y) = x log \frac{x}{y} - x + y \tag{2.10}$$

$$d_{IS}(x|y) = \frac{x}{y} - log \frac{x}{y} - 1 \tag{2.11}$$

It has all of the Bregman divergences properties:

Convexity. The second derivative of the function is greater than zero for all y where $\frac{d^2}{dx^2} f(x, y) \geq 0 \ \forall \ x, y \in \mathbb{R}^n$

Linearity. The function is linear with respect to its negative coefficients.

Duality. The function has a convex conjugate.

Being part of the Bregman distances also means that the set is one on one in correspondence to that of the regular exponential distributions. The IS divergence is in correspondence with a gamma distribution and the KL divergence with a Poisson distribution. The Gaussian distribution corresponds with a squared Euclidean distance. The reason for the β-divergence being a subset has to do with the Bregman generation function, which cannot create the Itakura-Saito or the Kullback-Leibler divergences. This is only the case when the Bregman generation function is a Legendre type, meaning that the function is locally bounded and strictly convex.

The β parameter defines which divergence is used and also defines on which data values the divergence relies. Choosing the optimal parameter for β depends on the characteristics of the data. When $\beta > 0$, the factorisation of the data relies more on the largest data values and less precision is expected in the estimation of the small values. When $\beta < 0$ the opposite happens.

In the following three divergences (see Figure 2.8) concentrate on two concepts: statistical efficiency and statistical robustness (see Box et al. [25]). When a divergence is efficient it means that the outliers have less influence on the result which allows the divergence to produce more precise estimates of the similarity between two probability distributions. Robust divergences divert less (lower bias) from the mean of the distributions, meaning that outliers have less influence on the result. Both these concepts are dependent on the weights given to the values of the two distributions.

Squared Euclidean

The squared Euclidean ($\beta = 2$) distance is based on the Euclidean distance however, it does not satisfy the triangle inequality. Therefore, it is considered to be a divergence and not a distance. This cost function is more robust to outliers however it is less efficient meaning that severe outliers have a greater influence on the end result thus creating less precise estimates of the parameters of the underlying linear relationship.

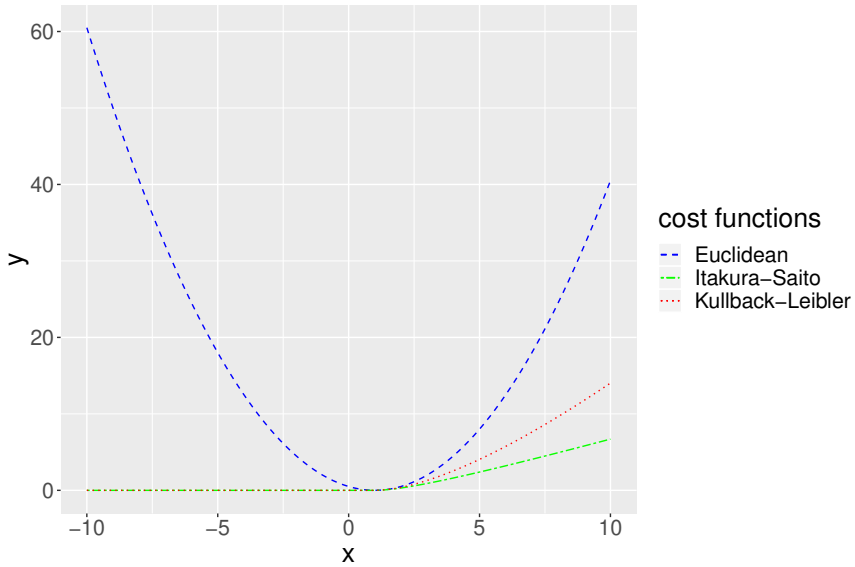


Figure 2.8: The Euclidean, Kullback-Leibler and Itakura-Saito divergences $d(y|x)$ (see Equations 2.8 to 2.11) assuming that y=1.

Kullback-Leibler

The KL ($\beta = 1$) is less robust and more efficient with outliers resulting in a cost functions that treats outliers equally in its calculations resulting in a lower influence of severe outliers on the result.

Itakura-Saito

The IS-divergence ($\beta = 0$) is the only one of the β-divergences that is scale invariant and show similar robustness and efficiency as the KL divergence. Note that this is different from the robustness and efficiency describe for the KL divergence and the squared Euclidean distance. Scale invariance means that $D(\alpha A||\alpha B) = D(A||B)$.

Cauchy divergence

The complex Cauchy distribution is part of the complex symmetric α-stable (SαS) distributions. This distribution is best defined by its characteristic function $\psi(\omega) = e^{j\delta\omega - \gamma|\omega|^\alpha}$, where α is the characteristic exponent restricted to the values $0 < \alpha \leq 2$, δ runs from $-\infty < \delta < \infty$, this is the location parameter and $\gamma > 0$ is the dispersion of the distribution.

The limit of α is important because it influences the characteristic function. When $\alpha < 0$, the characteristic function becomes $e^{|k|}$. As $|k|$ goes to ∞, the characteristic function goes to unity because it goes to e^0. This characteristic function

cannot be integrated and the Fourier transform cannot be used to get the probability density function. In the other case, when $\alpha > 2$, the inverse Fourier transform cannot be proven to be non-negative, however the probability density function is non-negative. There are two important special cases of (SαS) when α is 1 or 2, namely the Cauchy ($\alpha = 1$) and the Gaussian ($\alpha = 2$) distributions.

The multivariate SαS distributions, to which the complex Cauchy distribution belongs, is used for modelling signals among other things. The distribution is isotropic with respect to the point (δ_1, δ_2).

A complex random variable $X = X_1 + jX_2$ is considered to be SαS when its parts (X_1 and iX_2) are jointly SαS and the characteristic function from which it was drawn can be written as Equation 2.12.

$$\psi(\omega) == E^{i\Re[\omega X^*]} = E^{i(\omega_1 X_1 + \omega_2 X_2)} = exp\left[-\int_{S_2} |\omega_1 x_1 + \omega_2 x_2|^\alpha d\Gamma_{X_1,X_2}(x_1, x_2)\right] \tag{2.12}$$

In Equation 2.12 ω is $\omega_1 + j\omega_2$, $\Re$ is the real part operator, and Γ_{X_1,X_2} is a symmetric measure on the unit sphere S_2, called the spectral measure of the random variable X. A complex random variable $X = X_1 + jX_2$ is isotropic if and only if (X_1, X_2) has a uniform spectral measure. Several complex random variables are jointly SαS if their real and imaginary parts are jointly SαS. In the theory of second-order processes, the concept of covariance plays an important role in problems of linear prediction, filtering and smoothing of for example statistical signal processing problems.

As a cost function the Cauchy divergence is non-convex in the sense that the second derivative (see Equation 2.16) is not greater than zero for the whole domain (see Figure 2.9).

$$P_\gamma(x_1, x_2) = \frac{\gamma}{2\pi(x_1^2 + x_2^2 + \gamma^2)^{3/2}} \tag{2.13}$$

$$D_{cauchy}(x, y) = \frac{3}{2} log(x^2 + y^2) - log(y) \tag{2.14}$$

$$\frac{d}{dx} D_{cauchy}(x, y) = \frac{3x}{(x^2 + y^2)} \tag{2.15}$$

$$\frac{d^2}{dx^2} D_{cauchy}(x, y) = \frac{3(-x^2 + y^2)}{(x^2 + y^2)^2} \tag{2.16}$$

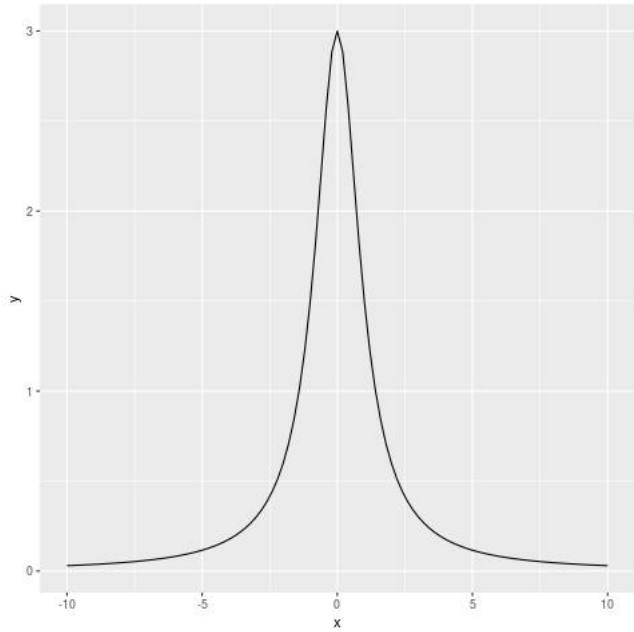


Figure 2.9: The second derivative of the Cauchy distance ($d(y|x)$, see Equation 2.16) with the assumption that y=1.

Gaussian distribution

The Gaussian distribution is another case of the SαS distributions (see Section 2.2.4). This distribution can be used to model the speech signal and to remove reverberation from it [26]. However, this is the case for short segments of speech ($<$ 2.5 ms). When the speech segments are smaller than 5 ms, they can be modelled by a multivariate Gaussian distribution [27]. The modelling of the speech works in the time domain and can also be used to model noise. Making it useful for developing new algorithms, but the variance of the Gaussian distribution needs to be chosen carefully to be able to model the speech. Therefore, both Gazor et al. [27] and Usman et al. [28] recommend using the Laplacian distribution to model the speech signals. This distribution gives a better prediction of the speech signal and its STFT coefficients.

2.3 Dereverberation

2.3.1 Room impulse Response

When a sound signal travels from the speaker to the listener, it interacts with the environment. In an enclosed space, for example an office or a workshop, the signal

bounces off the walls and the ceiling. This interaction is described by the room impulse response (RIR) in the time domain and the frequency response function (FRF) in the frequency domain. The resulting speech (y in the time domain or Y in the frequency domain) is described as the interaction between these functions and the clean speech (x in the time domain or X in the frequency domain). In the time domain this interaction is described by convolution (see Equation 2.17) and in the frequency domain this is described by multiplication (see Equation 2.18).

$$y(t) = x(t) \circledast RIR(t) \tag{2.17}$$

$$Y(f) = X(f)FRF(f) \tag{2.18}$$

Image method

The image method is a technique for calculating the room impulse response. It models a speaker in a room as a point source in a rectangular cavity. The method described by Allen et al. [29] starts off by assuming the speaker is in free space (i.e. no walls present) where the pressure wave emitted by the speaker is of the form described by Equation 2.19.

$$P(\omega, L_s, L_m) = \frac{exp(i\omega(R/v - t))}{4\pi R} \tag{2.19}$$

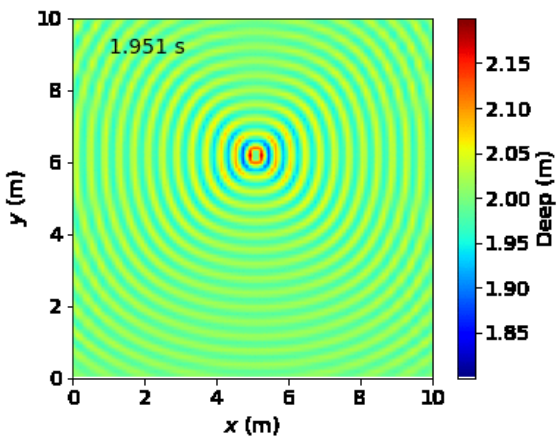


Figure 2.10: A single source in an 2D environment without walls as described by Equation 2.19. In this case the sound is not reflected by any surface.

When a rigid wall (i.e. a wall with zero normal velocity) is present, the boundary condition given by this wall is represented by placing an image symmetrically on the far side of the wall. The image in this case represents the reflection produced by the wall. This image varies by structure and surface of the wall. To include the image in the equation two distances are defined, one from the microphone to the source (R_{sou}) and one to the image (R_{img}). This expands Equation 2.19 to 2.20 and assumes that the wall is placed at x = 0 and where X represents the location of the source (x, y, z) and X' represents the microphone location (x', y', z').

$$
\begin{aligned}
P(\omega, X, X') &= \left[\frac{exp(i(\omega/v)R_{img})}{4\pi R_{img}} + \frac{exp(i(\omega/v)R_{sou})}{4\pi R_{sou}}\right] \\
R_{img}^2 &= (x+x')^2 + (y+y')^2 + (z+z')^2 \\
R_{sou}^2 &= (x-x')^2 + (y+y')^2 + (z+z')^2
\end{aligned}
\tag{2.20}
$$

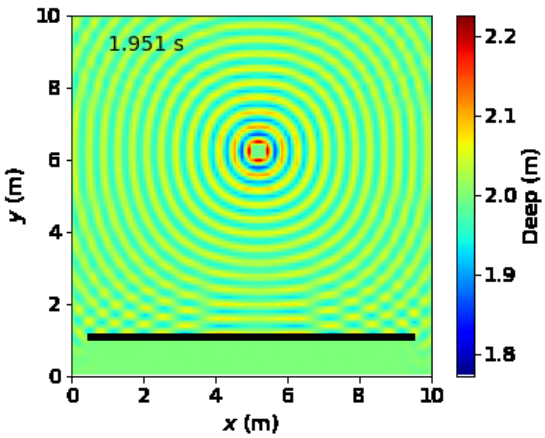


Figure 2.11: A single source in an 2D environment with one wall. The sound is reflected back from the wall towards the source as described by Equation 2.20. These reflections show up as waves moving back to the sound source.

Expanding this to 6 walls (representing a boxed room) increases the complexity because each image in itself is imaged to account for the reflections bouncing off the other walls (see Equation 2.21). In Equation 2.21, the R_p vector represents the eight permutation vectors and R_r the room dimensions and the influence of the dimensions (in the form of the integer variables n, l, m).

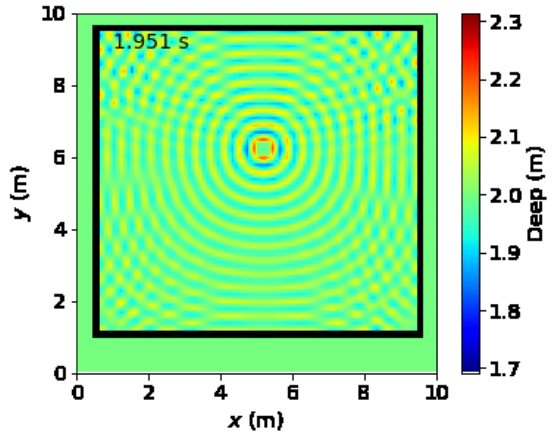


Figure 2.12: A single source in an 2D environment with four walls. The sound is reflecting back from the four walls towards the source (see Equation 2.21). These reflections show up as waves moving back to the sound source.

$$
\begin{aligned}
P(\omega, X, X') &= \sum_{p=1}^{8} \sum_{r=-\infty}^{\infty} \frac{exp(i(\omega/v)|R_p + R_r|)}{4\pi|R_p + R_r|} exp(-i\omega t) \\
R_p &= (x \pm x', y \pm y', z \pm z') \\
R_r &= 2(nL_x, lL_y, mL_z)
\end{aligned}
\tag{2.21}
$$

When the Fourier transform is applied to Equation 2.21, the room impulse response is found (see Equation 2.22).

$$
p(t, X, X') = \sum_{p=1}^{8} \sum_{r=-\infty}^{\infty} \frac{\delta(t - (|R_p + R_r|/v))}{4\pi|R_p + R_r|} \tag{2.22}
$$

The method for rigid walls is expanded to non-rigid walls by assuming the approximate point image model (see Equation 2.22) and an angle independent wall reflection coefficient β. These assumptions expand Equation 2.22 to include the effects of angle independent wall absorption (see Equation 2.23).

$$P(t, X, X') = \sum_{p=1}^{8} \sum_{r=-\infty}^{\infty} \beta_{x_1}^{|n-q|} \beta_{x_2}^{|n|} \beta_{y_1}^{|l-j|} \beta_{y_2}^{|l|} \beta_{z_1}^{|m-k|} \beta_{z_2}^{|m|} \times \frac{\delta(t - (|R_p + R_r|/v))}{4\pi |R_p + R_r|} \tag{2.23}$$

$$R_p = (x - x' + 2qx', y - y' + 2jy, z - z' + 2kz)$$

β represents the pressure reflection coefficients, subscript 1 refers to the adjacent walls and subscript 2 to the opposing wall (where j, k, l, m, n, q describe the influence of the walls). The reflection coefficient β can be calculated using the Sabine energy absorption coefficient α (see Equation 2.24).

$$\begin{aligned} RT_{60} &= 0.161V/S\bar{\alpha} \\ \alpha &= 1 - \beta^2 \end{aligned} \tag{2.24}$$

Reverberation through ray-tracing

Another more computationally expensive method for calculating the reverberation is ray-tracing. This method is currently used in the games industry and has an advantage in that it includes the influence of late reflections. Game engines such as Unity and rendering programmes such as Blender use ray-tracing for the calculation of the late reflections. An example of this is iSound [30] which uses specular reflections and edge diffraction to calculate the contribution paths of each ray emitted from the sound source. The hybrid acoustic model [31] combines the image method and a ray tracer which uses three different transforms (Finite difference time domain, beam tracing and acoustic radiance transfer) to calculate the reverberation. The finite difference time domain method is used for the low frequency modelling, beam tracing for low-order reflections and acoustic radiance transfer for late reflections.

2.3.2 Correlation-based dereverberation method

Reverberant speech is the result of clear speech coming from the speaker's mouth and reflections coming from the environment (see Section 2.3.1). The frequency response function (FRF) describes this interaction in the frequency domain and can be approximated by calculating the cross-correlation between the nonreverberant signal (Y) and the reverberant signal (X). Using the correlation between microphones is a computationally inexpensive way of determining the dereverberant signal. This means that can be easily run in real-time. The trade-off is that the technique is not adaptable, meaning the calculation needs to be correct the first time it runs and does not adapt when a speaker is moving. Instead, recalculation is needed to build a new mask. However, this one-shot approach is not as accurate as learning methods that are able to use a delayed signal.

There are three ways of calculating the cross-correlation between the nonreverberant and the reverberant signals. Two of these (H_1 and H_2) [32] assume that there is noise in either the reverberant (in the case of H_1, see Equation 2.26) or nonreverberant signal (in the case of H_2, see Equation 2.27). The H_1 algorithm will give an underestimation of the dereverberation mask when there is noise on the nonreverberant signal, whereas H_2 will give an overestimation when there is noise on the reverberant signal.

$$\begin{aligned} G_{xx} &= XX^H \\ G_{xy} &= XY^H \\ G_{yx} &= YX^H \\ G_{yy} &= YY^H \end{aligned} \tag{2.25}$$

$$H_1 = G_{xy}{G_{yy}}^{-1} \tag{2.26}$$

$$H_2 = G_{xx}{G_{yx}}^{-1} \tag{2.27}$$

The H_s algorithm (see Equation 2.28) uses a positive scaling factor s for balancing the reverberant and nonreverberant signals and accounting for noise present in the two signals.

$$H_s = \frac{s^2 G_{xx} - G_{yy} + \sqrt{(s^2 G_{xx} - G_{yy})^2 + 4s^2 |G_{yx}|^2}}{2s^2 G_{yx}} \tag{2.28}$$

A method for calculating H_s is described by Leclère et al. [32] and is computationally more expensive because of calculating the eigenvalues and using singular-value decomposition to build the mask. However, this technique accounts for noise in both the input and output signals, resulting in a signal that has less distortion than the results coming from the other two correlation techniques.

Leclère et al. [32] use the cross-correlation of the two signals and the auto-correlation of each to build a cross-correlation matrix (see Equations 2.25 and 2.29).

$$G_{xyxy} = \begin{pmatrix} G_{xx} & G_{xy} \\ G_{yx} & G_{yy} \end{pmatrix} \tag{2.29}$$

Taking the eigenvalue-decomposition of this matrix (G_{xyxy} in Equation 2.29) and discarding the smallest eigenvalues (λ_M) but keeping the eigenvectors (U and V), builds a mask that removes the reverberation from the original signal (see Equation 2.30). The λ smallest eigenvalues describe noise within the signal and therefore can be omitted. In addition to this, two scaling factors (s_x and s_y) are introduced that allows the algorithm to account for noise in both the nonreverberant and reverberant signal (see Equation 2.32).

$$G_{xyxy} = \begin{bmatrix} U_x & V_x \\ U_y & V_y \end{bmatrix} \begin{bmatrix} \lambda_N & 0 \\ 0 & \lambda_M \end{bmatrix} \begin{bmatrix} U_x & V_x \\ U_y & V_y \end{bmatrix}^H \tag{2.30}$$

$$G_{xyxy} = \begin{bmatrix} U \\ V \end{bmatrix} CD^H \tag{2.31}$$

$$H_s = \frac{1}{s_x} U_n {V_n}^{-1} s_y \tag{2.32}$$

The eigenvalue decomposition is by definition defined for square matrices. How-

ever, the input to the algorithm is not a square matrix. This means using the singular value decomposition (SVD) (see Equation 2.31) of the algorithm. SVD produces four matrices of which U and V are the left singular vectors, C contains the singular values and D^H the right singular vectors. The H_s filter uses the left singular matrices with the first n singular values (see Equation 2.32). The amount of singular values used is determined empirically.

Is is assumed that the singular values in matrix C are in decreasing order, where the first value describes the biggest contribution to the data and the last value the smallest. With this assumption the data can be reconstructed using a percentage of the singular values and by discarding the smallest singular values remove the reverberation. The dereverberant signal X_{approx} is subsequently approximated by multiplying the affected signal Y with the inverse of the approximated reverberation H_s (see Equation 2.33).

$$X_{approx} = Y {H_s}^{-1} \tag{2.33}$$

2.3.3 Weighted prediction error

Weighted prediction error (WPE) [26] is a multichannel technique that assumes that the speech signal is nonstationary and has short time Gaussianity. Each short time frame can be modelled by a stationary univariate Gaussian process with zero mean and covariance matrix $R_t = E_{s1,st}$. Similarly, the desired signal is also a Gaussian process. This gives a marginal pdf of d_t as $p(dt) = N(d_t; 0, \sigma^2)$ where σ^2 is $E|d_t|^2$. The σ^2 describes the time-varying variance which is stationary for a short time frame and varies over different frames, meaning that σ^2 is the varying average of a time frame.

The second assumption WPE makes is that there is only one speaker and limited background noise. Under this assumption the dereverberant signal is calculated from the prediction errors of one of the M-channels, meaning that only one channel has to be estimated instead of all channels.

To find the desired signal, WPE employs a log likelihood function where the goal is to find the optimal set of parameters that maximise the log likelihood equation. These parameters are found by calculating the covariance matrix of the delayed input

signal (Φ) and multiplying this with the covariance vector of the delayed and original input signal (ϕ), this results in $\bar{c}$ (see Equation 2.34). The optimal parameters are then found by alternately updating $\bar{c}$ and σ^2. The technique calculates the regression vector that created the resulting signal. In doing this, the technique also approximates the nonreverberant signal which multiplied with the regression vector gives the reverberant signal.

$$\begin{aligned} \sigma^2 &= \frac{1}{T}\sum^{T}|d_t|^2 \\ \Phi &= \frac{x_{t-D}x_{t-D}^T}{\sigma^2} \\ \phi &= \frac{(x_{t-D}x_t^{(1)})^T}{\sigma^2} \\ \bar{c}_t &= \Phi\phi \end{aligned} \tag{2.34}$$

2.3.4 Multiple Input Multiple Output WPE

Multiple input multiple output (MIMO) WPE [33] is a version of WPE which produces the same number of outgoing channels as incoming channels and is based on the principle of WPE. However, it changes the assumption of having only one source in the room to multiple speakers. MIMO WPE preserves the time-difference of arrival (TDoA) of the sources and microphones making it suitable for speaker localisation and tracking. This means that the data can be passed through this algorithm and then be used for speaker tracking or separation to improve the accuracy of speech recognisers. The algorithm uses the Hadamard-Fischer (HF) mutual correlation which is applied to multivariate random variables. The HF mutual correlation assumes that $U_1, \cdots, U_n$ are complex-valued multivariate random vectors and U is the vector in which these are stacked as $[U_1^T, \cdots, U_N^T]^T$. This correlation between these vectors is described in Equation 2.35.

$$C_{HF}(U_1, \cdots, U_N) = \frac{1}{N}(\sum_{n=1}^{N} log(detE(U_n{U_n}^T)) - log(detE(UU^T))) \tag{2.35}$$

The method works under the assumption that a positive or zero number means that all of the multivariate random vectors are mutually uncorrelated. This correlation method is used in WPE to determine the dereverberated signal.

The main difference between WPE and MIMO WPE is the calculation of the spatial correlation matrix and the usage of the variation. While WPE uses the variation to calculate the sample correlation matrix (Φ) and the sample correlation vector (ϕ), MIMO WPE uses the spatial correlation matrix (Δ). This ensures that MIMO WPE can use the information from all microphones instead of one, creating a more robust approximation of the dereverberant signal. In addition to this, using a spatial correlation matrix allows the algorithm to calculate the dereverberant signal for each microphone and produce a multichannel output.

$$\begin{aligned} \Delta &= E(d_t d_t^{+}) \\ \Phi &= x_{t-D} \Delta {x_{t-D}}^T \\ \phi &= x_{t-D} \Delta {x_t}^T \\ \bar{c}_t &= \Phi \phi \end{aligned} \tag{2.36}$$

2.4 Speaker separation

2.4.1 Ideal binary filter

Using an ideal binary filter (IBF) or ideal binary mask for speaker separation assumes that there are two sources of speech which can be combined at different volumes (measured in dB). The IBF assumes that both sources are known (i.e. a ground truth of each individual source is available) and describes the ideal case of separation (by subtracting one ground truth from the mixture). The algorithm works by building a binary mask where the number 1 means that this time-frequency (t-f) bin is used and 0 means that this t-f bin is ignored (see Equation 2.37, where X is the mixture, S one of the two sources and LC a local criterion measured in dB). Knowing the second source allows this mask to be tailored to remove this source from the mixture. The IBF is used to see how well the different sources can be separated and to create

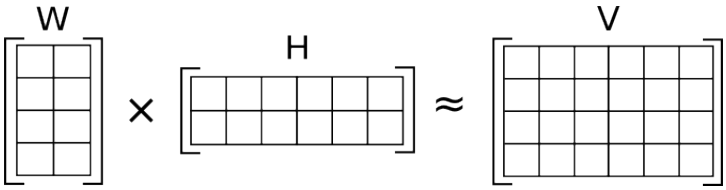


Figure 2.13: Nonnegative matrix factorisation [35]

an upper bound for the algorithms.

$$IBM(t,f) = \begin{cases} 1 \text{ if X(t,f) - S(t,f)} > \text{LC}, \\ 0 \text{ otherwise.} \end{cases} \tag{2.37}$$

2.4.2 Non-negative matrix factorisation

Non-negative matrix factorisation (NMF) [34] is an unsupervised clustering technique that tries to approximate its input by multiplying two randomly initialised matrices together (see Figure 2.13). These two matrices are updated every iteration to create a new approximation. By multiplying columns and rows of the two matrices, a mask is approximated for filtering out sources in the input using a Wiener filter (see Equation 2.38, where V_s and X_s are the source s of the approximation and original mixture respectively). With NMF, the cost function influences the update rules and how fast the algorithm converges. In addition to the cost function NMF can be expanded with sparsity, convolution and direction of arrival.

$$X_s = \frac{V_s}{V} X \tag{2.38}$$

The input to an NMF algorithm needs to be non-negative for the technique to work. This technique is applied to speech data and in particular, the short-time Fourier transform (STFT) of the speech data. The squared magnitude of the STFT applied to the raw speech data ($V = |X|^2$) is used as input to the technique (the size of V is $F \times T$).

$$V \approx \widetilde{V} = WH \tag{2.39}$$

NMF multiplies two matrices (W and H) together to approximate the input (see Equation 2.39). The size of W is $F \times K$, where K is the number of sources to extract

from the input and the size of H is $K \times T$. Multiplying W and H together produces $\widetilde{V}$. The W matrix is seen as a feature matrix for each source and the H matrix is seen as the activation matrix for each source. This means that multiplying one column of W and one row of H results in the activation of the features for one source and thus the approximation of the one source ($\widetilde{V}_s$). To get a good approximation, NMF is run for a number of iterations. After each iteration the difference between the input and the approximation is calculated by the cost function (see Equation 2.40) and the updates are modified accordingly until convergence of the cost function is reached.

$$D(V||\widetilde{V}) = \sum_{f=1}^{F}\sum_{n=1}^{N} d([V]_{fn}|[\widetilde{V}]_{fn}) \tag{2.40}$$

$$W = W\frac{((WH)^{\beta-2} \times V)H^T}{(WH)^{\beta-1}H^T} \tag{2.41}$$

$$H = H\frac{W^T((WH)^{\beta-2} \times V)}{W^T(WH)^{\beta-1}} \tag{2.42}$$

The β parameter in Equations 2.41 and 2.42 defines which cost function from the β divergence is used and adapts the update rules accordingly. When $\beta = 2$ the squared Euclidean distance is used, if $\beta = 1$ KL divergence and if $\beta = 0$, the IS divergence is used.

In addition to the cost functions, four different versions of NMF are described here: sparse, convolution, direction of arrival (DoA) and time-difference of arrival (TDoA). Note that, the latter two are multichannel techniques meaning that they are only usable when working with multiple microphones and require the distance between microphones.

Sparse NMF

The sparse NMF [36, 37] adds a parameter λ to the update rule for the H matrix - this ensures that features are not lost by multiplying by 0 for the activation (see Equation 2.43). With sparsity, the W and H matrices are overcomplete, meaning that the dimensionality of the factorisation space is bigger than the effective dimensionality of the input space. This results in fewer items in the W and H matrices

being needed to represent the separated signals. However, it takes longer to reach this overcompletion by the nature of the signals. Sparsity is only enforced when $\lambda > 0$.

$$H = H\frac{W^T((WH)^{\beta-2} \times V)}{W^T(WH)^{\beta-1} + \lambda} \tag{2.43}$$

Convolution NMF

For convolution [38], $\widetilde{V}$ is calculated differently, each feature W is multiplied with a shifted version of H by t timesteps (see Equation 2.44). Similarly, it changes the update rules for updating the W and H matrices (see Equation 2.45 and 2.46). Convolution is better at dealing with noise because it averages over different timesteps meaning that some of the noise will be removed because it is not present at all the timesteps. Similarly, it will be able to deal with overlapping speech because by taking an average there is a higher chance of including timesteps where there is only one speaker present.

$$\widetilde{V} = \sum_{t=0}^{T} W_t \overrightarrow{H}^{\mathrm{t}} \tag{2.44}$$

$$W = W\frac{\overrightarrow{H}^{\mathrm{t}}[\frac{V}{(WH)^{\beta-2}}]H^T}{(WH)^{\beta-1}H^T} \tag{2.45}$$

$$H = H\frac{W^T\overrightarrow{H}^{\mathrm{t}}[\frac{V}{(WH)^{\beta-2}}]}{W^T(WH)^{\beta-1} + \lambda} \tag{2.46}$$

Direction of arrival NMF

Direction of arrival is combined with NMF into a version that uses the difference of angle of arrival at which the sound arrives at the microphones. The research by Stein [7] describes the usage of three microphones. In the case of NMF, the direction of arrival (D_W) is seen as extra features and is multiplied each time with W (see Equation 2.48). Whereas for the NTF version, the matrix D is approximated using the least mean squares method of the location and angles of the microphones (see Equation 2.47).

$$V \approx \widetilde{V} = DWH \tag{2.47}$$

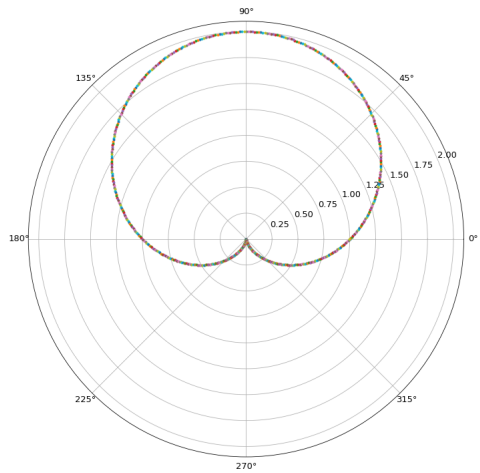


Figure 2.14: Example of the polar plot of a directional microphone showing the angles where the microphone can receive the signal from. The best response is given by a source in front of the microphone (90 degrees) and the worst response by a source behind the microphone (270 degrees). These angles are used to determine the look directions for Equation 2.50

$$W = D_W W \frac{((WH)^{\beta-2}V)H^T}{(WH)^{\beta-1}H^T} \tag{2.48}$$

Information in the form of a spatial covariance matrix, which is said to describe the time-difference of arrival (TDoA), can also be added to the NMF algorithm. However, this makes NMF more complex than when adding the direction of arrival (DoA) information.

Time-difference of arrival NTF

The spatial covariance matrix (SCM) contains the phase difference for all the coordinates, in a 2D plane, that are within the field of view of the microphone. This field of view is described as a polar pattern that shows under which angle, between source and microphone, the microphone still can receive sound from the source (see Figure 2.14). Each of the coordinates within the SCM is described by a specific azimuth and elevation and shows the mixing of signals by phase and magnitude differences. The specific combination of azimuth and elevation describes a look direction.

This information is stored in a spatial covariance matrix (A) which describes the mixing of signals by phase and magnitude differences and is multiplied with

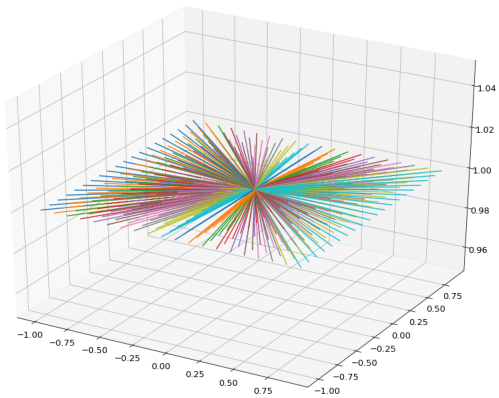


Figure 2.15: Example of the look directions (τ) showing the angles under which the time difference of arrival (k_o) is calculated (see Equation 2.50). The different colours represent the individual look directions.

the estimation of the sources (see Equation 2.49). The size of the matrix A is $O \times F \times M \times M$.

$$V \approx S\widetilde{V} = AWH \tag{2.49}$$

Nikunen et al. [39] use the TDoA to calculate the SCM first, the look directions (τ) between microphone pairs are calculated. First the TDoA for each look direction (k_o) and each microphone (τ_n) is calculated (see Equation 2.50) with respect to the array centre point (c) using the speed of sound (v). From this, it is possible to calculate the TDoA between microphone pairs (τ_{np}) with the same look direction (see Equation 2.51).

$$\tau_n(k_o) = \frac{-k_o^T(n-c)}{v} = \frac{-k_o^T n}{v} \tag{2.50}$$

$$\tau_{np}(k_o) = \tau_n(k_o) - \tau_p(k_o) \tag{2.51}$$

With knowing the TDoA for each microphone pair, the phase difference per frequency can be calculated (see Equation 2.52). For this it is important to know the sample frequency and the number of STFT bins.

$$[A_{o,f}]_{np} = exp(i2\pi f_i \tau_{np}(k_o))$$
$$f_i = (i-1)F_s/F \tag{2.52}$$

This now gives an approximation of the location of the speaker which can be used to approximate the mixture. This is combined with a direction weight (Q) of the size K × O to cluster the NMF components (see Equation 2.55). N.B. $X_{f,n}$ is determined by Equations 2.53 and 2.54 to create a non-negative input to the algorithm.

$$\hat{x}_{f,n} = [|x_{f,n,1}|^{1/2} sign(x_{f,n,1}), \ldots, |x_{f,n,M}|^{1/2} sign(x_{f,n,M})]^T \tag{2.53}$$

$$X_{f,n} = \hat{x}_{f,n} \hat{x}_{f,n}{}^H \tag{2.54}$$

$$X_{f,n} \approx V_{f,n} = \sum_{k=1}^{K} \sum_{o=1}^{O} A_{o,f} q_{k,o} w_{f,k} h_{k,n} \tag{2.55}$$

After the first approximation of the SCM (A), it will be updated to approximate the mixture better. For this first the error between the approximation and the original mixture is used (see Equation 2.56). This combined with the directional weight, the feature matrix and the approximation of the mixture to create an update for the SCM (see Equation 2.57).

$$\mathbf{E}_{f,n} = X_{f,n} - V_{f,n} \tag{2.56}$$

$$\hat{A}_{o,f} \leftarrow A_{o,f} \left[\sum_{n,k} q_{k,o} w_{f,k} v_{f,n} + \sum_{n,k} q_{k,o} w_{f,k} \mathbf{E}_{f,n} \right] \tag{2.57}$$

Within the calculation of the SCM the singular value decomposition (SVD) is used to determine the eigenvalues of the SCM and to remove any negative eigenvalues (see Equation 2.58). After the eigenvalues are determined and the phase of the signal is added to the SCM (see Equation 2.59), it needs to be renormalised by dividing it by the Frobenius norm of the updated matrix (see Equation 2.60).

$$\hat{A}_{o,f} \leftarrow V\hat{D}V^H \tag{2.58}$$

$$\hat{A}_{o,f} \leftarrow |\hat{A}_{o,f}| \exp(i \arg(A_{f,n})) \tag{2.59}$$

$$A_{o,f} \leftarrow \frac{\hat{A}_{o,f}}{||\hat{A}_{o,f}||_F} \tag{2.60}$$

For updating the Q, W, and H matrices (see Equations 2.61 to 2.63), the trace between the SCM and error is used. This error is the difference between the original mixture and the approximation at each time step and each frequency. This is not the result of the cost function, which is a single number. The trace is combined with the other matrices to build a new approximation, depending on the matrix being updated two of the following Q, W or H as well as the approximation are used. For example, for updating the Q matrix the trace is multiplied with the W and H matrices and divided by the multiplication of the W, H and V matrices (see Equation 2.61). This makes the new approximation of Q and later on the new approximation of the mixture closer to the original mixture. However, the result of this multiplication is in the range of -1 to 1, which means that certain items result in being negative. This is not allowed in NMF therefore 1 is added to the result of the division to make sure that all items stay positive.

$$q_{k,o} \leftarrow q_{k,o}\left(1 + \frac{\sum_{f,n} w_{f,k}h_{k,n}tr(\mathbf{E}_{f,n}A_{o,f})}{\sum_{f,n} w_{f,k}h_{k,n}v_{f,n}}\right) \tag{2.61}$$

$$w_{f,k} \leftarrow w_{f,k}\left(1 + \frac{\sum_{n,o} q_{k,o}h_{k,n}tr(\mathbf{E}_{f,n}A_{o,f})}{\sum_{n,j} q_{k,o}h_{k,n}v_{f,n}}\right) \tag{2.62}$$

$$h_{k,n} \leftarrow h_{k,n}\left(1 + \frac{\sum_{f,o} q_{k,o}w_{f,k}tr(\mathbf{E}_{f,n}A_{o,f})}{\sum_{f,s} q_{k,o}w_{f,k}v_{f,n}}\right) \tag{2.63}$$

These three matrices need to be normalised using first the square root of the sum of the activation to normalise W and H (see Equation 2.64). Followed by the square root of the direction weights which normalises Q and W (see Equation 2.65). The latter matrix is normalised twice because it has an effect on both matrices (Q

and H). The normalisation ensures the the respective matrices and the resulting approximation is in the same range as the original mixture.

$$\begin{aligned} \hat{a}_k &= \left(\sum_{k=1}^{K} h_{k,n}^2 \right)^{1/2} \\ h_{k,n} &\leftarrow \frac{h_{k,n}}{\hat{a}_k} \\ w_{f,k} &\leftarrow w_{f,k} \hat{a}_k \end{aligned} \tag{2.64}$$

$$\begin{aligned} \hat{b}_k &= \left(\sum_{o=1}^{O} q_{k,o}^2 \right)^{1/2} \\ q_{k,o} &\leftarrow \frac{q_{k,o}}{\hat{b}_k} \\ w_{f,k} &\leftarrow w_{f,k} \hat{b}_k \end{aligned} \tag{2.65}$$

To separate the sources in this algorithm a clustering algorithm (k-means) is employed. Its outcome (B) is multiplied with the Q, W and H matrices to form the separated sources (see Equation 2.66). The number of sources is equal to the number of cluster that the clustering algorithm uses. This algorithm gives the matrix b which represents to which cluster the spatial weight belong. In this case $\mathbf{x}$ is the mixture that has not been modified for the input meaning that is not modified by Equations 2.53 and 2.54.

$$y_{s,f,n} = x_{f,n} \frac{\sum_{k,o} b_{s,k} q_{k,o} w_{f,k} h_{k,n}}{\sum_{s,k,o} b_{s,k} q_{k,o} w_{f,k} h_{k,n}} \tag{2.66}$$

Covariance NTF

This is however not the only way to calculate the SCM. A different way is to use the cross-correlation between microphones [38, 40]. This gives the location of the different speakers and also takes into account the reverberation coming from different parts of the room. The location is determined by the highest values in the

covariance whereas the lower values represent the the reverberation coming from the room. Using a correlation technique to determine the SCM and the FRF, this method is similar to the WPE method discussed in Section 2.3.3. It accounts for the reverberation of the environment making the resulting signal free from reverberation. This should make it easier for the speech recogniser to determine when a speaker finishes and what the speaker has said.

The calculation of the spectral covariance and the spatial covariance matrices are dependent on each other. First the algorithm creates a non-negative version of the input (see Equation 2.67), this is done instead of the magnitude power spectrum which removes the phase information of the signal (i.e. removes the imaginary component and thus removing the ability to recover the phase information). This non-negative version of the input is used in determining the spatial covariance matrix.

$$\hat{\Sigma}_x = xx^H \tag{2.67}$$

For the spectral covariance matrix, first the mixture needs to be transformed to a matrix of the size M $\times$ S $\times$ F $\times$ N (see Equation 2.68). Now there is a mixture present per source and per microphone to use for the spectral covariance matrix where the speakers have a different spectral response. This makes it easier to recognise the dominant (or loudest) features in the spectrum which should correspond to the dominant speaker for that microphone.

$$\Sigma_s = diag\Big([\underbrace{v_{1,fn}, \ldots, v_{1,fn}}_{\text{M times}}, \underbrace{v_{2,fn}, \ldots, v_{2,fn}}_{\text{M times}}, \ldots, \underbrace{v_{J,fn}, \ldots, v_{S,fn}}_{\text{M times}}]\Big) \tag{2.68}$$

The next step is to determine the complex approximate mixture which takes into account the spectral covariance matrix (A) (see Equation 2.69). This equation also takes into account the noise that is present in the room (in the form of matrix Σ_n).

$$\Sigma_x = A\Sigma_s A_H + \Sigma_n \tag{2.69}$$

$$\Sigma_n = diag(\Sigma_x - A\Sigma_{xs}{}^H - \Sigma_{xs}A^H + A\Sigma_s A^H) \tag{2.70}$$

To determine the relative error between the complex approximation and the mixture, the algorithm takes into account the spatial covariance matrix (see Equation 2.71).

$$\Omega_s = \Sigma_s A^H \Sigma_x^{-1} \tag{2.71}$$

The last step in this part of the process; the spectral covariance matrix is calculated by using the relative error, the non-negative version of the input, the spatial covariance matrix and the microphone matrix (see Equation 2.72). The latter ensures that the information is available per microphone which means it is easier to determine the dominant source for a microphone.

$$\hat{\Sigma}_s = \Omega_s \hat{\Sigma}_x \Omega_s{}^H + (I - \Omega_s A)\Sigma_s \tag{2.72}$$

For the spatial covariance matrix, it is needed to determine the cross-correlation between the original non-negative input and the relative error of the complex approximation (see Equation 2.73). This cross-correlation takes in to account the noise that is present in the relative error and shows the strongly correlated components between the two which can be used to determine the spatial covariance matrix. To do this, the cross-correlation is divided by the spectral covariance matrix (see Equation 2.74).

$$\hat{\Sigma}_{xs} = \hat{\Sigma}_x \Omega_s{}^H \tag{2.73}$$

$$A = \frac{\hat{\Sigma}_{xs}}{\hat{\Sigma}_s} \tag{2.74}$$

For the non-negative tensor factorisation part of this algorithm, it is important to determine the approximation. This is done by multiplying three matrices (Q, W and H) together (see Equation 2.75).

$$V = \sum_{k=1}^{K} w_k \circ h_k^T \circ q_k \tag{2.75}$$

All three matrices are dependent on the spatial covariance matrix ($\hat{\Sigma}_s$) which should be non-negative. The spatial covariance matrix is averaged per microphone and the only the information from the diagonal is used (see Equation 2.76). These are the items which the highest correlation with the respective sources and should therefore be non-negative.

$$\xi_{s,f,n} = \frac{1}{M} \sum_{i=(s-1)M+1}^{sM} \hat{\Sigma}_s(i,i) \tag{2.76}$$

To update Q, W and H, its respective matrices are used as well as the average of the spatial covariance matrix and the approximation (see Equations 2.77, 2.78 and 2.79). For example for updating the Q matrix, W, H, ξ and V are multiplied and divided by W, H and V. This means that the updates are relying on the outcome of the spatial covariance matrix to determine which source in stronger in which microphone. Apart from the usage of ξ, the update rule is similar to that of NMF.

$$q_{sk} \leftarrow q_{sk} \left(\frac{\sum_{f,n} w_{f,k} h_{k,n} \xi_{s,f,n} v_{s,f,n}^{-2}}{\sum_{f,n} w_{f,k} h_{k,n} v_{s,f,n}^{-1}} \right) \tag{2.77}$$

$$w_{fk} \leftarrow w_{fk} \left(\frac{\sum_{s,n} h_{k,n} q_{s,k} \xi_{s,f,n} v_{s,f,n}^{-2}}{\sum_{s,n} h_{k,n} q_{j,k} v_{s,f,n}^{-1}} \right) \tag{2.78}$$

$$h_{kn} \leftarrow h_{kn} \left(\frac{\sum_{,f} w_{f,k} q_{j,k} \xi_{s,f,n} v_{j,f,n}^{-2}}{\sum_{j,f} w_{f,k} q_{s,k} v_{s,f,n}^{-1}} \right) \tag{2.79}$$

All four matrices (A, Q, W and H) need to be normalised to ensure that the approximation is in the same range as the original mixture (see Equations 2.80, 2.81 and 2.82). First the spatial covariance matrix is normalised and with it also W has its first normalisation with respect to the SCM (see Equation 2.80).

$$
\begin{aligned}
A &\leftarrow \frac{A}{sign(A)} \\
\hat{a} &= \left(\sum_{m=1}^{M} |A|^2 \right) \\
A &\leftarrow \frac{A}{\hat{a}} \\
W &\leftarrow W\hat{a}
\end{aligned}
\tag{2.80}
$$

This is followed by the normalisation of Q and finally the normalisation of H with respect to W (see Equations 2.80 and 2.81 and 2.82). The W matrix is normalised 3 times to make sure that in the end both W and H are in the same range and these two matrices have the largest influence on the end result.

$$
\begin{aligned}
\hat{b}_k &= \left(\sum_{s=1}^{S} q_{s,k} \right) \\
q_{k,o} &\leftarrow \frac{q_{k,o}}{\hat{b}_k} \\
w_{f,k} &\leftarrow w_{f,k}\hat{b}_k
\end{aligned}
\tag{2.81}
$$

$$
\begin{aligned}
\hat{c}_k &= \left(\sum_{k=1}^{K} w_{f,k} \right) \\
h_{k,n} &\leftarrow h_{k,n}\hat{c}_k{}^T \\
w_{f,k} &\leftarrow w_{f,k}\hat{c}_k
\end{aligned}
\tag{2.82}
$$

As the final step a multichannel Wiener filter is employed to separate the sources from the mixture (see Equation 2.83). For this filter, the SCM is used to account for the locations of the speakers and to create a cleaner separation between them. N.B. it also uses the original mixture that has not been multiplied by a Hermitian transpose nor does it use the magnitude power spectrum of the mixture. The algorithm

applies this separation to each microphone individually therefore it gives a result per microphone per source.

$$
\begin{aligned}
R &= AA^H \\
y_{s,f,n} &= R_{s,f,n}V_{s,f,n}\left[\sum_{s=1}^{S} R_{s,f,n}V_{s,f,n}\right]^{-1}\mathbf{x}_{f,n}
\end{aligned}
\tag{2.83}
$$

2.4.3 Deep Learning

There are three different deep learning techniques used in this thesis, namely the deep neural networks (DNN), recurrent neural networks (RNN) and convolution neural networks (CNN). These three techniques are used as supervised learning techniques. There are also two popular unsupervised learning techniques, namely autoencoder (AE) and generative adversarial network (GAN), described. Their usage is popular for dereverberation but are also used for speaker separation.

Deep neural network

To understand how the three techniques work, it is important to first understand what a neural network is and how it works. A normal single layer perceptron, which is a neural network without any hidden layers, works by multiplying the input with a weights matrix (W) and adding a bias (b) to get an output (see Equation 2.84 and Figure 2.16). A multilayer perceptron (MLP) [41] expands on this by adding a so-called hidden layer (see Equation 2.85 and Figure 2.17). In this case the input will be multiplied with a weights matrix ($W^{(1)}$) and passed through an activation function (G), for example a sigmoid function. The outcome of this will again be multiplied with a weights matrix ($W^{(2)}$) and passed through an activation function (S) to generate the output. A MLP is an example of a neural network and the more recently introduced deep neural network, these two definitions generally assume two or more hidden layers whereas the name MLP is associated with a network with one hidden layer. The MLP can be expanded to a DNN [17] which has a similar structure with more hidden layers (see Figure 2.18).

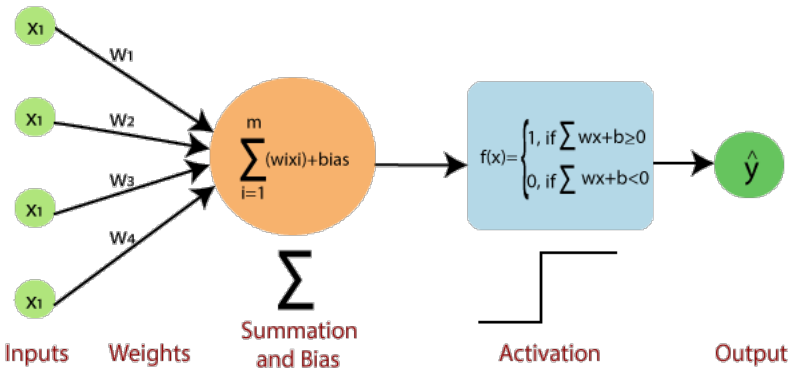


Figure 2.16: A single layer perceptron [42] which is a schematic representation of Equation 2.84

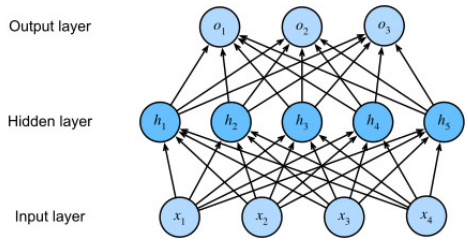


Figure 2.17: A multilayer layer perceptron [43] which is a schematic representation of Equation 2.85

$$f(x) = WX + b \tag{2.84}$$

$$f(x) = S(b^{(2)} + W^{(2)}G(b^{(1)} + W^{(1)}X)) \tag{2.85}$$

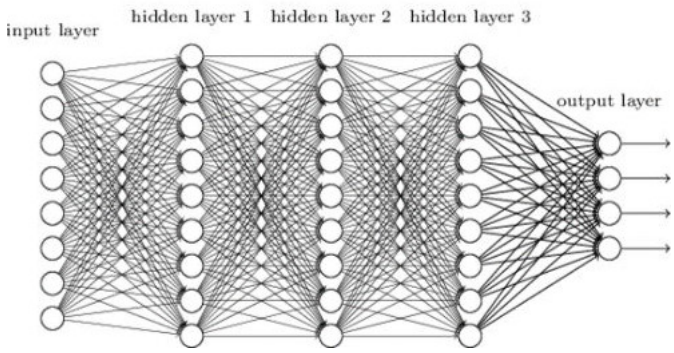


Figure 2.18: A 3 layer deep neural network [44] which expands Equation 2.85

Recurrent neural network

A recurrent neural network (RNN) [45] is a special case of a neural network with loops to allow information to persist (see Figure 2.19). This can be thought of as multiple copies of the same network with the ability to pass information from one copy to the next. The information that is passed through can be the previous output of the network in the case of a Jordan network [46] or the previous output of the hidden layer in the case of an Elman network [47] (see Figure 2.20). Adding information about the previous input to the current input is particularly useful in the case of time-series data, where information from the past helps in making a decision.

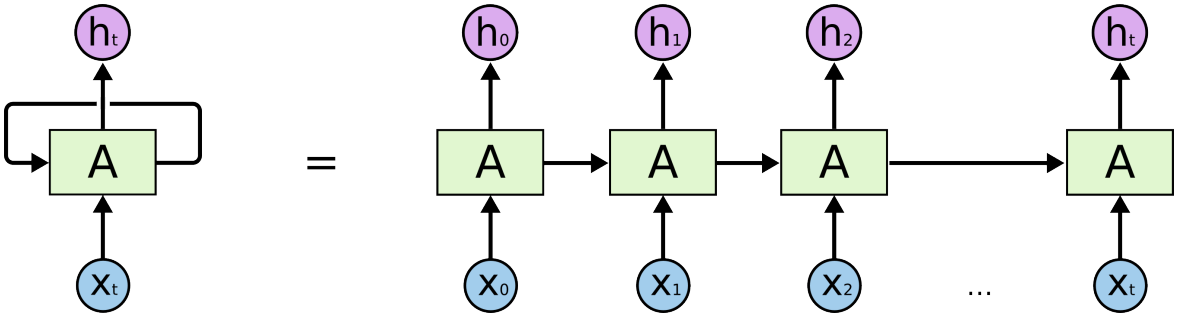


Figure 2.19: An unrolled RNN [48] where the information of the hidden layer (layer A) is being passed to the next cell.

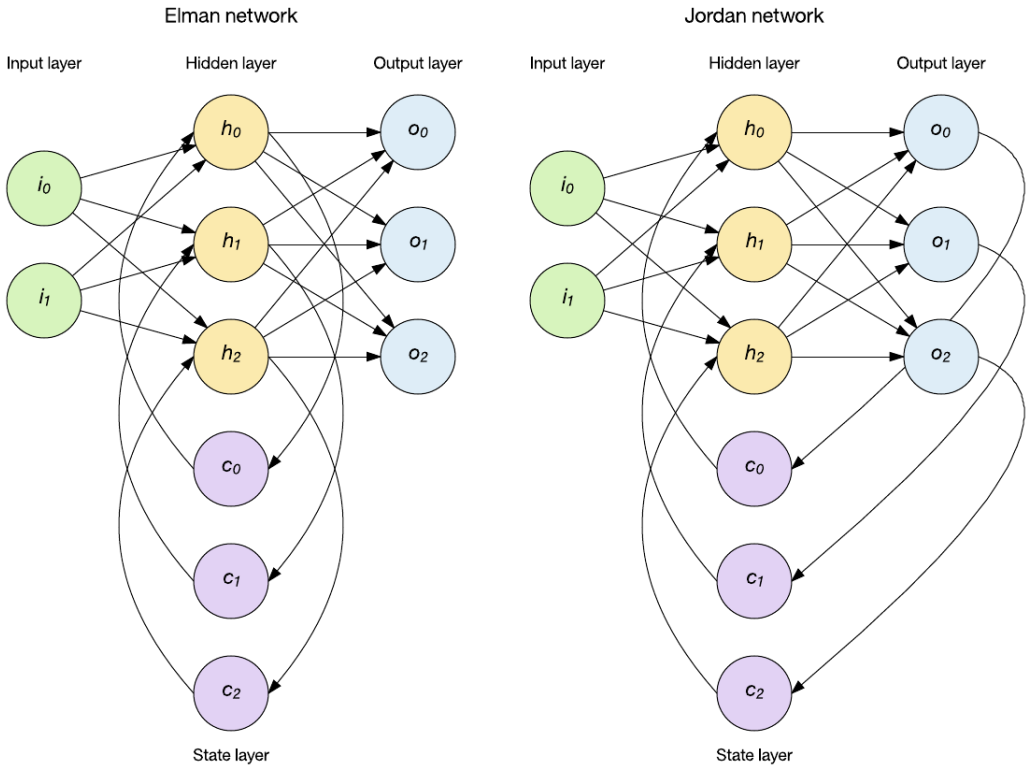


Figure 2.20: An Elman and Jordan networks [49]. N.B. the recurrent connections in an Elman network are between the hidden layer (h) and the context layer (c) whereas in an Jordan network this is between the output layer (o) and the context layer (c).

RNNs have issues with learning long term dependencies, this has to do with the exploding and vanishing gradients problems. The exploding gradients problem refers to the explosion of long term components. This means that there are more long term components than short term ones.

Pascanu et al. [50] describe the exploding gradients problem as a wall in the error surface of a recurrent network, where a regular gradient step would jump this wall and thus disrupt the learning process. Instead, a small-norm step would follow this wall or fall to a (lower error) valley and starts to follow this to a solution.

The vanishing gradients problem describes the opposite where the long term components disappear (i.e. go exponentially fast to norm 0) and the network loses the ability to learn the relation between distant events. Being the opposite of the exploding gradients problem implies that in this case there are more short term components than long term ones. According to Bengio et al. [51] and Pascanu et al. [50] the vanishing gradients problem is an effect of the backpropagation algorithms that is inefficient for learning long term dependencies in the input/output sequence.

For the vanishing gradients problem Bengio et al. [51] assume that a neuron has two attractors $\bar{x} > 0$ and $-\bar{x}$. When this attractor is hyperbolic than the gradients quickly vanish when t increases making it very difficult to continue training as the short term dependencies dominate in the weight gradients. Pascanu et al. [50] suggest that the vanishing gradients problem can be solved with the use of LSTM units because these units have a recurrent connection to itself (fixed to 1) and learn the input and output gates. However, this solution does reduce the problem of exploding gradients but does not solve it. This still exists because it is possible for the gradient connected to the path through the inpout or forget gates to explode due to the self-multiplication of matrices (see Greff et al. [52]). One solution for the exploding gradient problem is gradient thresholding (or clipping [17]) where when the gradient passes a threshold it will be downscaled (see Pascanu et al. [50]) or by introducing a L1 or L2 penalty term on the gradients.

Long short-term memory

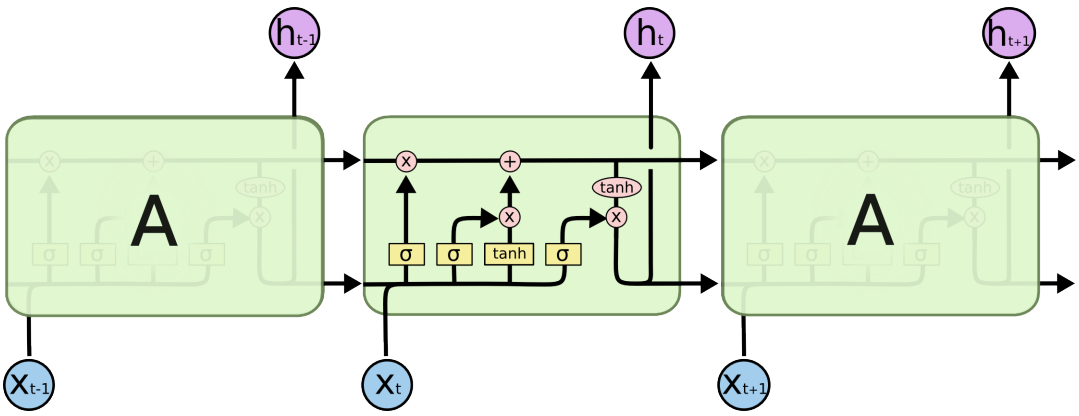


Figure 2.21: The LSTM chain [48] showing the inner workings of an LSTM cell with the three gates. The top line being the repeating cell state to which information can be added. The bottom line being the information passes from the previous cell (containing input X_{t-1}) which is combined with new information for the current cell (the input being X_t)

Long short-term memory (LSTM) networks [53] are capable of learning long-term dependencies and remembering this information is their strength. A LSTM has four internal dense layers in the repeating modules opposed to one (see Figure 2.21). The repeating cell state runs down the chain of LSTM nodes and each node has the ability to add or remove data from the cell state by using three gates (forget, store

and output).

The input gate consists of a dense layer followed by a sigmoid activation function. Firstly non-important information is forgotten - this is determined by the previous cell state (f_t). After this the information from the current input is updated. What information to update the cell state with is determined by the sigmoid activation function in the input gate, 1 means completely update this information and 0 means do not update this information.

The second gate is a forget gate. This decides which to forget from the cell state. Again, the output of a sigmoid function determines what to remove and what to keep, 1 means completely keep this information and 0 means completely forget this information. The output of the sigmoid function is combined with the output of a tanh function. The latter provides a vector of candidate values that could be stored in the cell's memory. The combination of the sigmoid function and the tanh function is the update for the cell's state.

The LSTM has an output gate that decides which information to output and is combined with the output of a different tanh layer which provides the candidate values. The output is based on the current cell's state but is a filtered version of this.

Convolution neural network

The convolution neural network (CNN) [54] contains one or more convolution layers which are followed by neural network layers (see Figure 2.22). Convolution layers combine information seen in the data and try to build a pattern that describes this. It uses the same principle for convolution as described in Section 2.4.2. In addition to this it uses a max or average pooling layer to sum up the result from convolution. However, by its design the max pooling can act as a noise suppressor removing the smaller values coming from convolution.

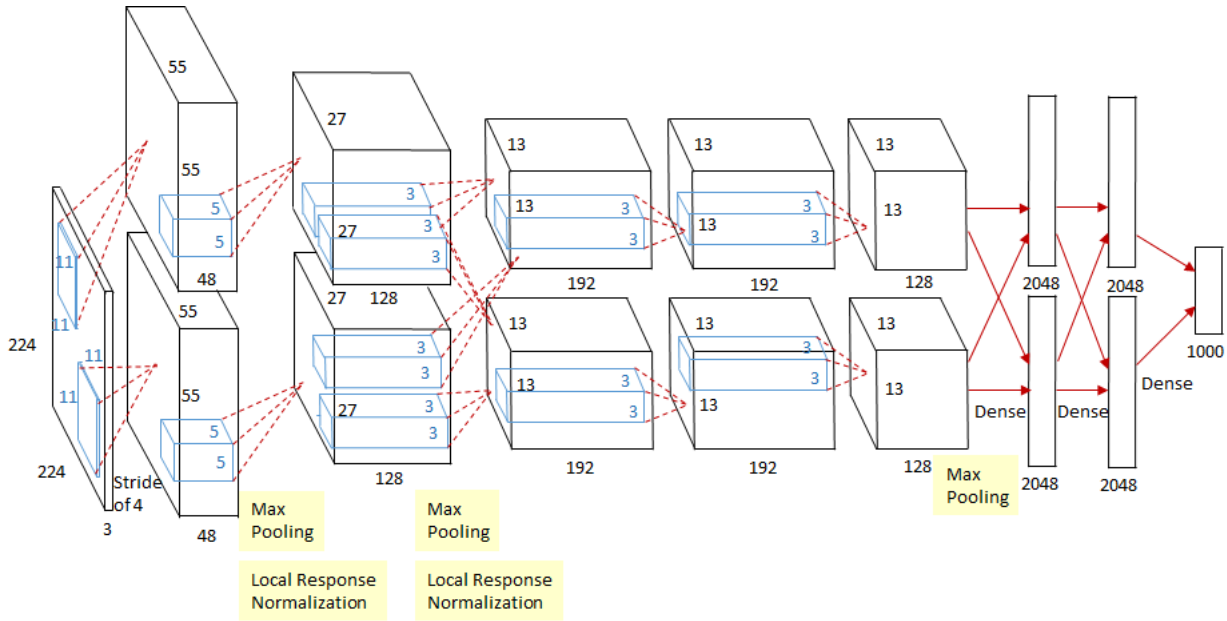


Figure 2.22: The architecture of AlexNet [54] showing the different convolution layers with the size of the kernel (e.g. 11 x 11 in the first layer) forllow by two dense layers to determine which class the image belongs to.

AutoEncoder

AutoEncoders (AE) [55, 56] try to compress the input into fewer dimensions (see Figure 2.23). This is done by stacking the layers of a neural network that becomes smaller in size until the bottleneck layer. After this layer the opposite happens. This means that the input and output of the network are the same and the network learns a compressed (or latent) representation of the input. These autoencoders can be used for denoising a signal where the latent representation learns the important features of the data. In this case the input to the network is the noisy speech signal and the output of the network is compared against a clean signal. When the autoencoder is given new data it should be able to remove the noise and produce the clean speech signal.

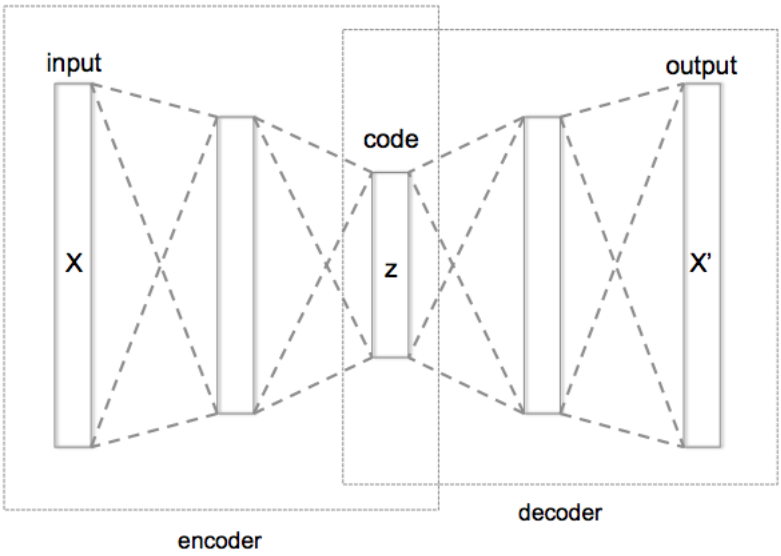


Figure 2.23: Schematic overview of an AutoEncoder [57] where the input (X) is offered to the encoder which changes it in a latent description (z). This latent description is then used by the decoder to recreate the image (X')

Generative Adversarial Network

Generative Adversarial Networks (GAN) [58] are trained to generate new input from white noise (see Figure 2.24). This input is based on real word examples. A GAN consists of two parts:

1. Generator creating the input from white noise
2. Discriminator which classifies the samples as being real or coming from the generator.

The GAN uses the input to learn how to represent this from white noise. To do this the discriminator receives two inputs and determines which is coming from the generator and which is real input. This allows the generator to create output that is more similar to the real data making it more difficult to distinguish between the two. The white noise that the generator is using can be seen as a latent description used in the AutoEncoder (see Section 2.4.3). After training, the discriminator is discarded and the generator is used to create new data. GANs can be used for separating speakers from a mixture. Instead of having a latent description the mixture is given as input to the generator. The outputs of the generator are the unmixed speaker files which are provided to the discriminator together with the real

unmixed speaker files. The discriminator then classifies these. After training has finished the generator should be able to unmix new mixtures.

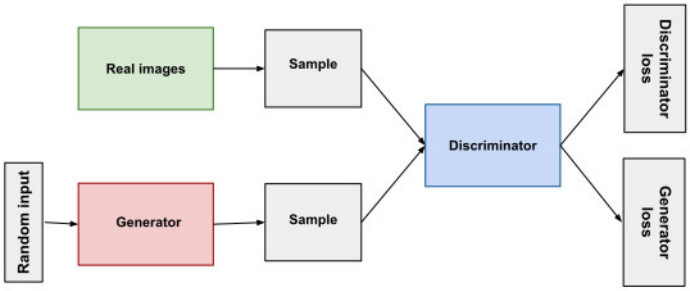
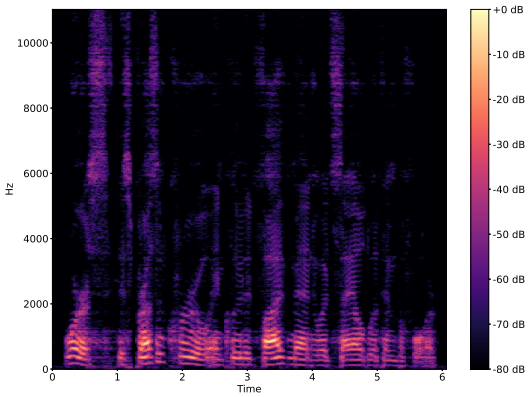


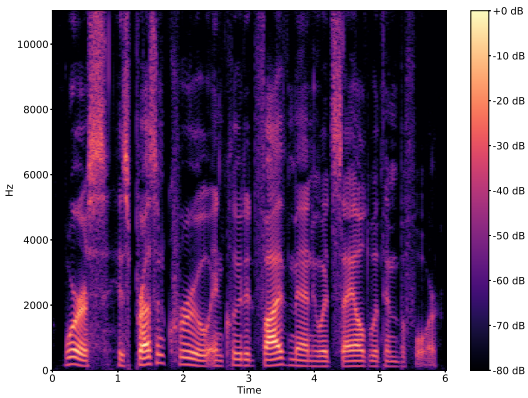
Figure 2.24: Schematic overview of an generative adversarial network [59] showing the random input used by the generator to create a sample image which is pass to the discriminator together with a real image. The discriminator produces two output one showing the performance of the generator and one for the performance of the discriminator.

2.5 Measurements

Signal-to-artifact (SAR), signal-to-distortion (SDR) and signal-to-interference (SIR) ratios are used for measuring the performance of speaker separation algorithms (see Vincent et al. [1]). Each of these measure a different aspect of the signal and are designed to work with multichannel recordings. SAR describes the number of artefacts present in the result of the algorithm. These artefacts are introduced by the algorithm during the process of separating the speakers. The distortions measured by the SDR are similar to the noise introduced by the algorithm. This measurement is also expanded in a scale invariant version called (SISDR) [60] that reduces the effect of the amplitude on the result. The SIR measurement measures how much of the interfering signal is still present in the result of the algorithms. In addition to these measurements, the perceptual evaluation of speech quality (PESQ) is used. This last measurement describes the quality of noise reduction and dereverberation algorithms and looks at how listeners would perceive the quality of the recording.



(a) The spectrogram of the subband projection ($P_{\text{approx\_j}}$) with a SNR of -5 dB.



(b) The spectrogram of the reference signal containing no noise.

Figure 2.25: The spectrograms of the subspace projection ($P_{\text{approx\_j}}$) and the reference source used for calculating the different measurements (SAR, SDR, SISDR and SIR). The subspace projection contains 5 dB of noise.

2.5.1 SAR and SIR

Artefacts are described as "burbling" noise or also called musical noise. This kind of noise is created by random statistical variations in the different frequency bins or as a left over product of noise reduction where spectral subtraction is applied. These sounds cannot be classified as distortion nor attributed to interfering sound sources.

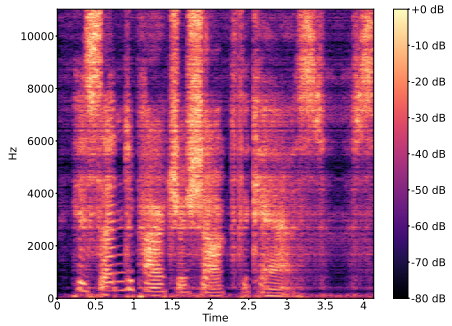
For the SAR measurement (see Equation 2.86), the artefacts ($\text{error}_{\text{artifacts}}$) are calculated by subtracting the subspace projection of all estimated sources ($P_{\text{approx\_J}}$

from the reference source (see Equation 2.87). Using a subspace projection means that the best fit for every subspace is found - this is advantageous when the signals are not aligned properly.

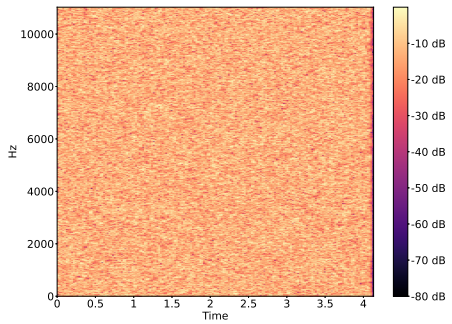
$$\text{SAR} = \frac{||S_{target}||^2}{||e_{interference} + e_{filterdistortion} + e_{artifacts}||^2} \tag{2.86}$$

Next to the artefacts, the SAR also depends on the interference and filter distortion. The latter is calculated per source by subtracting the reference signal from the subspace projection within a single source ($P_{\text{approx\_j}}$) but looking at all channels (I) for that source. Whereas, the interference is calculated by subtracting the least-squares subspace projection of one estimated source from that of all estimated sources.

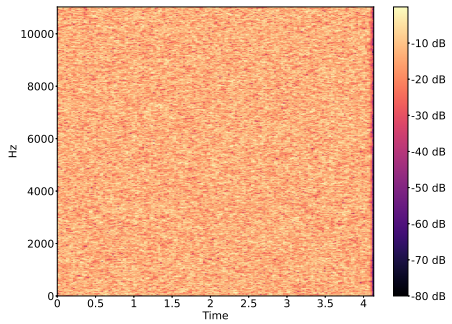
$$\begin{aligned}
\text{error}_{\text{filter distortion}} &= P_{\text{approx_j}} - \text{reference signal} \\
\text{error}_{\text{artifacts}} &= P_{\text{approx_j}} - P_{\text{approx_J}} \\
\text{error}_{\text{interference}} &= \text{reference signal} - P_{\text{approx_J}}
\end{aligned} \tag{2.87}$$



(a) The spectrogram of the filter distortion error ($P_{\text{approx\_j}}$ − reference signal.)



(b) The spectrogram of the artefacts error ($P_{\text{approx\_j}} - P_{\text{approx\_J}}$.)



(c) The spectrogram of the interference error (reference signal − $P_{\text{approx\_J}}$.)

Figure 2.26: The spectrograms of the different variables ($\text{error}_{\text{filter distortion}}$, $\text{error}_{\text{artifacts}}$ and $\text{error}_{\text{interference}}$) used for calculating the measurements (SAR, SDR and SIR).

Sounds from other sources, than the one of interest, that are also present in a recording are called interferences. These are found when the recording is compared with the ground truth that does not have interferences. Interference can range from reverberation to other speakers. To calculate the interference, a ground truth signal (S_{target}) is needed which is divided by the interferences ($error_{interference}$) to create the SIR (see Equation 2.88).

$$\text{SIR} = \frac{||S_{target}||^2}{||error_{interference}||^2} \tag{2.88}$$

2.5.2 SDR and SISDR

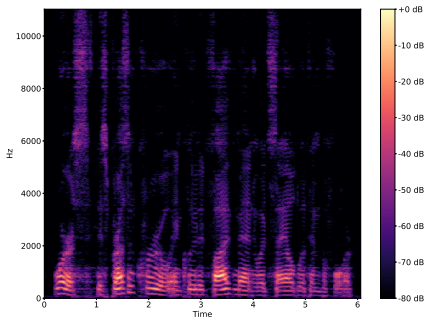
Both SDR and SISDR are closely related to the signal-to-noise (SNR) ratio. The main difference is the way they are calculated. All three measurements (SNR in-

cluded) need a reference signal. Dividing the reference signal by the noise gives us the SNR (see Equation 2.89). Noise is calculated by subtracting the approximated signal from the reference signal.

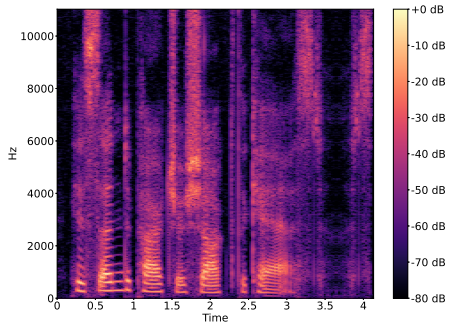
$$\begin{aligned} &\text{source} = \text{reference source} \\ &\text{noise} = (\text{reference signal} - \text{approximated signal}) \\ &\text{SNR} = 10log_{10}(\text{source/noise}) \end{aligned} \tag{2.89}$$

The SISDR uses the same approach but uses a scaling factor to make sure both signals have the same amplitude (see Equation 2.90).

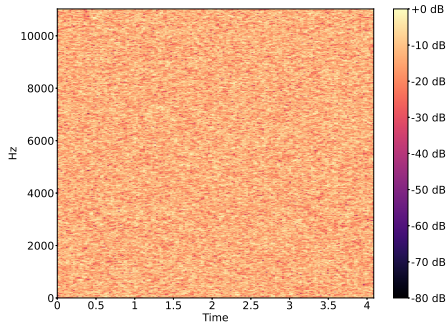
$$\begin{aligned} &s = \text{approximated signal} * \text{reference signal}/||\text{reference signal}||^2 \\ &\text{source} = s * \text{reference signal} \\ &\text{noise} = (s * \text{reference signal} - \text{approximated signal}) \\ &\text{SISDR} = 10log_{10}(\text{source/noise}) \end{aligned} \tag{2.90}$$



(a) The spectrogram of the scaled version of $P_{\text{approx\_j}}$ with a SNR of -5 dB.



(b) The spectrogram of $P_{\text{approx\_j}}$ with a SNR of -5 dB.



(c) The spectrogram of the noise ($s *$ reference signal $-$ approximated signal).

Figure 2.27: The spectrograms of the SISDR calculation compared with the subspace projection used for the SDR measurement.

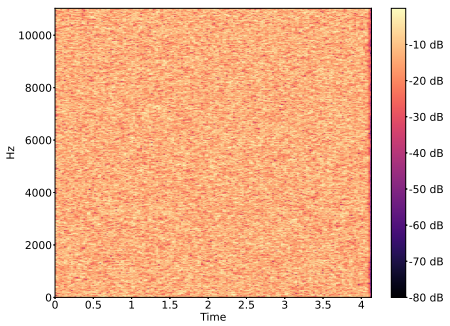
The SDR assumes that there are three different types of error (artefacts, interference and spatial distortion). These three are used to calculate the SDR (see Equations 2.87 and 2.91).

$$\begin{aligned} \text{source} &= \text{reference signal} + \text{error}_{\text{filter distortion}} \\ \text{noise} &= (\text{error}_{\text{artifacts}} + \text{error}_{\text{interference}}) \\ \text{SDR} &= 10log_{10}(\text{source}/\text{noise}) \end{aligned} \tag{2.91}$$

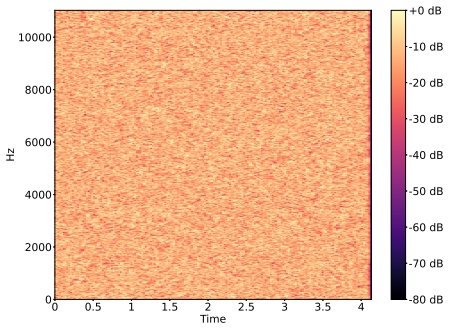
$$\text{source} = \text{reference signal}$$

$$\text{noise} = (\text{error}_{\text{artifacts}} + \text{error}_{\text{filter distortion}} + \text{error}_{\text{interference}}) \tag{2.92}$$

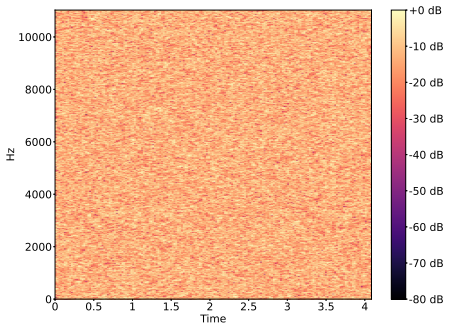
$$\text{SDR}_{\text{mir}} = 10log_{10}(\text{source}/\text{noise})$$



(a) The spectrogram of the noise used for SDR.



(b) The spectrogram of the noise used for SDR_{mir}.



(c) The spectrogram of the noise used for the SISDR.

Figure 2.28: The spectrograms of the different calculations of noise used SDR, SDR_{mir} and SISDR.

There are two implementations of the SDR measurement; one works as described, the second adds the filter distortion to the reference signal before dividing it by the artefacts and the interference. This gives a bias towards the reference signal, finding less noise in the noisy signal and returning a higher value (see Equation 2.92). This bias is more pronounced when using the correlation between clean and noisy speech (see Figure 2.31). It can also be seen in a noisy or reverberant gunshot as well as

reverberant speech.

The measurement for calculating the SAR is also dependent on the filter distortion and the interference. If there is no interfering signal for the SIR to measure apart from the reverberation, the value for the SIR will be high. For the SAR this means that it resembles the SDR too closely to be informative. When running the SAR and SDR on 800 files from the TIMIT dataset which have a very low interference, the main input for both measurements is coming from the number of artefacts (see Figure 2.29). The result of both measurements over these 800 files is the same, when looking at one file no difference can be seen. When looking at the results of all 800 files, then there is no significant difference between the two measurements (see Figure 2.29). The results for the individual number for the artefacts, distortion and interference, suggest that the artefacts are the main contributor for both the SDR and the SAR measurements (see Figure 2.30).

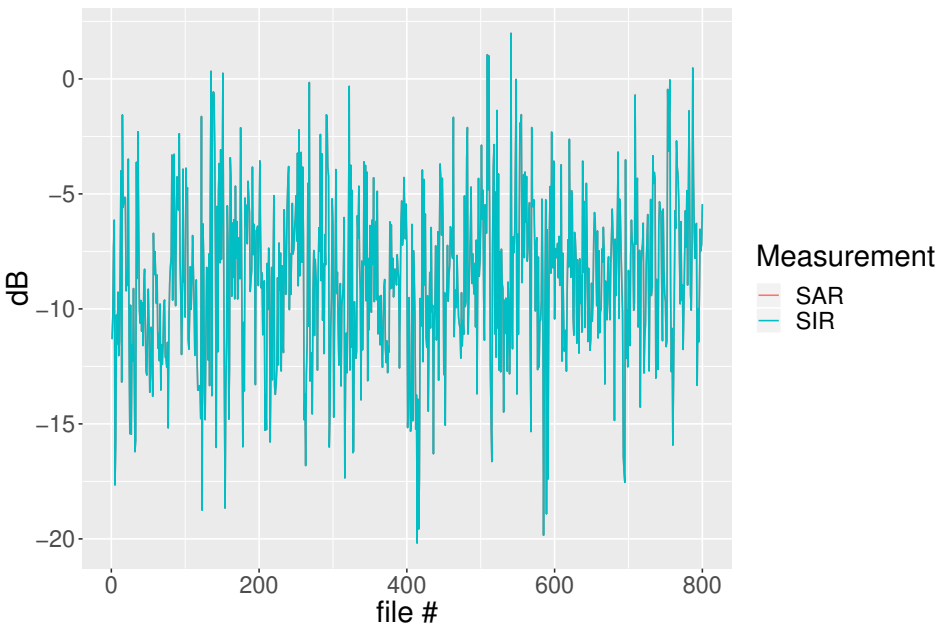


Figure 2.29: A comparison between the SAR and SDR values over 800 single speaker files. The values represent per file results.

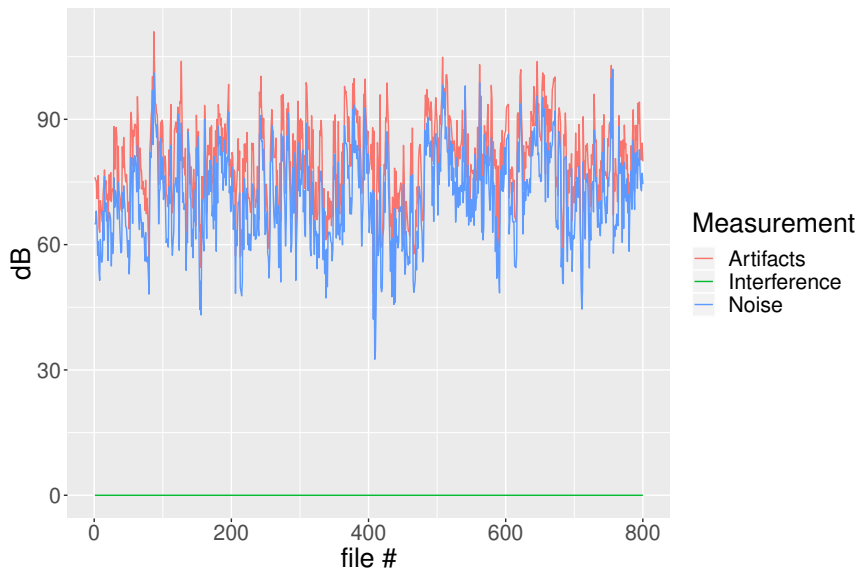


Figure 2.30: A per file comparison between the number of artefacts, noise and interference present in 800 single speaker files.

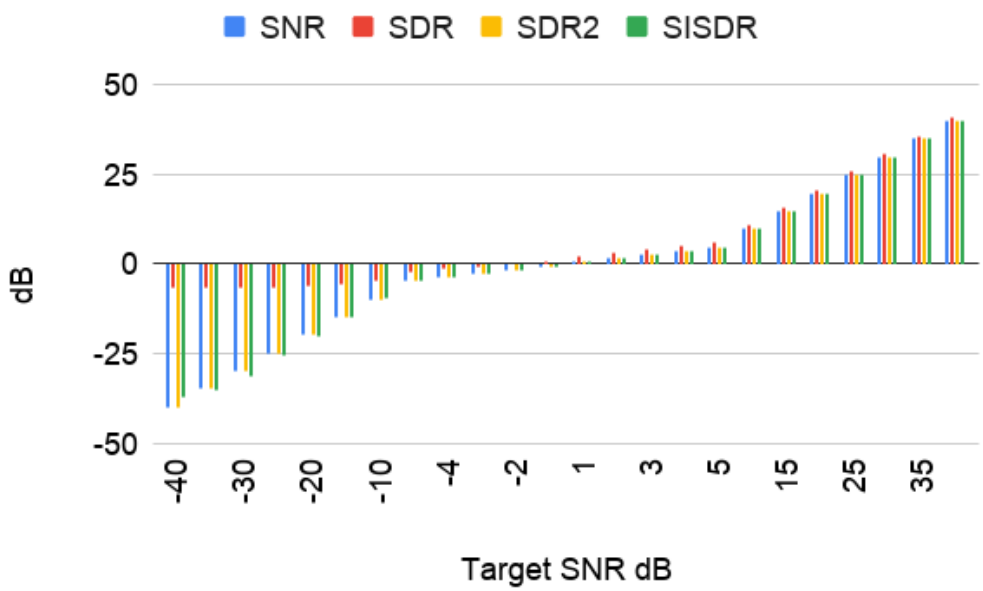
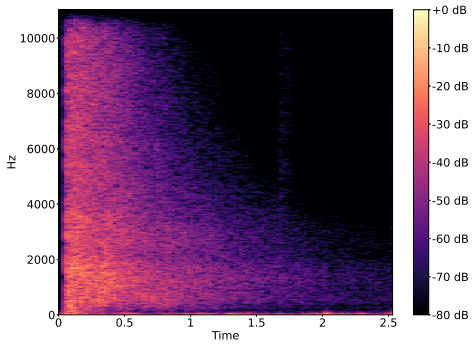


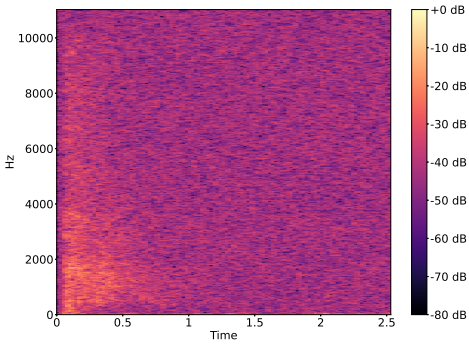
Figure 2.31: Comparing the result of the different measurements using different target SNR showing the bias of the measurements towards a positive (> 0 dB) result.

Applying SDR and SISDR to gunshot data

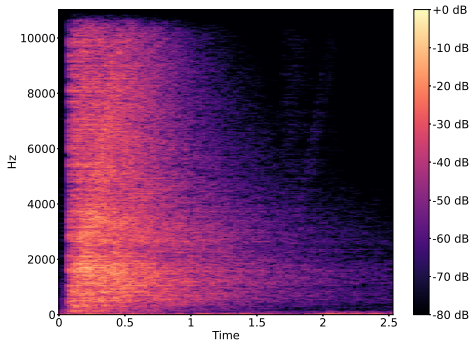
A gunshot has a clear start and end to the sound (see Figure 2.32a). This makes it easier if we want to use it for denoising or dereverberation. When we add -5 dB noise to the signal (using Equation 2.93) we can still see the start of the gunshot and



(a) The STFT of a gunshot



(b) The STFT of a gunshot with -5 dB added noise



(c) The STFT of a reverberant gunshot (RT_{60}=1.0)

Figure 2.32: The STFT of the normal, noisy and reverberant gunshot signal

its end (see Figure 2.32b). If instead of noise we place the gunshot in a reverberant room (RT_{60} is 1 second) then we see that the signal gets stretched but it still has a clear start and end (see Figure 2.32c).

$$
\begin{aligned}
noise_{db} &= |x_{db}| - SNR_{db} \\
N &\sim \mathcal{N}(0, 10^{\frac{noise_{db}}{10}}) \\
y &= x + N
\end{aligned}
\tag{2.93}
$$

To see what the three different measurements consider to be noise, the original signal is subtracted from the noisy signal. The latter is then divided by the original signal to show the noise levels in the noisy signal (see Figure 2.33a). The process is also applied to reverberant signals (see Figure 2.33b) to determine if there is a

difference between the recognition of noise and reverberation.

We see that in the case of the noise, both SDR measurements show artefacts introduced at the end of the signal (see Figures 2.33c and 2.33d). The reverberation case shows that the outcome of the SDR_{mir} and SISDR is similar to the reverberation (see Figures 2.32c, 2.33h and 2.33f) whereas the SDR is biased towards the original sound (see Figure 2.33d), this was also described by LeRoux et al. [60]. As an additional test an approximated signal is used which is given by the H_1 correlation between the signal and the noisy or reverberant signal (see Equation 2.26). We see that the SDR_{mir} measurement (see Figure 2.34g) follows the SNR (see Figure 2.34c) closely where the SDR and SISDR measurements (see Figures 2.34e and 2.34i) detect more noise and reverberation. This confirms that the SDR_{mir} is similar to the SNR. However, the SISDR lacks this similarity and is better at detecting noise than the SDR_{mir} and the SNR. Adding the filter distortion error seems to replicate this, however, it also detects parts of the signal as noise.

2.5.3 PESQ

The perceptual evaluation of speech quality (PESQ) developed by Rix et al. [2, 61] is a measurement that runs from 1.0 to 4.5 where 1.0 is the worst quality and 4.5 is the best quality. PESQ predicts how users perceive the recording when they are listening to it. First, the files are time aligned using a narrowband filter for emphasising perceptual important parts, division of the reference signal in utterances, utterance alignment, splitting and re-alignment to test for delay changes during speech (see Figure 2.35). This process is followed by transforming the signal in to the power spectrum in order to do frequency and gain equalisation and loudness mapping. The latter part of the process gives the perceived loudness of the files. After time and level alignment, the algorithm compares the two signals and determines the localised errors also called disturbances. These include deletion or negative delay change where there is an overlapping section in the degraded signal and masking where the disturbance needs to be above a threshold to be counted. Localised errors are counted on a frequency level basis giving a frame-by-frame measure of the perceived distortion, frequency wrapping and loudness scaling. The representations are processed to calculate the severe effects and rapid variation between the two

signals.

Applying PESQ to speech data

PESQ is used to evaluate the speech quality of audio recordings when there is noise and reverberation present. The result of PESQ shows how well the algorithms work, this can be simulated with adding noise to a speech signal and comparing this to the original signal. For this, the same parameters as for the gunshot data in Section 2.5.2 are used, the signal-to-noise ratio is varied from -10 to 10 dB and the RT_{60} time from 0 to 2 seconds, both using increments of 5 units (5 dB and 0.5 sec respectively).

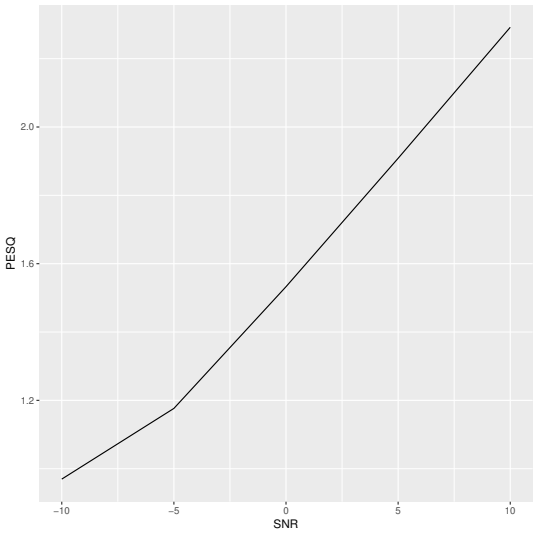


Figure 2.36: Applying PESQ to a file with varying SNR values

There is a relation between the increase in SNR and the value of PESQ (see Figure 2.36) when the SNR increases the value of PESQ also increases. This shows that when there is more signal than noise present the result according to PESQ is higher.

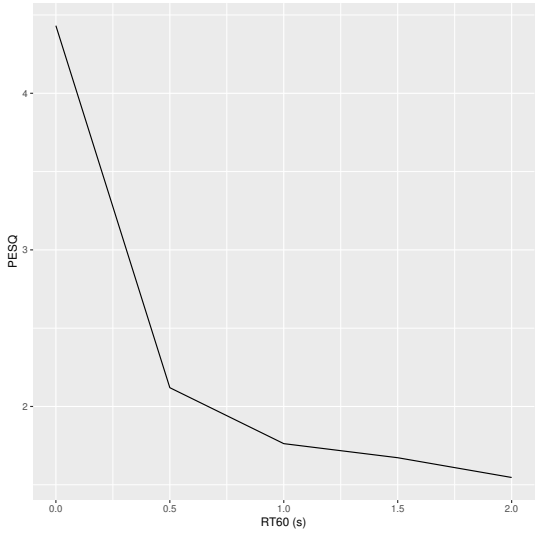


Figure 2.37: Applying PESQ to different RT_{60} times.

In the case of reverberation the same pattern can be seen (see Figure 2.37). When more reverberation (higher RT_{60} time) is added to the signal the result on the PESQ scale is lower. However, unlike the SNR case, the result is not linear but exponential. This shows that the reverberation is affecting the signal more than the SNR does.

2.5.4 Cepstral Distance

The cepstral distance (CD) [62] is the Euclidean distance in the cepstrum domain (see Equation 2.94). It measures the distance between two cepstrum coefficients and can be used for voice activity detection (VAD), emotion classification and to measure the enhancement of a speech signal.

$$D = \sum_{n=0}^{N} (c(n) - c(n+1)) \tag{2.94}$$

2.6 Corpora/Data collection

2.6.1 Near field and far field recordings

The difference between near field and far field is the distance between the sound source and the microphone. Hansen [63] and Siano et al. [64] classify near field as the region of the sound field where the sound pressure does not decrease by 6dB

each time the distance travelled from the source is doubled. Far field is the region of the sound field where the sound pressure decreases each time the distance travelled from the source is doubled. When the sound source is close to the microphone then the sound does not have far to travel therefore the sound pressure does not decay significantly. On the other hand when there is a large distance between the sound source and the microphone where the sound has to travel a significant distance then the sound pressure has time to decay. The case of the former is called near field where the decay or decrease is less than 6 dB each time the distance travelled is doubled, the latter is call far field when the decay is 6dB or more each time the distance travelled is doubled. Doclo et al. [65] describe a formula for calculating the distance where the near-field regions ends and far-field starts. This formula uses the size of a microphone array to calculate the minimum distance for this assumption (see Equation 2.95).

$$R \leftarrow \frac{(L_{m_{M-1}} - L_{m_0}) f_s}{v} \tag{2.95}$$

For example, when using a microphone array with a distance between microphones of 1.5 metres and a sampling frequency of 16kHz, the far field area starts at 160 metres. When the sampling frequency is reduced to 8kHz, the far-field area starts at 80 metres. This formula assumes a linear array and does not work for a circular array where the first and last microphones are close together. Instead, for a non-linear array $L_{m_{M-1}} - L_{m_0}$ is replaced by the maximum distance between two microphones. This definition is used for microphone arrays but not for single microphones. In the case of single microphone recordings, the minimum distance for far field is considered to be 2 metres by Zhao et al [66], whereas Gelbart et al. [67] consider the area from 3 feet (0.91 metres) onwards to be far-field. Therefore, in this thesis the area from 1m onwards is considered to be far-field for monaural recordings (i.e. recordings made with one microphone).

2.6.2 Microphone Array

Microphone arrays are used for a number of things, including: sound source tracking, noise reduction and speaker separation. These microphone arrays differ greatly in size. From a four microphones array placed in the Microsoft Kinect or a circular

array with 32 microphones and 440mm in diameter [68] to a 4096 microphones array [69, 70]. Beamforming is traditionally used for sound source localisation with Delay-and-Sum being the most popular method. The smaller sized microphone arrays are implemented in robots like Softbanks' Nao [71] and Pepper [72, 73]. It helps the robot to localise the speaker and follow them around with its head. For noise reduction, the placement of one microphone pointing away from (instead of towards) the speaker could help in determining the noise sources and in removing these from speech [74, 75]. Microphone arrays can be used in combination with beamforming algorithms to determine where the sound sources are located. There are different beamforming algorithms that work in either the frequency domain or the time domain.

2.6.3 Beamforming

Beamforming works by aligning peaks in the recordings of different microphones. It works under the assumption that the location of each microphone is known and that each recording is of the same subject. For each microphone the field of view needs to be known, this being the angles from which it is still possible for a microphone to record a sound. Together with the distance between microphones and location of the microphone, this is used to determine the direction of arrival of the sound. The output of a beamforming algorithm is an area with high sound pressure, which should correspond to the sound source that has to be tracked.

The concentration is the four most used beamforming algorithms:

- Two compute the time-difference of arrival (TDoA) locally in each time-frequency bin (these are Generalized cross-correlation with phase transform, GCC-PHAT [76, 77], or with a nonlinear function, GCC-NONLINEAR [78]).
- Two build a TDoA function for each time-frequency bin that is likely to get a high value for the true TDoA and pool it across the time-frequency plane to get an angular spectrum (these are minimum variance distortionless response, MVDR [79, 80], and minimum variance distortionless response weighted, MVDRW [80]).

The input for the methods is an empirical covariance matrix (ECM). This is calculated using the neighbourhood of every time-frequency bin (t,f), by multiplying

a neighbourhood windowing function (w) of length L_f x L_t by the input signal (x) of all microphones and the Hermitian transpose of the input signal (see Equation 2.96).

$$\widetilde{X}_{mm}(t,f) = \frac{\sum_{t',f'} w(t'-t, f'-f)x(t',f')x(t',f')^H}{\sum_{t',f'} w(t'-t, f'-f)} \tag{2.96}$$

GCC-PHAT and MVDR are both angular spectrum techniques in which the methods try to construct a function ϕ of the TDoA (τ) where the peaks indicate the TDoAs of the different sources. GCC-PHAT assumes that in each time-frequency bin the sound of one source is more noticeable than of the others. The TDoA of this source is estimated by taking the phase difference between two channels (see Equation 2.97).

$$\phi^{\mathrm{GCC}}(t,f,\tau) = real\left(\frac{\widetilde{X}_{mm}(t,f)_{1,2}}{|\widetilde{X}_{mm}(t,f)_{1,2}|}e^{-2i\pi f\tau}\right) \tag{2.97}$$

For MVDR, the signal-to-noise ratio (SNR) between the sound power and residual power in the direction of the TDoA is used. This function overestimates the SNR at the low frequencies, where the phase differences are small. This happens regardless of the number of sources. MVDR uses a steering vector (see Equation 2.98) for estimating the power in the direction τ (see Equation 2.99). The residual power is computed by subtracting the estimated power in the direction τ from the total power. This is used to calculate the SNR in the direction τ (see Equation 2.100).

$$\mathrm{d}(f,\tau_n) = [1, e^{-2i\pi f\tau}]^T \tag{2.98}$$

$$\mathrm{P}(t,f,\tau_n) = \left(d(f,\tau)^H \widetilde{X}_{mm}(t,f)^{-1} d(f,\tau)\right)^{-1} \tag{2.99}$$

$$\phi^{\mathrm{MVDR}}(t,f,\tau) = \frac{P(t,f,\tau)}{\frac{1}{2}tr\left(\widetilde{X}_{mm}(t,f)\right) - P(t,f,\tau)} \tag{2.100}$$

GCC-NONLINEAR and MVDRW are variants of GCC-PHAT and MVDR respectively. GCC-NONLINEAR uses a nonlinear function which assumes sparseness in the sound signal (see Equation 2.102). α is a non-linear parameter based on the

speed of sound, the distance between microphones d and the sampling frequency (see Equation 2.101).

$$\alpha = \frac{10 * c}{d * F_s} \tag{2.101}$$

$$\phi^{\text{GCC-NONLINEAR}}(t, f, \tau) = 1 - tanh(\alpha(|2 - 2 * GCC - PHAT(t, f, \tau)|)^{1/2}) \tag{2.102}$$

MVDRW is the weighted version of MVDR and uses frequency weighted beamforming. MVDRW assumes that the input signal consists of a single source TDoA and diffuse noise. Instead of using the ECM as input it uses a covariance matrix and inverts the MVDR. However, because it is based on the MVDR and the input is different, the MVDR equation changes too (see Equation 2.103). The weight factor w_d (see Equation 2.104) depends on the frequency, the distance between microphones d and the speed of sound. This factor reduces the impact on frequencies that are below 1kHz.

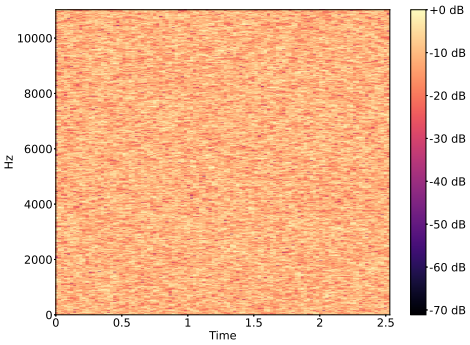
$$\phi^{\text{MVDR}}(t, f, \tau) = \frac{1 + 2v^s(t, f, \tau)/v^b(t, f, \tau) + sinc(2\pi f \frac{d}{c})}{1 - sinc(2\pi f \frac{d}{c})} \tag{2.103}$$

$$\mathrm{w}_d(f) = \frac{1}{2}(1 - sinc(2\pi f \frac{d}{c})) \tag{2.104}$$

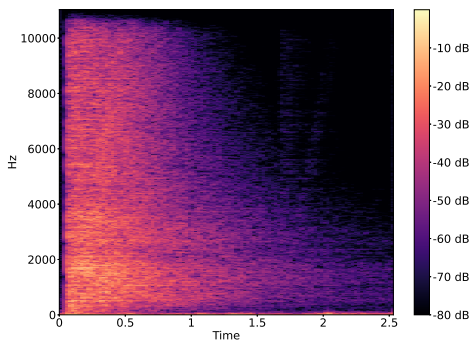
$$\phi^{\text{MVDRW}}(t, f, \tau) = w_d(f)\phi^{\text{MVDR}}(t, f, \tau) + w_d(f) - 1 \tag{2.105}$$

2.7 Conclusion

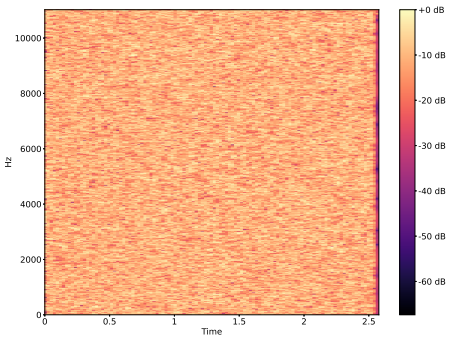
The techniques described in the chapter are used as the foundation of the coming chapters. In Chapter 3, related work based on the WPE, NMF and deep learning techniques are described. The beamforming techniques are used in Chapters 3, 4 and 7. Techniques as the Fourier transform, cost functions and window functions form the basis for the experiments described in Chapters 5, 6 and 7.



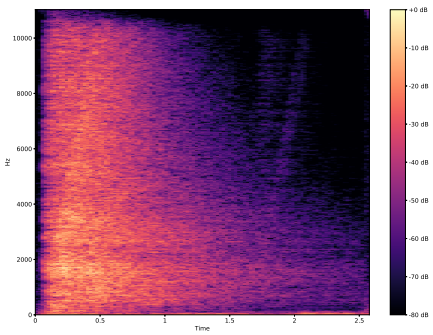
(a) The STFT of the noise added to the gunshot



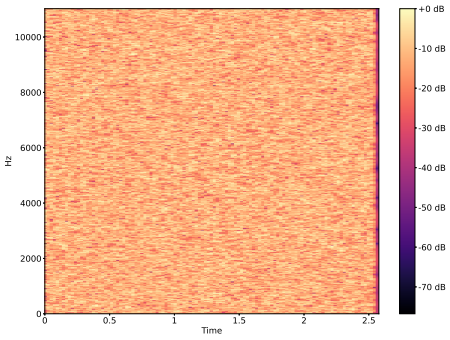
(b) The STFT of the reverberation added to the gunshot



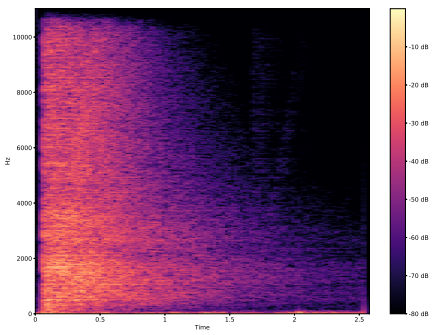
(c) Noise of the SDR measurement of an approximated noisy gunshot



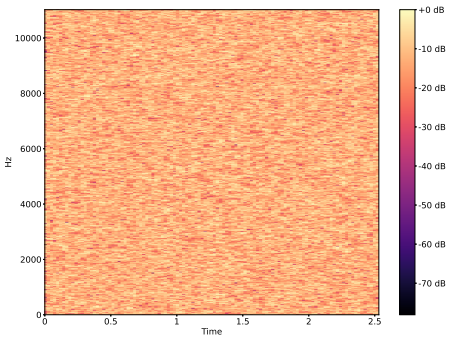
(d) Noise of the SDR measurement of an approximated reverberant gunshot



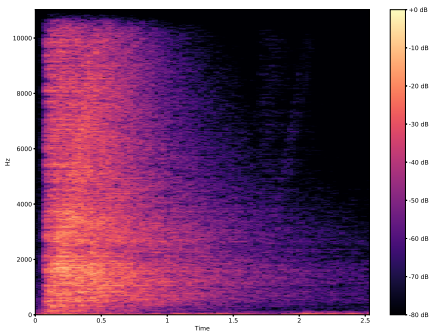
(e) Noise of the SDR_{mir} measurement of a noisy gunshot



(f) Noise of the SDR_{mir} measurement of a reverberant gunshot

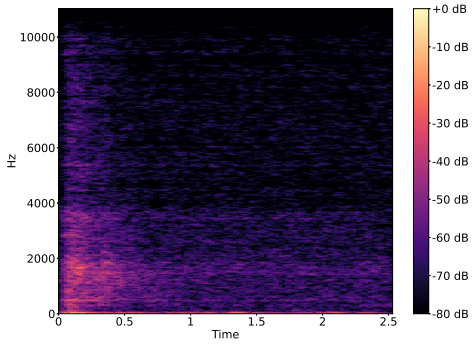


(g) Noise of the SISDR measurement of a noisy gunshot

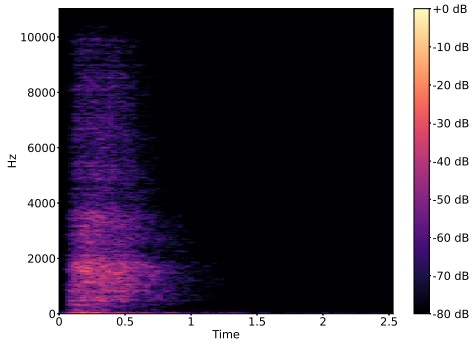


(h) Noise of the SISDR measurement of a reverberant gunshot

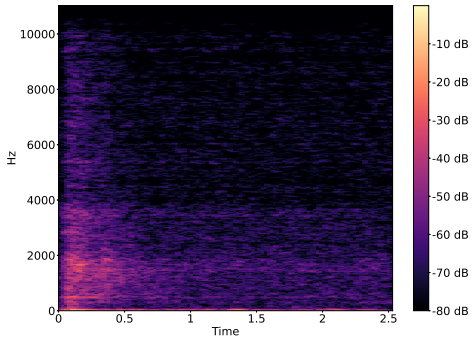
Figure 2.33: The spectrogram of a gunshot in a reverberant and noisy environment of which the noise of the SDR, SDR_{mir} and SISDR measurements are calculated.



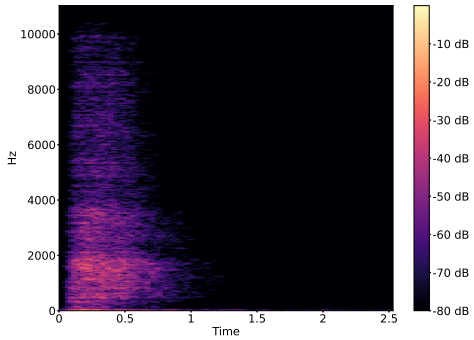
(a) H_1 result of denoising the gunshot



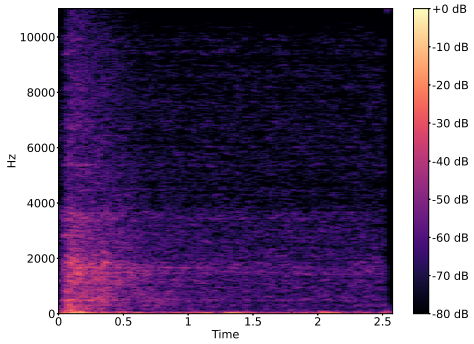
(b) H_1 result of dereverbarating the gunshot



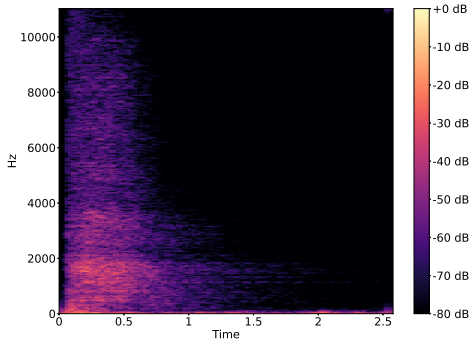
(c) Noise of the SNR measurement of an approximated noisy gunshot



(d) Noise of the SNR measurement of an approximated reverberant gunshot

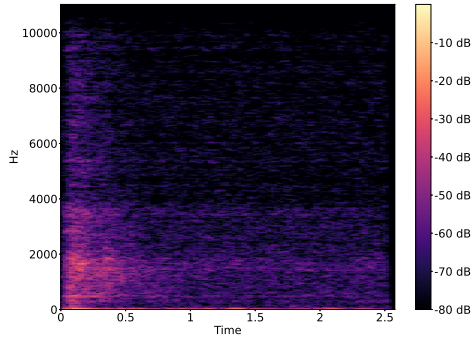


(e) Noise of the SDR measurement of an approximated noisy gunshot

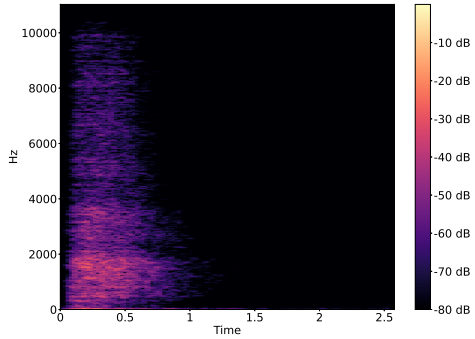


(f) Noise of the SDR measurement of an approximated reverberant gunshot

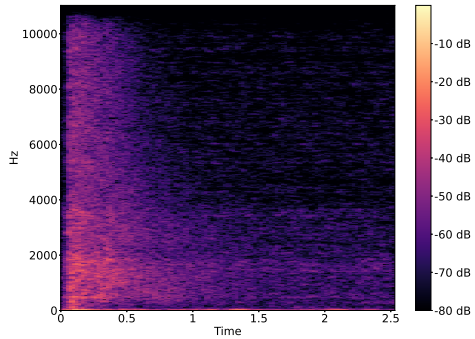
Figure 2.34: The H_1 results on the noisy and reverberant gunshot signals with the result of the SDR, SDR_{mir} and SISDR measurements.



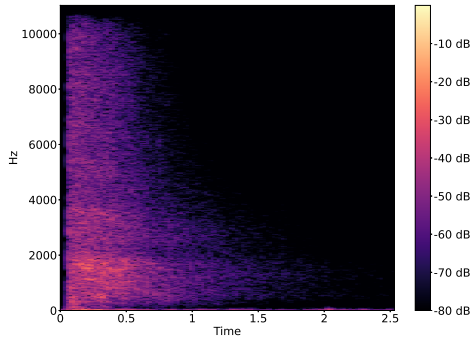
(g) Noise of the $\mathrm{SDR}_{\mathrm{mir}}$ measurement of an approximated noisy gunshot



(h) Noise of the $\mathrm{SDR}_{\mathrm{mir}}$ measurement of an approximated reverberant gunshot



(i) Noise of the SISDR measurement of an approximated noisy gunshot



(j) Noise of the SISDR measurement of an approximated reverberant signal

Figure 2.34: The H_1 results on the noisy and reverberant gunshot signals with the result of the SDR, $\mathrm{SDR}_{\mathrm{mir}}$ and SISDR measurements (cont.).

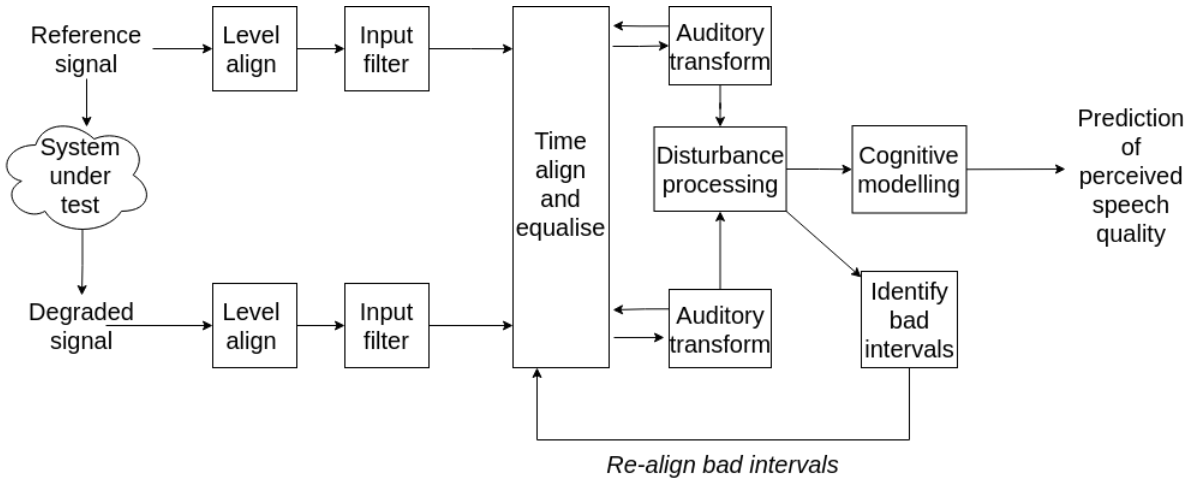


Figure 2.35: The flow of calculating the perceptual evaluation of speech quality value [61]

Chapter 3

Related work

Many corpora have been created for the training and testing of algorithms. These corpora are often designed with a specific task in mind whether it is to do dereverberation (REVERB [81]), speech recognition (TIMIT [3]) or speaker separation (CHiME5 [82]). The task often defines the recording conditions and environments of the corpus. These corpora are used as a benchmark to measure different algorithms against. It is not always the case that a corpus is used for the task it was designed for (e.g. TIMIT is being used for dereverberation and noise reduction).

Just like corpora are designed for a specific task so are algorithms. However, algorithms are often less adaptable once they are trained. If their purpose is changed the algorithm will have to be redesigned or retrained. Certain parts of the algorithm (e.g. permutation invariant training) can be used for different purposes, for example, for dereverberation or speaker separation. Deep learning techniques use the underlying method but the network is redesigned to fit the purpose of the algorithm.

This chapter consists of four main parts:

- Speech Corpora, where the different corpora are described that have been used for dereverberation and speaker separation.
- Dereverberation, describing the different algorithms and their performance for this purpose
- Single channel speaker separation, this is divided into non-negative matrix factorisation and deep learning and describes the algorithms and their performance.

- Multichannel speaker separation, this contains different algorithms and their performance which is divided into non-negative matrix factorisation and deep learning.

3.1 Speech Corpora

There are multi-speaker corpora (see Tables 3.1 and 3.2), some of which do contain reverberation, but mainly they are recordings of conversations that contain little to no overlapping speech. Also, many of the corpora only contain static speech, meaning that the speakers are located in a fixed place during the recording. These corpora are compared based on:

- whether they contain scripted or natural conversations
- whether they are recorded in different environments
- whether they contain reverberation or noise
- whether they are recorded with a microphone array and/or contain a close-talk channel
- what the maximum distance is between the speaker and the microphone
- what they are used for
- if they have multiple speakers in a single recording
- whether the speakers move around or are stationary

When a corpus contains scripted speech, it means that it can be used for mixing recordings to create overlapping speech because the speaker is speaking continuously during the recording. Whereas when it is a natural conversation there is a second speaker who will fill in the pauses created by the first speaker. If a corpus is recorded in multiple rooms or contains noise and/or reverberation it means that it can be used for testing noise reduction and dereverberation. In addition to this, when the corpus is used for speaker separation the problem becomes more difficult if the algorithm has to deal with noise and reverberation than when these are not present. The location of the recording also suggests which kind of noise can be expected in the recording. It is unlikely to hear an extractor fan in an office environment for example. When the corpus only contains a close-talk channel, it can be used for simulating different rooms. On the other hand, when there is also a recording

present from a microphone array, the close-talk channel can be used as a ground-truth for comparing the output of an algorithm against. If the speaker is further away from the microphone then the speech has more time to interact with the environment, meaning that using these corpora will make it more difficult for an algorithm to match the performance on a corpus with a shorter distance. If the corpus has multiple speakers in a single recording it means that the recordings can be used for speaker separation and tracking. This is more complicated when they are moving around because the location needs to be determined every timestep and measurements such as time-difference of arrival change continuously. These eight items present an overview of what a corpus can be used for and what the complexity is of this corpus.

TIMIT corpus [3] contains recordings of short utterances and is designed for testing automatic speech recognition (ASR) systems. It contains different American English dialects. The recordings are near-field with speakers speaking clearly in the microphone. There is no noise or reverberation present in the recordings. The recordings are annotated on word and phoneme level. This corpus is also used for evaluating dereverberation algorithms and extended in GlobalTIMIT [83] which includes speakers of different languages. However, the recordings for GlobalTIMIT were made with head-mounted noise-cancelling microphones to get a clean recording with as little noise as possible. The CSR-WSJ corpus [84] is similar to TIMIT but was originally designed for statistical language modelling. Participants read Wall Street Journal texts which were published around 1987. The corpus can be used for speaker dependent and independent training. Like TIMIT, the recorded speech is clean and transcribed, however it misses the phonetic description of the text which TIMIT includes.

The TSP corpus [85] is a single channel single speaker corpus in which a stereo microphone is placed at 15 cm from the speaker instead of a microphone close to the mouth like TIMIT, GlobalTIMIT and CSR-WSJ have. During post-processing, this was reduced to a single channel by averaging between the two channels. The recordings contain speech where the speaker reads Harvard sentences which contain an unusual word order. This is because the Harvard sentences are phonetically balanced and use phonemes at the same frequency as used in English. The majority

of the speakers were Canadian.

The REVERB corpus [81] is specifically designed for removing reverb from speakers. Unlike the previous corpora, it contains recordings made with a single microphone and with two microphone arrays (one with two microphones and one with eight). The corpus contains simulated and real data, the former has recordings made with three simulated rooms (small RT_{60}=0.3, medium RT_{60}=0.6 and large RT_{60}=0.7) and two distances between the microphone and the speaker (50cm and 2 metres). Recordings in the real corpus are made in one room with the microphones at two distances (1 metre and 2.5 metres). The corpus contains text read from the Wall Street Journal and speakers are stationary in a room. Lincoln et al. [86] describe a corpus (MC-WSJ-AV) that contains speech of people walking around a meeting room. This corpus contains reverberation but no noise. It is an expansion of the original REVERB corpus with its main difference being that the speakers are allowed to walk through a meeting room.

The vocalization corpus[1] [87] contains recorded telephone conversations of 120 different subjects. Unlike the previous corpora, background speech of a second speaker is present in the recordings. This does not provide us with a clean ground truth but allows us to use the corpus for testing in noisy environments. The HCRC MapTask corpus [88] is similar to the vocalization corpus. The main difference is that people need to explain to each other how to get from A to B on a map and use head-mounted microphones to explain this. This corpus contains speech of 64 subjects. As with the vocalization corpus, the second speaker can be heard in the background, meaning that the corpus can be used for the same purposes as the vocalization corpus.

The CHIL corpus [89] is used for tracking and separation and contains information about the speakers in 2D and 3D space. The corpus contains audio and video recordings of lectures and meetings which have been annotated on speech and location of the speaker. This corpus has been extended into the AMI corpus [90] which also contains recordings of multiple people in a meeting. The rooms in which the meetings take place are fitted with close-talking and far-field microphones. However, this corpus simulates an office environment. The speech is transcribed and objects

[1] http://www.dcs.gla.ac.uk/vincia/?p=378

referred to in the spoken dialogues have been marked.

Barker et al. [91] describe a corpus (CHiME3) that contains noisy recordings. These recordings are near-field and the speaker is stationary. The corpus can be used for the same purposes as the HCRC MapTask and vocalization corpora. It can also be used for denoising the speech signal. The CHiME5 corpus [82] is an extension of the challenge for which the CHiME 3 corpus was designed. The main differences between this and CHiME 3 are that the corpus contains speech recorded in different rooms and with the microphone array in the Microsoft Kinect. The Kinect is used instead of a custom designed microphone array that is attached to the tablet computer. In addition to these changes, the speaker carried binaural microphones for close mouth speech which both the AMI and CHIL corpora do not have. The recorded speech is not clean, it contains noise from other speakers and also from various noise sources in the room. It is similar to the CHiME3, vocalization and HCRC MapTask corpora. The speakers are allowed to walk through the room making the sound-to-microphone distance variable. There are 32 speakers in the training set and 16 in the test set. The COSINE corpus [92] does not have the location of the speaker but does have transcriptions of speech in noisy environments with channel distortions. This allows the corpus to be used for the same purposes as the CHiME3 corpus.

The VOiCES corpus [93] contains distant speech recorded in two different furnished rooms. Speech and noise are played through loudspeakers and recorded by 12 distant microphones. The rooms have different sizes and contain 3 loudspeakers playing noise and one main loudspeaker playing speech. Two microphones are placed close to the main loudspeaker whereas the rest is distributed throughout the room. There are four different conditions: one contains only ambient noise from the room and the three others contain pre-recorded noise of overlapping speech, television or music. This corpus has similar conditions to CHiME5, AMI and the REVERB corpus.

The SiSEC2008 corpus [94] contains under-, overdetermined and determined mixtures recorded in different rooms. These rooms vary from an anechoic laboratory to a cafeteria or a living room. The SiSEC2010 corpus [95] is an extension to SiSEC2008 containing datasets with speech-music mixtures as well as extensions to

datasets introduced in 2008. One of the datasets in SiSEC2010 is used for simulating human-robot interaction where the recordings were made with a robot head containing eight microphones. Both SiSEC corpora are similar to the CHiME corpora and are used for the same purposes.

corpus	**natural conversation**	**many rooms**	**high reverb**	**many noises**	**mic array**	**close-talk channel**	**far-field speech**	**max distance mic to source**
TIMIT [3]						X		<1m
Global TIMIT [83]						X		<1m
CSR-WSJ [84]						X		<1m
TSP [85]						X		<1m
REVERB [81]	X	X	X		X	X	X	2.5m
MC-WSJ-AV [86]	X	X	X		X	X	X	2.5m
CHiME-3 [91]		X		X	X	X		<1m
CHiME-5 [82]	X	X	X	X	X	X	X	<1m
vocalization [87]	X			X		X		<1m
HCRC MapTask [88]	X			X		X		<1m
COSINE [92]	X	X		X	X	X		<1m
CHIL [89]	X			X	X	X	X	≈3m
AMI [90]	X				X	X	X	≈3m
VOiCES [93]		X		X	X	X	X	
SiSEC2008 [94]	X	X	X	X	X		X	
SiSEC2010 [95]	X	X	X	X	X		X	

Table 3.1: Comparison between multi-speaker corpora

The majority of these corpora have a close talking channel (see Table 3.1) making them suitable for speaker separation. However, not many of the corpora contain reverberation (5 out of 16) or contain far-field speech (7 out of 16). The ones that do contain far-field speech (i.e. an additional channel not being the close talking channel) have recordings of less than 5 metres. This limits the testing of algorithms that are not tested on distances over 5 metres. There is also a limited number of corpora offering moving speakers meaning that the algorithms are trained and tested on stationary speakers where the reverberation does not change due to a moving speaker. In Chapter 4, a corpus is introduced which addresses these limitations.

3.2 Dereverberation

Dereverberation is the process of removing the reverberation from an audio file creating a nonreverberant file that can be further processed such as by a speaker separation algorithm. There are various different algorithms used for this process from long short-term memory networks to minimum variance distortionless response. These techniques (see Table 3.3) are where possible compared on their performance using four different measurements:

1. cepstral distance (CD) [62]
2. signal-to-distortion ratio (SDR) [1]
3. signal-to-noise ratio (SNR) [1]
4. perceptual evaluation of speech quality (PESQ) [2]

One application of dereverberation is to improve the estimation of the time-difference of arrival (TDoA), when an audio file has reverberation then it is difficult to determine the TDoA. One approach is to use a neural network to select the frames that are clean and use these for determining the TDoA. Wang et al. [96] and Mack et al. [97] use this kind of masking to create an ideal ratio mask (IRM). However, Wang et al. [96] use it as an input for a generalized cross correlation with phase transform (GCC-PHAT) algorithm. This uses utterances from the TIMIT corpus and is successful in creating the masks. On the other hand, Mack et al. [97] apply minimum mean squared error to create a ratio mask. For testing this method, different source to microphone distances are used as well as different reverberation

corpus	usage	recording environment	multi-speaker	moving speakers
TIMIT [3]	recognition, dereverberation, speaker separation	lab		
Global TIMIT [83]	speech recognition	lab		
CSR-WSJ [84]	speech recognition, speaker separation	lab		
TSP [85]	speech recognition, denoising, dereverberation, speaker separation	lab		
REVERB [81]	dereverberation	4 reverberant rooms		
MC-WSJ-AV [86]	dereverberation	4 reverberant rooms		X
CHiME-3 [91]	denoise, dereverberation, target speaker separation	outdoor, cafe	X	
CHiME-5 [82]	denoise, dereverberation, speaker separation	kitchen, dining, living room	X	X
COSINE [92]	denoising, dereverberation, target speaker separation	noisy	X	X
vocalization [87]	dereverberation, speaker separation	lab	X	
HCRC MapTask [88]	dereverberation, speaker separation	lab	X	
CHIL [89]	speaker tracking, speaker separation	office	X	X
AMI [90]	speaker tracking, speaker separation	office	X	X
VOiCES [93]	denoising, target speaker separation	unfurnished room		
SiSEC2008 [94]	speaker separation, denoising	lab, living room, cafeteria	X	
SiSEC2010 [95]	speaker separation, denoising	lab, living room, cafeteria	X	

Table 3.2: Comparison of single and multi-speaker corpora on environment and speakers.

times. The training is done on the Libra Free speech corpus and the testing on the TIMIT corpus. They found that mask estimation compensates for the destructive interference, however, there is a risk of amplifying noise sources. These two bidirectional long short-term memory (biLSTM) techniques are not the only ones both Luo et al. [98] and Zhang et al. [99] use biLSTMs. Where Zhang et al. [99] creates an IRM, Luo et al. [98] directly separates the reverberation from the original signal using a time-domain audio separation network (TASNET). Instead of converting the signal to the frequency domain, Luo et al. [98] keep the signal in the time domain and use a 1D convolution auto-encoder to rescale the input before it is processed by a biLSTM network. To revert the signal to the original scale a 1D deconvolutional decoder is used. They simulate the reverberated speech from the WSJ0 corpus and use three different rooms (small, medium and large) with different RT_{60}s (0.3, 0.6 and 0.9). This network is also used to do speaker separation. On the other hand, Zhang et al. [99] use the log-magnitude spectrum of the microphone and echo signal as input to the network. Their network has been trained on the TIMIT corpus with a reverberation time of 0.2 seconds in a room of 4m $\times$ 4m $\times$ 3m (L $\times$ W $\times$ H).

Gomez et al. [100, 101] describe the use of a hybrid multi- and single channel model for the dereverberation and localisation of speakers. Their technique uses the input of eight microphones located in the robot's head. First, the technique removes noise from the speech signal and transforms it to a single channel model of which the reverberation is removed. The technique is applied to two rooms of different size with different distances between the speaker and the robot. Takeda et al. [102] use the ASIMO which has eight microphones, similar to that of Gomez et al. [100]. The technique used by Takeda et al. is frequency domain independent component analysis (FD-ICA) and tries to separate the clean speech from the reverberation. This technique is tested in two different rooms where the speaker was 1.5 metres away from the robot.

The technique described by Li et al. [103] is another one using long short-term memory (LSTM) layers. However, these are now used in the form of a generative adversarial network (GAN). Unlike Ernst et al. [104], the approach of Li et al. [103] uses a single channel as input to the network which consists of five convolution layers followed by two biLSTM and a neural network layer. For training and testing they

simulate five different rooms (3m × 3m × 3m, 6m × 6m × 4m, 9m × 9m × 5m, 4m × 5m × 3m, 10m × 12m × 6m), three are used for training and two for testing. For the audio, utterances of the WSJ0 are used. Ernst et al. [104] use the convolution neural network (CNN), instead of biLSTMs for dereverberation and a GAN for the improvement of the signal. They base their architecture on the u-net image-2-image architecture and combine this with a GAN. As input to the network, they use the REVERB corpus instead of the WSJ0 corpus. Unlike Ernst et al. [104] a CNN can be used as a stand alone measure to remove the dereverberation as presented by Guzewich et al. [105]. In this case, a CNN is trained on two different corpora (TIMIT and multiroom8) for speech dereverberation. The network contains nine convolution layers and two neural network layers. The accuracy was measured in terms of speech enhancement and speaker verification instead of the measurements described at the start of this section. CNNs can also be combined with self attention as presented by Zhao et al. [66]. They use self attention as a preprocessing step that recognises the direct signal from information it has seen in the past. This is used as input to a temporal convolutional network (TCN) which will learn the mapping to the clean speech spectrum. This technique is trained on the WSJ0 and REVERB corpus separately in three rooms and using two distances between the source and microphone (0.5 and 2 metres).

A denoising autoencoder (dAE) is a popular technique for removing noise from a signal (see Section 2.4.3). Kodrasi et al. [106] see reverberation as noise and concentrate on suppressing late reverberation in single channel recordings. They combine this technique with a power spectral density estimator. The reverberation times varied from 0.2 seconds to 2 seconds and was applied to the TIMIT corpus. They extended this by also using utterances from the HINT database. Their approach gives a higher power spectral density (PSD) estimation accuracy and similar dereverberation results as state-of-the-art techniques, where the latter requires prior knowledge about the reverberation times.

Wang et al. [107] use a deep neural network for this problem as input the sub-band inter-sensor ratios which are effective DoA cues. During the training phase, they also use a voice activity detection network to be able to make smarter choices about which frames to choose for the DoA estimation. As input to the network, the

TIMIT corpus is used and is simulated in a room of 8m × 6m × 3m (L × W × H). The testing scenario is conducted in two different rooms of 6m × 7m × 4m for the simulation and 8.5m × 3m × 5m for the real situation.

On the other hand, a deep neural network can be combined with weighted prediction error (WPE) and beamforming as presented by Drude et al. [108]. Beamforming helps the WPE to determine the tail of the reverberation and to remove this from the affected signal. They build three different configurations of the network one where the beamforming is followed by WPE, one where WPE is followed by beamforming and a third where WPE and beamforming happen simultaneously. These configurations are trained on the WSJ corpus with VoiceHome impulse responses. They found that WPE, followed by beamforming gives the best result. This is not the only technique that combines WPE with another technique. Mosayyebpour et al. [109] describe an online dereverberation method based on a neural network and compare this with a Kalman filter and WPE. They use MFCCs as the input to the deep neural network (DNN) which produces the source activity posteriors which are combined with the variance to build a filter. The official name for WPE used in the previous papers is Normalized Delayed Linear Prediction (NDLP) and is introduced by Nakatani et al. [26]. This technique concentrates on late reverberation and removes this by modelling two processes: the capture process which is modelled using NDLP and the sources process which is assumed to be a Gaussian process. The extension with variance normalisation allows the technique to improve the result using short segments of the input. This technique was extended by Parchami et al. [110, 111] to linear prediction WPE. They used the TIMIT corpus as input for both techniques and concentrate mainly on early reverberation. WPE can also be optimised with the outcome of the minimum variance distortionless response (MVDR) beamformer instead of the two algorithms working separately; this is introduced by Boeddeker et al. [112]. This simplifies the algorithm because the spatial covariance matrix is calculated by the MVDR algorithm instead of the WPE algorithm. More successful is multiple inputs multiple outputs WPE by Yoshioka et al. [33]. This is an expansion of WPE with the main difference that the same number of channels coming in are also produced by the algorithm. Making the technique ideal as a preprocessing technique for speaker separation.

Apart from combining the MVDR with WPE, it can be combined with multi-channel linear prediction and a Kalman filter [113] or QR decomposition [114], for the removal of reverberation. The former is introduced by Hashemgeloogerdi et al. [113]. Their technique removes the need for a blocking matrix by using constrained minimization to minimise the output signal. To update the parameters they use a filter error correlation matrix that measures the difference between the predicted signal and the ground truth. The method is applied to the REVERB corpus and tested in three rooms; each with two settings (far and near). On the other hand, Cohen et al. [114] combine QR decomposition with MVDR for echo cancellation and noise reduction. The method consists of two stages: the first stage is an IQRD relative least squares echo cancellation followed by a weighted MVDR in the second stage. The technique is based on a multi-channel approach where the cross correlation between the reference and the microphone signal determines the echo path. The echo is removed by using a Wiener filter. Cohen et al. [114] are using two methods to test this technique by using a speaker and mobile phone in scenario 1 and a smart speaker reading an audiobook and a talker in scenario 2. Gannot et al. [115] describe QR decomposition as being a suboptimal solution, instead, they use a null subspace and total least squares for FRF estimation. Finally, they use a decimated subband method for reducing complexity and increasing robustness. The noise is drawn from the NOISEX database and the audio signals from TIMIT.

Mohanan et al. [116] use a convolutive nonnegative matrix factorisation (NMF) model for doing single channel speech dereverberation instead of deep learning which was discussed in the previous papers. Their model assumes that there are two matrices, one for the original speech and one for the reverberation. The speech matrix consists of one dictionary and one activation matrix. The reverberation matrix is separated in a gain and a frequency matrix. The technique requires prior knowledge about the room and the source to microphone distance. They apply this technique to the TIMIT corpus. A technique similar to NMF is nonnegative tensor factorisation (NTF), Wager et al. [117] use this with multiplicative update rules for speech dereverberation. They convolve a second matrix with the mixture's estimate to replicate the reverberation. For testing, their technique they simulate three rooms with an impulse response of 0.6, 1.2 and 1.6. These impulse responses are convolved

with clean data from TIMIT.

Spectral subtraction [118, 119] is another technique used for dereverberation in the subbands of the signal. The technique works by estimating the power spectral density of the reverberation and multiplying this with the original signal.

Attics et al. [120] use a variational speech enhancement algorithm. This algorithm is designed to do denoising and dereverberation. The algorithm tries to minimise the distance between the approximated and actual conditions. For the reconstruction of the affected signal, a Wiener filter is used. The algorithm is tested on the WSJ corpus.

Carini et al. [121] measure the room impulse response by assuming modelling of the acoustic path as a Legendre nonlinear filter with perfect periodic sequences (PPs). They use Wiener filters to estimate the coefficients from the PPs. To build the PPs for the Wiener filters, all the joint movement need to be estimated, a period which involves measuring the first order kernel with ideal Gaussian noise. This can be solved using the Newton-Raphson method. With this, the PPs and the Wiener filters can be used for measuring the RIR.

Instead of the STFT, also different signal transforms can be used for the dereverberation of the signal, for example, the cepstrum [122], wavelet transform [123], temporal envelope filtering [124] or modulated complex lapped transform [125]. Subramaniam et al. [122] use the cepstrum as input to a deconvolution algorithm for dereverberating the signal. They are reconstructing two channels and first compute the log spectral of the observations. Griebel et al. [123] use wavelet based extrema clustering, instead of the STFT, to decompose the linear predictive coding residuals. The wavelet extreme of quadratic spline wavelets proves to be appropriate indicators of discontinuities in a signal. These extremes are well clustered across all channels, this captures the underlying impulsive structure of the original non-reverberant speech. To test this, a reverberation time of 200ms was used. Avendano et al. [124] use temporal envelope filtering for the removal of reverberation. This filter works by minimising the Euclidean distance between the affected and the clean speech. It filters the magnitude spectrum of speech. Gillespie et al. [125] apply maximum-kurtosis as subband adaptive filtering to remove reverberation. They use a modulated complex lapped transform which has subband filters that maximise the

kurtosis of the linear prediction residual. This technique improves when multiple microphones are used. They are using a mouth simulator to create speech sounds for testing the technique. The signals are convolved with a room impulse response to add reverberation.

technique	algorithm	room size	RT_{60}	train corpus	test corpus	CD	SDR	SNR	PESQ
Deep learning	GCC-PHAT biLSTM [96]			TIMIT	TIMIT				
Deep learning	MMSE-biLSTM [97]			Libri-speech	TIMIT	0.9			
Deep learning	TASNET-biLSTM [98]		0.3, 0.6, 0.9	WSJ0	WSJ0		11.1		
Deep learning	biLSTM [99]	4m x 4m x 3m	0.2	TIMIT	TIMIT				2.62
Deep learning	GAT [103]	3m x 3m x 3m, 6m x 6m x 4m, 9m x 9m x 5m, 4m x 5m x 3m, 10m x 12m x 6m	0.6	WSJ0	WSJ0				2.76
Deep learning	CNN+GAN [104]		0.5	REVERB	REVERB	3.19			
Deep learning	CNN [105]			TIMIT, multiroom8	TIMIT, multiroom8				
Deep learning	TCN [66]	5.6m x 3.8m x 2.5m, 6.3m x 4.9m x 2.6m, 6.2m x 6.7m x 3.0m		WSJ0, REVERB	WSJ0, REVERB	2.20			2.51 2.58

technique	algorithm	room size	RT_{60}	train corpus	test corpus	CD	SDR	SNR	PESQ
Deep learning	dAE [106]			TIMIT, HINT	TIMIT, HINT	-0.18			
Deep learning	DNN-SDD [107]	8m x 6m x 3m, 6m x 7m x 4m, 8.5m x 3m x 5m		TIMIT	TIMIT				
Deep learning	WPE-DNN [108]			CHiME3	CHiME3				
Deep learning	WPE-Kalman-DNN [109]			TIMIT	TIMIT				
correlation	NDLP [26]	3.7m x 5.5m		TIMIT	TIMIT	-0.7			
correlation	LP-WPE [110]	3.7m x 5.5m		TIMIT	TIMIT	-0.85			
correlation	WPE-wMVDR [112]			CHiME3 REVERB	CHiME3 REVERB				
beamforming	MVDR [113]			REVERB	REVERB	3.32			2.46
beamforming	QRD-MVDR [114]			speech	speech			36.7	
	total least squares [115]			speech	speech				
NMF	NMF [116]		0.7	TIMIT	TIMIT	4.48			
NTF	NTF [117]		0.6, 1.2, 1.6	TIMIT	TIMIT			-1.64	

technique	algorithm	room size	RT_{60}	train corpus	test corpus	CD	SDR	SNR	PESQ
	specsub [118]		1.7	speech	speech	0.06		1.2	
	VSE [120]			WSJ0	WSJ0			6	
	single channel [100]	5.5m x 4.8m	0.24, 0.64	JNAS	JNAS				
	FD-ICA [102]		0.24, 0.67	human speech	human speech				
	single channel [101]		0.94	JNAS	JNAS				

Table 3.3: Comparison between different dereverberation techniques

Many of the techniques concentrate on short reverberation times and are not tested in reverberation times over 1 second. Apart from this, many of the described experiments do not describe the environment or the RT_{60} time but only describe the corpus used. This makes them difficult to reproduce. When trying to compare the techniques some papers state the improvement of the different measurements but fail to describe which technique this is compared against or that technique also only describes the improvement. This makes it hard to compare against others. Currently, MIMO WPE [33] is considered to be the standard to beat. In Chapter 5, both NDLP and MIMO-WPE are tested on the TIMIT corpus to build comparable results to the ones described in Table 3.3.

3.3 Single channel speaker separation

Blind source separation ([126, 127]) is a technique in which one is trying to separate the audio sources with only the knowledge of how many sources and sensors there are. The problem is separated into two different categories: the overdetermined case with a higher or equal number of sensors to sources ([128]), and the underdetermined case with a lower or equal number of sensors to sources. The underdetermined case includes single channel or monaural speech separation. These cases have been extensively studied producing methods that show promising results. For example, independent component analysis (ICA) is used for both cases ([129]), and for the underdetermined category, directional Laplacian distribution ([130]), Laplacian mixture models (LMM, [131, 132]) and hidden Markov models (HMM, [133]) are used with success. These techniques can be compared with each other using the measurement described by Vincent et al. [1]. An overview of this and the corpora used is given in Table 3.4.

The ICA methods are either fixed point or use gradient descent to divide the sources. The fixed point algorithm converges faster than gradient descent but also uses more computing power ([134]). Smaragdis et al. [135] applied the natural gradient ICA algorithm with a complex non-linear activation function. One downside of the ICA algorithm is that when it is combined with the time-difference of arrival (TDoA), it cannot be used to solve the permutation problem for high frequencies.

This problem arises when the frequency exceeds the spatial aliasing limit and the signal becomes indistinguishable. Davies [136] introduced a time-frequency model to solve the permutation problem by adding a time dependent term to the frequency model of the separated sources. It is possible to create a method using both fixed point and gradient descent as shown by Hyvarinen et al. [137]. Mitianoudis [138] used this method, described by Hyvarinen et al. [137], but replaced the gradient descent algorithm with another fixed point algorithm that works by prewhitening the STFT coefficient of the mixtures and storing these prewhitened matrices. The algorithm uses a time-varying parameter aiming to model the audio signals more effectively.

The LMM [131] is fitted to the phase difference between the two sensors and is used to perform separation using either a soft or hard (winner takes all) threshold. The directional Laplacian distribution [130] is extended by using finite impulse response (FIR) filters to model the impulse response between the different sound sources and the microphones to remove the influence of the room acoustics. HMMs [133] are used to model the spatial diversity of the sources in the mixing matrix and the more structured source priors.

3.3.1 Non-negative matrix factorisation

NMF has been applied to topic modelling [139] where it determines which topics best describe a document. Facial feature extraction is another topic on which NMF has been applied [140], here it dissects a photo of a face into a number of different features. This can be applied to face recognition where the features can be used for comparing against a new image.

Separating near field speech (less than 5 metres) and premixed audio files [7, 39, 141] is another topic to which NMF has been applied. With NMF, it is very important to choose the optimal cost function; the Kullback-Leibler (KL) cost function is the most popular one. Another cost function is the Itakura-Saito (IS) divergence, which has been successfully used for music analysis to separate different instruments in an audio track [141]. The main difference between the IS divergence and the KL divergence is that the former is scale invariant. This means that in the cost function the same relative weight is given to small and large coefficients and results in

that the factorisation does not only rely on the largest coefficients and has a higher precision in the estimation of low-power components.

SAGE-NMF [142] is a different version using the expectation maximization (EM) algorithm instead of multiplicative updates (MU). In this case, the EM algorithm is modified to the space-alternating generalized EM. Magron et al. [143] apply this to the GRID corpus. An EM approach converges quicker but is also computationally more expensive than MU.

Tensor factorisation multiplies matrices with more than two dimensions together. This is a natural extension of NMF. Stein [7] and Ikeshita [144] both explore this in different ways. Stein describes using NTF together with direction of arrival whereas Ikeshita uses positive semidefinite tensor factorisation to do source separation. Ikeshita works under the assumption that the sources are independent of each other in time. The technique uses the same number of microphones as sources. The technique is similar to NMF but according to Ikeshita outperforms NMF in the monaural source separation tasks. For testing the algorithm they use the SiSEC2008 corpus and concentrate on stereo mixtures.

3.3.2 Deep learning

There are two main drawbacks with using NMF for speaker separation: it is trained on each mixture individually to create the best separation mask and the non-negative linear combination of the trained vectors does not necessarily result in the best mask. Another approach is to use deep learning to build one mask that can be applied to the environment in which the corpus was recorded. Deep learning needs to be used on bigger corpora in order to build a mask that best represents the environmental noise and reverberation. When the algorithm has converged to a solution it will run in almost real-time on new unseen data. The downside of these masks is that when there are major changes to the environment then the algorithm needs to be retrained. However, for retraining the algorithm, a smaller dataset can be used and this is best described as transfer learning, where the original model can still be used but is now adapted to a new situation.

Deep learning is successfully applied to speech recognition providing machines with almost human-level speech recognition. Deep neural networks (DNN) is one

of the methods for building a T-F mask [145, 146]. A DNN consists of multiple dense (fully connected) layers which generate a Time-Frequency mask by training the network to recognize two files given a single file. This network can be combined with principal component analysis to learn the orthogonal component of the interferer [147]. DNNs can also be trained to perform three separate tasks (denoising, dereverberation and speaker separation) as shown by Sun et al. [148]. They train a DNN on the mixture, individual speech signals and the noise to build these masks and to create a successful separation. To simulate the noise in the recordings they make use of the NOISEX corpus [149].

Another option is convolution neural networks (CNN) which have successfully been applied to object tracking [150], segmentation [151–153] and music classification [154]. These networks are versatile - the same network structure is applied to voice reconstruction [155] and image generation [156]. Also, CNNs can be used for localising and discriminating between sound sources [157]. In this particular case, the sound is played through loudspeakers and recorded by the microphone array of the pepper robot.

Recurrent neural networks provide a different way of modelling speech and are used for speech recognition [158] and in the form of LSTMs they can be used for multi-task learning [159] or for building a deep clustering network [160]. Xu et al. [159] build an LSTM network to create separation masks and estimate the T-F label as a subtask. As extra input, they use a shifted delta coefficient which ensures the spectral continuation of a speaker. Aihara et al. [160] use LSTMs in combination with linear layers to create a deep clustering network for speaker separation. They are building a complex mask of the interfering sounds which can be removed from the input signal. Another technique is a variational Autoencoder (vAE), see Pandy et al. [161], which can be used for separating speakers in a mixture. As input to this network, the magnitude spectrogram of the speaker in the TIMIT corpus is used.

More recently generative adversarial networks (GAN) were developed to generate new images and audio segments based on existing ones. These networks can also be used for speaker separation. Chen et al. [162] combine a speech enhancement GAN (SEGAN) with permutation invariant training (PIT) to separate sources in a single channel mixture. PIT sees the separation problem as a supervised multi-

class segregation problem. In this case, the generator builds the masks based on the mixture and gives clean separated speech as output. The discriminator gets separated speech as input either from the generator or from the corpus.

Phasenet [163] expects as input the STFT of the mixture. It assumes that the phase of the mixture is mainly affected by the source which is dominant in the mixture at that time step. The DNN tries to minimise the loss between the phase of the speakers it has been trained on and the predicted phase in the mixture. It has been trained on speech signals from the WSJ0 corpus. TASnet [98] on the other hand, works in the time domain taking the raw waveform and using this for the separation of speakers. The input is passed through an encoder containing a convolution layer followed by a separation network with multiple LSTMs and a fully connected layer. After the separation, the signal will be decoded back to a waveform. FurcaPy is another network that uses the raw waveform just like TASnet. The FurcaPy network [164] uses dilated convolutions for speaker separation. The network uses the same underlying principle as Wavenet which combines the dilated convolutions with residual connections creating a temporal convolutional network. This is to prevent input information from being lost in the convolutions.

technique	**algorithm**	**corpus**	**SAR**	**SDR**	**SIR**
ICA	Laplacian distribution [130]	music	4.58	14.61	5.78
ICA	LMM_EM_Hard [131]	music		2.32	
ICA	LMM_EM_Soft [131]	music		1.18	
ICA	MoWL_EM_hard [132]	Groove (music)		0.52	
ICA	MoL_EM_hard [132]	Groove (music)		0.2	
ICA	SCICA (alg. I) [136]	unknown			22.2
ICA	SCICA (alg. II) [136]	unknown			16.33
HMM	Spatial div all sources [133]	unknown	4.33		
HMM	Spatial div closest source [133]	unknown	26.67		9.17
HMM	Spatial div 1 or 2 closest sources [133]	unknown	22.33		11.2

technique	algorithm	corpus	SAR	SDR	SIR
HMM	Souce priors [133]	unknown	10.33		6.47
HMM	spatial div + Bernoulli source priors [133]	unknown	26.33		64.93
HMM	spatial div + Markov source priors [133]	unknown	28		14.13
HMM	spatial div + source priors oracle [133]	unknown	29.67		15.93
HMM	oracle [133]	unknown	47.33		25.43
NMF	NMF-IS [141]	music			
NMF	SAGENMF [142]	unknown			
NMF	ML-MUR [143]	GRID	6.7	5.7	13.5
NFM	SAGE-MUR [143]	GRID	4.0	1.6	12.1
NMF	EM-MUR [143]	GRID	6.8	5.8	13.4
NTF	NTF [144]	SiSEC2008			
Deep Learning	DNN [145]	WSJ			7.8
Deep Learning	DNN [146]	unknown	6	6	21
Deep Learning	DNN [147]	TIMIT, TSP	9.89	6.39	9.72
Deep Learning	DNN [148]	TIMIT			
Deep Learning	MTNN [157]	human speech			
Deep Learning	LSTM [159]	WJS0		10.5	
Deep Learning	VAE [161]	TIMIT	6.92	6.73	9.02
Deep Learning	SEGAN [162]	WSJ0			
Deep Learning	PhaseNet [163]	WJS0		13.83	
Deep Learning	TASNET [98]	WSJ0		11.1	
Deep Learning	FurcaPy [164]	WSJ0		18.4	

Table 3.4: Comparison between single channel speech separation techniques

When comparing the results of the different techniques (see Table 3.4), deep learning does outperform the ICA and NMF techniques on SDR. However, in terms of the number of artefacts and interference deep learning has difficulties in surpassing the HMM techniques. The latter techniques have not presented an SDR

measurement for comparison. When concentrating on just NMF and DNNs, then there is an overlap in performance showing that an unsupervised technique (NMF) performs as well as a supervised technique (DNN)

3.4 Multichannel speaker separation

Multichannel speaker separation uses the information of two or more microphones to determine where the speakers are by looking at when the signal arrives at a microphone. This allows the algorithm to calculate the difference in time of arrival of the signal between the microphones and with knowledge of the location of the microphones, it is able to determine the direction of arrival. The location information helps when sources are spread through a room or moving around, but is less valuable when the sources are behind each other (or overlap) from a microphone's perspective.

Beamforming (minimum variance distortionless response) gives location information about the sources which can be used to improve source separation that can be combined with speaker activity detection (SAD). Ceolini et al. [165] use this algorithm to separate the speakers in a mixture created from audiobook recordings. The algorithm first applies SAD, which is followed by determining the steering vectors. This information is given to the minimum variance beamformer which is able to separate the speakers. For the recordings, a microphone array with eight microphones is used. These have been scattered around the room.

Ito et al. [166] use a complex Gaussian mixture model with non-sparse noise model for source separation. To improve computation they use a diagonal based EM algorithm for the updates. This reduces the matrix inversion, multiplication and determinant computation to scalar operations of the diagonal entries. However, this requires computation of the generalized eigenvalues. As input to the algorithm recordings from the SiSEC2010 corpus are used.

Within robotics, it is important to determine where the speaker is located so that the robot can face them allowing for a more natural interaction. Nakadai et al. [167] describe an active direction pass filter that is able to locate the speaker using microphones located on the sides of the robot's head. This technique allows the

robot to locate, track and separate the speakers. This technique is an improvement on the head related transfer function (HRTF) which according to Nakadai et al. [167] can only be used in anechoic chambers. Keyrouz et al. [168] apply these HRTFs to speaker separation and use multiple HRTFs to determine where the speaker is located. For their technique, it is needed to know how many speakers there are in the room.

These techniques can be compared with each other using the same measurements as in Section 3.3 described by Vincent et al. [1]. An overview of this and the corpora used is given in Table 3.5.

3.4.1 Non-negative matrix factorisation

Multichannel NMF adds a form of directionality to NMF [7, 38–40]. Intensity information about the different sources is conveyed to the algorithm. There are two ways of adding direction information; one is in the form of another matrix thus creating non-negative tensor factorisation instead of NMF. The other is by multiplying the input with the direction information. The latter will not work for larger microphone arrays because of phase wrapping, this happens when the input is constrained to its principal value.

Combining the knowledge of the possible source locations with information from multiple microphones allows the algorithm to separate the sources. However, this assumes that the location of the two sources is differentiable, which on a plane is not always the case when sources move around. For example, when the sources are directly behind each other this does not show up on a plane, only in a 3D environment.

With multichannel NMF, cost functions are combined with different techniques, among others time-difference of arrival (TDoA) [7, 39] and convolution to improve the accuracy of NMF and to use the temporal information encoded in the examples. TDoA is used to describe the spectral covariance matrix of a source between microphones. This, combined with the NMF approximation of the mixture, gives a source per microphone. Another way of calculating the SCM is to calculate the covariance between the signal before and after it is affected by reverberation. The before (or "clean") signal can be approximated with NMF in a joint fashion with

calculating the mixture. The first step is to calculate how the sources are affected by the reverberation and additional noise (i.e. calculating the frequency response function or the room impulse response). This is followed by an approximation of the mixture including reverberation. For demixing, a Wiener filter is used in addition to the frequency response function (FRF) for the removal of the reverberation. A different way of calculating the SCM is by determining the correlation between the microphones [38, 40] and combining these to form the FRF. This is updated every iteration to create the best fitting FRF for the separation.

Yoshii et al. [169] use independent low-rank tensor factorisation. This is based on NMF and independent vector analysis. The latter solves the permutation problem that ICA has and assumes that the source spectra follow multivariate probability distributions. This technique is very similar to IS-NMF and uses the multi-channel IS cost function. It uses EM update rules for determining the different matrices. They use piano tones for training and testing this technique but the technique has not been used on speech.

3.4.2 Deep Learning

As with single channel source separation, we see similar techniques used in multichannel deep learning, as well as combinations between deep learning and NMF [170]. The DNN model estimates the variance matrix between sources and tries to build a demixing matrix using this by estimating the source spectrogram and updating the demixing matrix simultaneously. This method is supervised because this network needs the original mixture matrix for the NMF part of the algorithm as well as ground truth files for the DNN part. The DNN model can also be used on its own to create a time-frequency mask of the different speakers [171].

Instead of using a DNN to create a mask, RNNs are also used for this [172]. These RNNs have shared weights and are trained to output masks for all the speakers in the mixture, which are fed into minimum variance distortionless response beamformers for each speaker. Instead of using the audio signal as input, a mask can be built using the interchannel phase difference (IPD). Wang et al. [173] use this input for a permutation invariant training (PIT) network, passing it through a three layer network with two biLSTM layers and a neural network layer.

Another technique for building ratio masks uses a gated residual network [174]. A GRN is based on dilated convolutions which expand the receptive fields. It requires the use of the magnitude spectrogram and captures the patterns along the frequency direction. The first stage is to apply denoising and after that, it separates the speakers. The network is trained on the WSJ corpus with added noise from the NOISEX-92 corpus [149].

Autoencoders can use the raw input signal in a similar way as TASnet does to separate the speakers. A multi resolution convolutional autoencoder [175] passes the input through different convolution layers to extract features from the audio signals. These features are combined and separated in the following layers to create the separated signals.

technique	**algorithm**	**max. speaker-to-mic**	**corpus**	**SAR**	**SDR**	**SIR**
beamforming	MVBF-SAD [165]	20cm	audiobook	-1.40	-1.46	21.60
	ADP[167]	50 cm	newspaper			
	HRTF[168]		human speech			21
GMM	GMM [166]		SiSEC2010			
NMF	NMF-TDoA [39]	1.5m	audiobook	13.1	5.6	6.8
NMF	NMF-DoA [7]		TIMIT	10.4	3.0	6.8
NMF	CNMF-EM [38]	1.2m	SiSEC2008		12.3	
NMF	CNMF-MU [38]	1.2m	SiSEC2008		4.4	
NTF	NTF-DoA [7]		TIMIT	14.2	9.6	14.6
NTF	NTF-SCM [40]		unknown			
Deep Learning/NMF	DeepNMF [170]		SiSEC2016			
Deep Learning	DNN [171]	<1m	CHiME3	18.23	13.25	15.58
Deep Learning	PIT-RNN [172]	<1m	WSJ0		10.3	
Deep Learning	PIT-LSTM [173]	<1m	CHiME3			
Deep Learning	GRN [174]	<1m	WSJ0			
Deep Learning	MR-CAE [175]		SISEC-2016-MUS	5.89	4.71	8.43

Table 3.5: Comparison between multi channel speech separation techniques

The results of the multichannel algorithms are very similar to the ones of the single channel algorithms. The DNN performs better than the NMF algorithms but the difference between the techniques is not large.

3.5 Conclusion

In this chapter, several different corpora were presented. Many of these are used for speaker separation or dereverberation. However, these corpora lack reverberant and noisy speech and need to be run in a simulated environment or convolved with a preselected room impulse response to be able to test this. The advantage of this is that there is always a clean signal present and the corpus can easily be convolved with different environments. On the other hand, the recorded sound does not have a natural interaction with the environment resulting in additional artefacts created by the convolution. For the noise, often the NOISEX-92 corpus is used to create noisy speech. How the files are combined is often not described nor the kind of noise used. In chapter 4, a corpus will be presented that address these issues.

For the dereverberation, different measurements are used to describe the performance of the algorithms. Even within these measurements, there is a difference in presentation. Often the improvement of a technique is given, but it is not described to which this is compared. In Chapter 5, the performance (not improvement) of two techniques (WPE and MIMO WPE) is presented. Another thing that makes the comparison with other techniques difficult is the lack of RT_{60} times. As will be shown in Chapter 5, the choice of RT_{60} times has a great influence on the performance of an algorithm. Currently, the MIMO-WPE method is considered to be state of the art for the real-time removal of reverberation.

The single and multichannel algorithms have a more unified approach to presenting their performance. Here the techniques can be more easily compared. However, some still lack mentioning the training and testing corpora. Many of these techniques have been tested on near-field speech (when this is described) but not on far-field speech ($>$ 5 metres). Knowing this distance is important for the multichannel algorithms. Otherwise, it is difficult to replicate the results and compare them with the results presented in Chapters 6 and 7.

Chapter 4

Acoustic camera corpus

4.1 Introduction

When addressing a robot, the speaker needs to be understood to enable the robot to execute the tasks the speaker demands from it. In general, robots are able to understand the speaker when tested in a lab setting with no other speakers around, but in real-world environments, noise, reverberation and the presence of multiple speakers make the speech recognition task more difficult. Robots need to work in these environments and in industrial environments where there is noise from heavy machinery. The noise makes it difficult for the speech recogniser to understand the speaker which does not happen in the ideal, clean lab scenario.

The majority of corpora that are currently used for speaker separation, derevereberation or denoising contain only clean speech. This can be used as a ground truth to which noise or reverberation is added. However, it is difficult to represent a particular environment by adding noise or reverberation to clean speech because the artificial noise and reverberation do not interact with each other and contain artefacts from the recording environments. The corpus introduced here has recordings containing noise and reverberation recorded in realistic environments. With 72 microphones and a maximum sample rate of 192 kHz, this microphone array is able to collect high quality recordings. The recordings in this corpus can be used to simulate any number of speakers in the recorded environment by mixing them together.

The recordings within the corpus can be used for speaker separation, tracking,

dereverberation and noise cancelling. There are two types of recordings - one with a single speaker per file and one with multiple speakers per file. The second allows for a more natural speaker separation problem where speakers do not necessarily speak at the same volume but naturally increase their volume in order to be heard over the other speakers. The first allows multiple mixtures to be created by randomly mixing speakers. For tracking speakers, there is one speaker per file, combined with the speech is ground truth information of where the speaker is located in the room. This location information is recorded with a Microsoft KinectV2 which gives the location of a speaker in 3D space.

Speech is recorded with a microphone array called the Acoustic Camera (AC). All recordings are made in one of two rooms, room A representing a realistic office and the other (room B) representing a workshop environment, both with the presence of noise and reverberation. These recordings contain people reading a short story while standing still and walking around a room. These rooms allow the data to be used for noise cancelling and dereverberation experiments.

The AC is chosen for its ability to make high quality recordings which can be downsampled to the problem space. Apart from the high recording quality, the number of microphones allow the device to produce accurate localisation of the sound source and allow for usage in a problem space where multiple microphones are needed. However, the device is not portable and cannot be mounted on a moving platform nor does is allow for continuous recording.

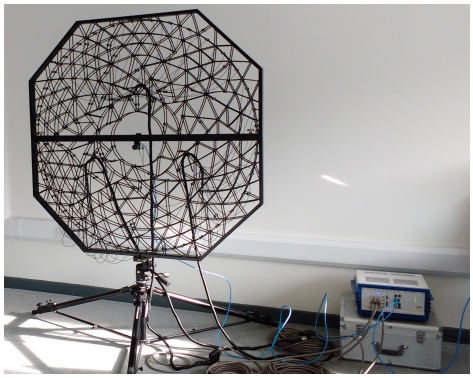
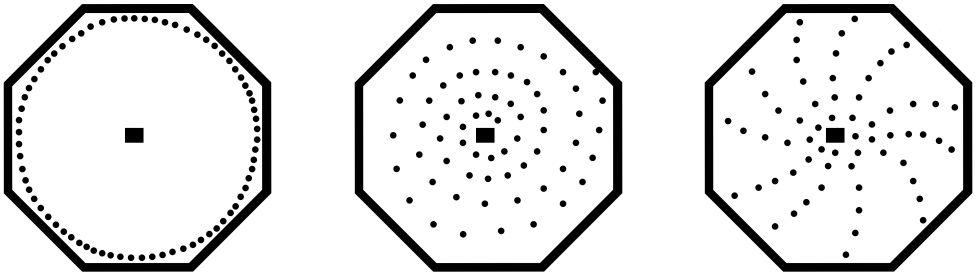


Figure 4.1: Frontal view of the acoustic camera microphone array.

4.2 Acoustic-camera

The Acoustic Camera (AC) is developed by GFAI tech as a microphone array that is able to locate sound sources. Its primary usage is in the automotive industry where analysis of the location of the noise allows designers to improve noise dampening for the reduction of engine noise. This is expanded to noise reduction in office environments where the sound sources can be localised and fault detection in engines. Compared to the Kinect v2 or MIT's LOUD [176], the AC uses 72 (see Figure 4.1) in three (ring, spiral and wheel) different 2D configurations (see Figure 4.2). LOUD uses 1020 microphones in one configuration, the Kinect v2 uses four microphones also in one configuration. Where the Kinect v2 has two cameras (RGB and depth cameras) the AC has only one camera. On the other hand, LOUD only has microphones. Another difference between these three microphone arrays is that the AC and LOUD are specifically designed for sound engineering whereas the Kinect v2 was designed for gaming. The three microphone configurations of the AC are built around a video camera which is located in the middle of each configuration. The distance between two microphones located at opposite ends of each configuration is 1 metre. Each of the 72 microphones is capable of making recordings up to a sample rate of 192kHz. The Kinect v2 and LOUD have a maximum sample rate of 16 kHz. Due to the design of the AC, it is not possible to make continuous recordings exceeding 90 seconds. After 90 seconds, the recording is stopped and written to the hard drive (which can take up to 20 minutes). The advantage of this high sampling rate in the quality and detail of the information in the recordings. This allows for using the recordings to detect the onset of speech as well as highlighting the times the speaker is breathing in or out. For this device, it means that the recordings with the device can be used not only speaker separation but also for accurate localisation (using up to 72 microphones), emotion detection, speech onset detection and speech impairment detection.

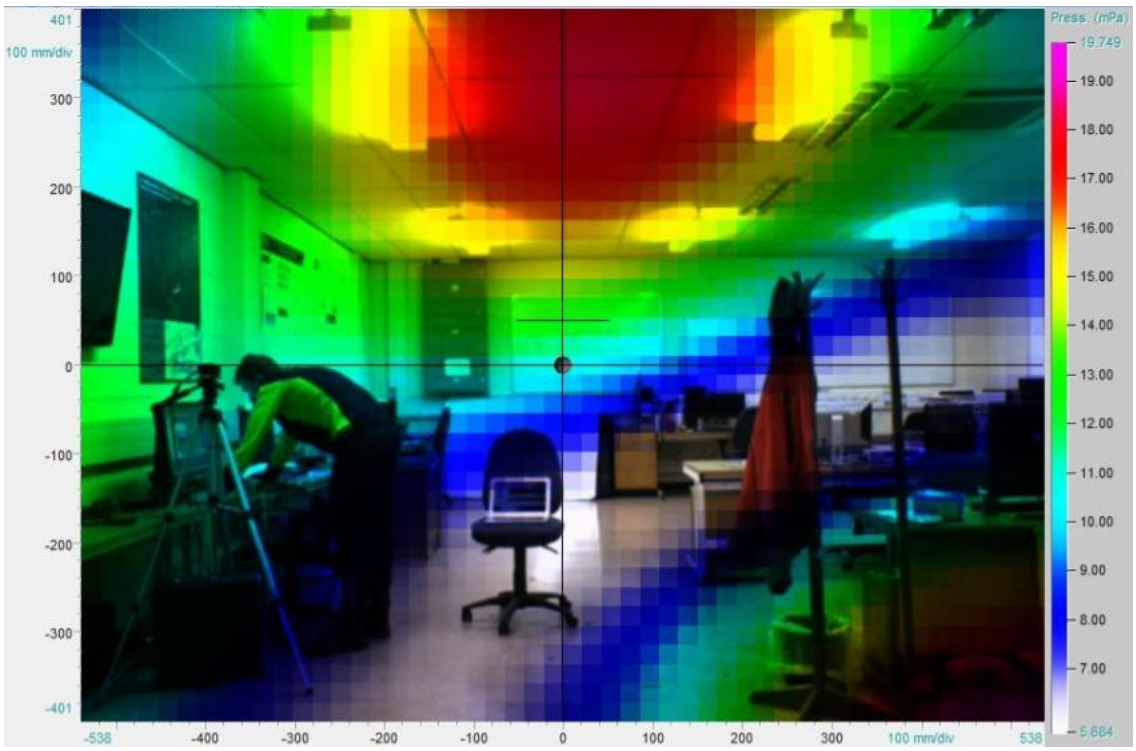
The corpora discussed in Chapter 3 have a sample rate of 16kHz. This means that for comparison purposes the sample rate of this corpus will be brought down to 16 kHz. Downsampling means that there will be a loss of information but it does not introduce the artefacts that upsampling will introduce.



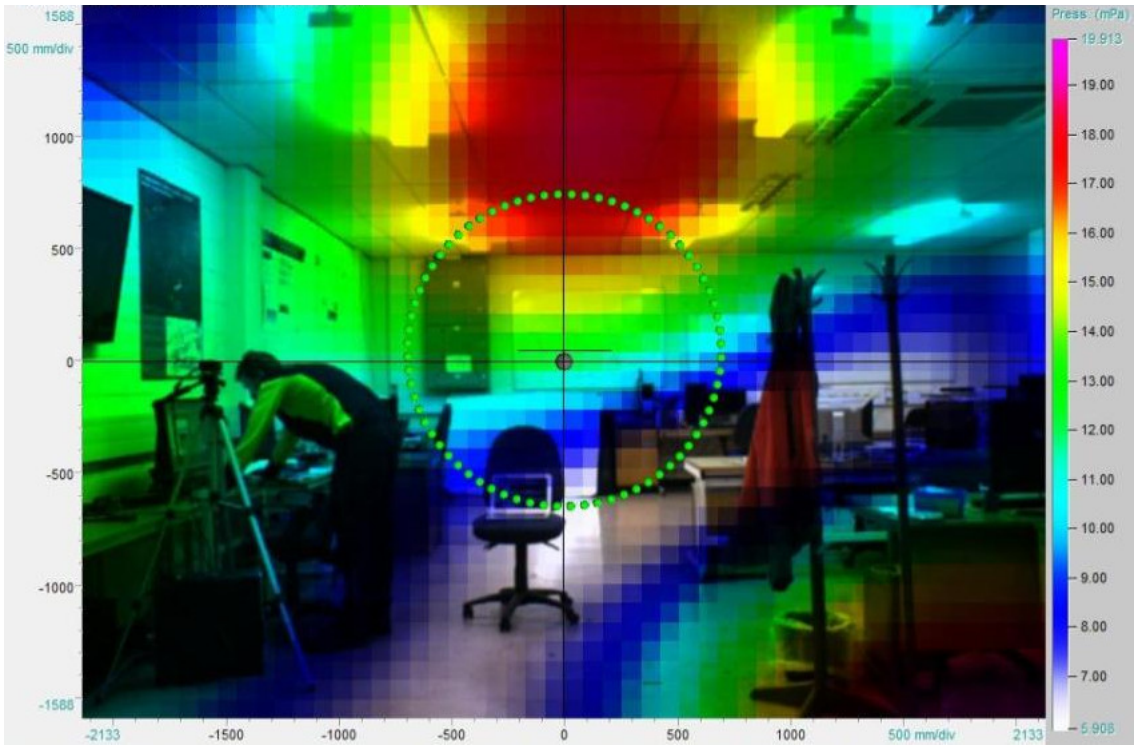
(a) ring (b) spiral (c) wheel

Figure 4.2: The location of the microphones (dots) and camera (square) in the three different configuration as seen from the front of the AC.

The AC comes with four different beam-forming algorithms (delay-and-sum [177], phase shifting [178], cross spectral matrix [179] and CLEAN [180, 181]). These algorithms are separated into two different domains: time domain and frequency domain. Using beam-forming techniques (see Section 2.6.3), the AC is able to locate and display sound sources in both 2D and 3D environments. It is important to set the correct focus (i.e. depth or distance between camera and sound source) when using the beam-forming techniques. When the microphones of the AC are arranged in a 2D configuration, the 3D localisation is not always accurate, and two issues can arise. Firstly, when the AC is using an incorrect focus, the calculated sound pressure is different compared to when the correct focus is used. From this follows an incorrect mapping of the sound source onto a 2D plane. This occurs especially when a sound source is moving, because the focus of the AC is set before the recording is started and cannot be changed during the recording. This effect is dependent on the configuration of the microphones and on the distance between the AC and the sound source. For instance, if a source (in this particular case a computer on an office chair) is placed in a room with varying distance between the source and the AC, then this effect is seen in the different configurations (see Figures 4.3, 4.4 and 4.5, where the green dots represent the location of the microphones). When the source is closer than six metres and the microphones are arranged in a ring or wheel configuration, the effect is very small (see Figures 4.3 and 4.5). However, when the microphones are in a spiral configuration or when the distance between AC and sound source is more than 6 metres, the effect is more noticeable (see Figures 4.4e and 4.4f).

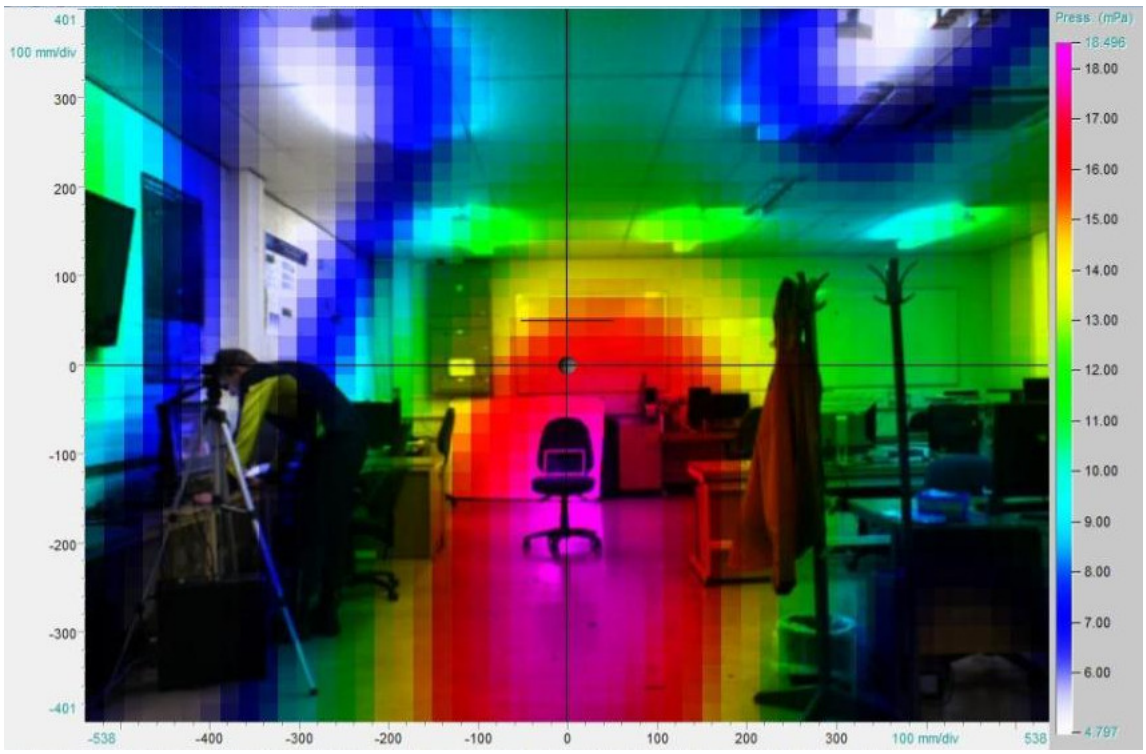


(a) Focus set to 1 metre but sound source is located at 4 metres distance

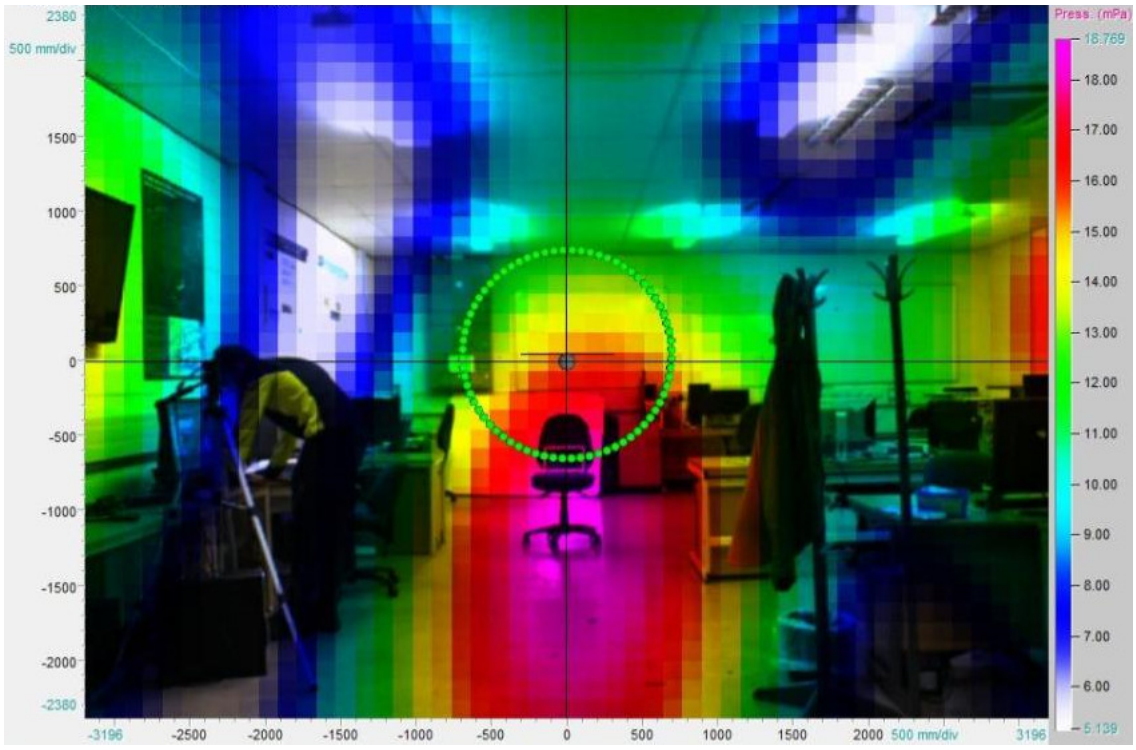


(b) Focus set to the same distance (4 metres) as the sound source

Figure 4.3: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a ring microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera.

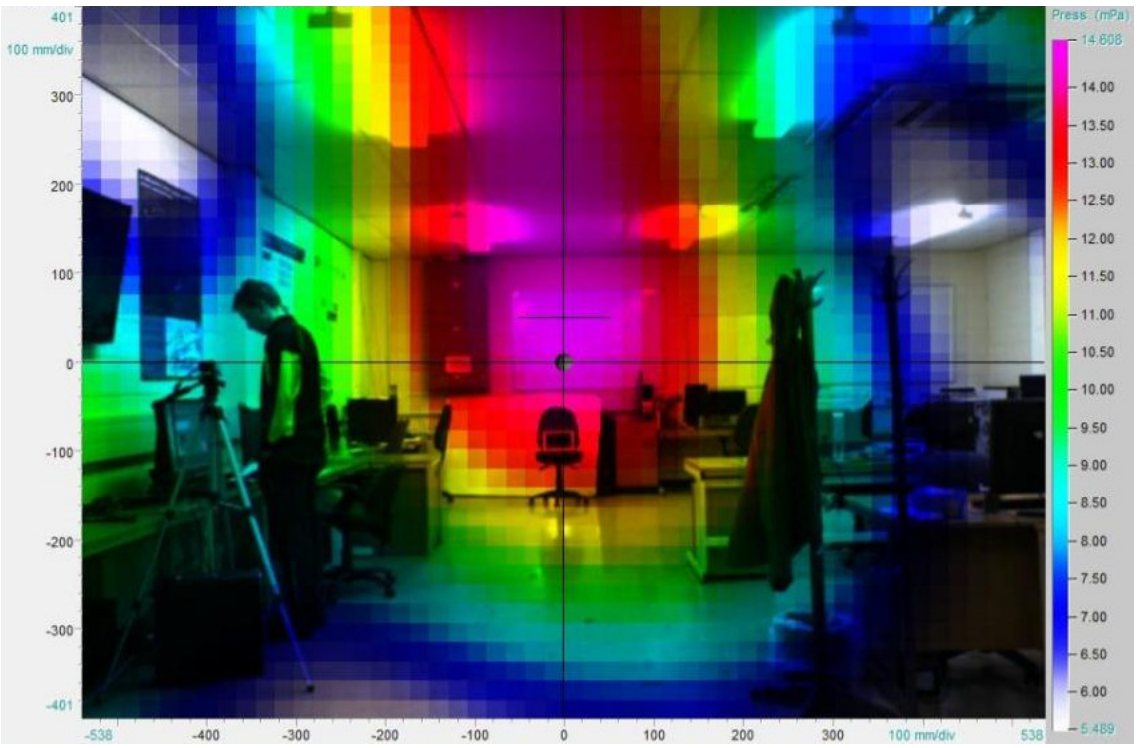


(c) Focus set to 1 metre but sound source is located at 6 metres distance

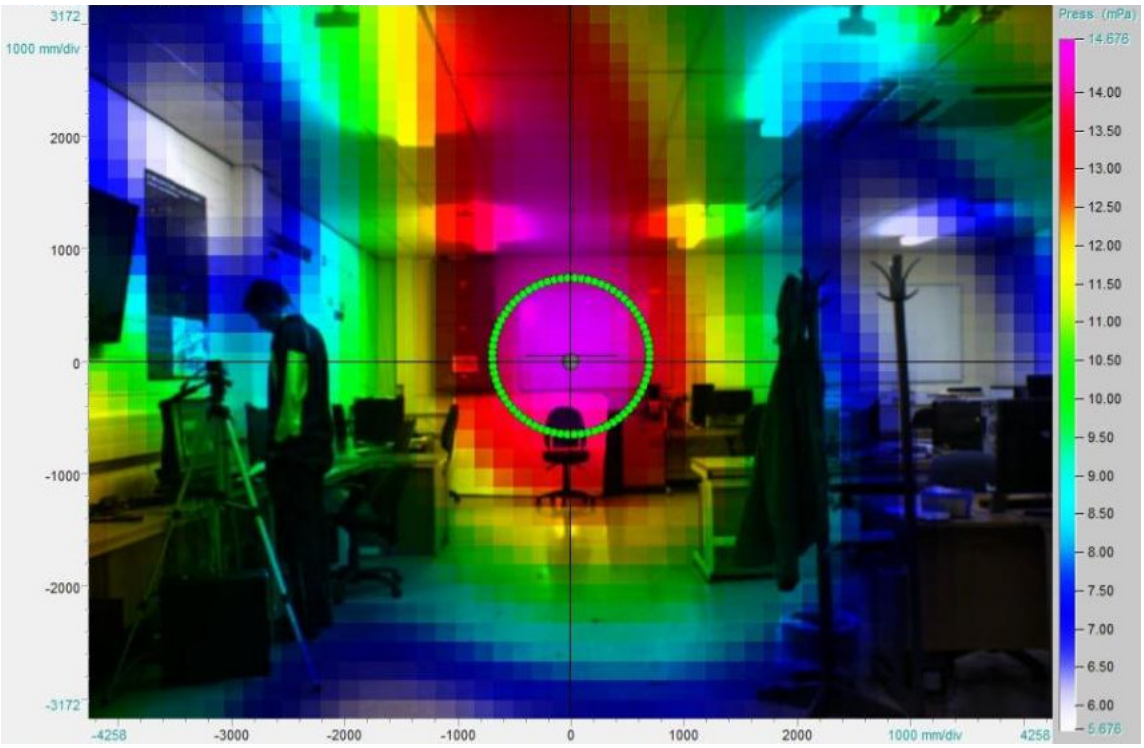


(d) Focus set to the same distance (6 metres) as the sound source

Figure 4.3: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a ring microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera (contd.).



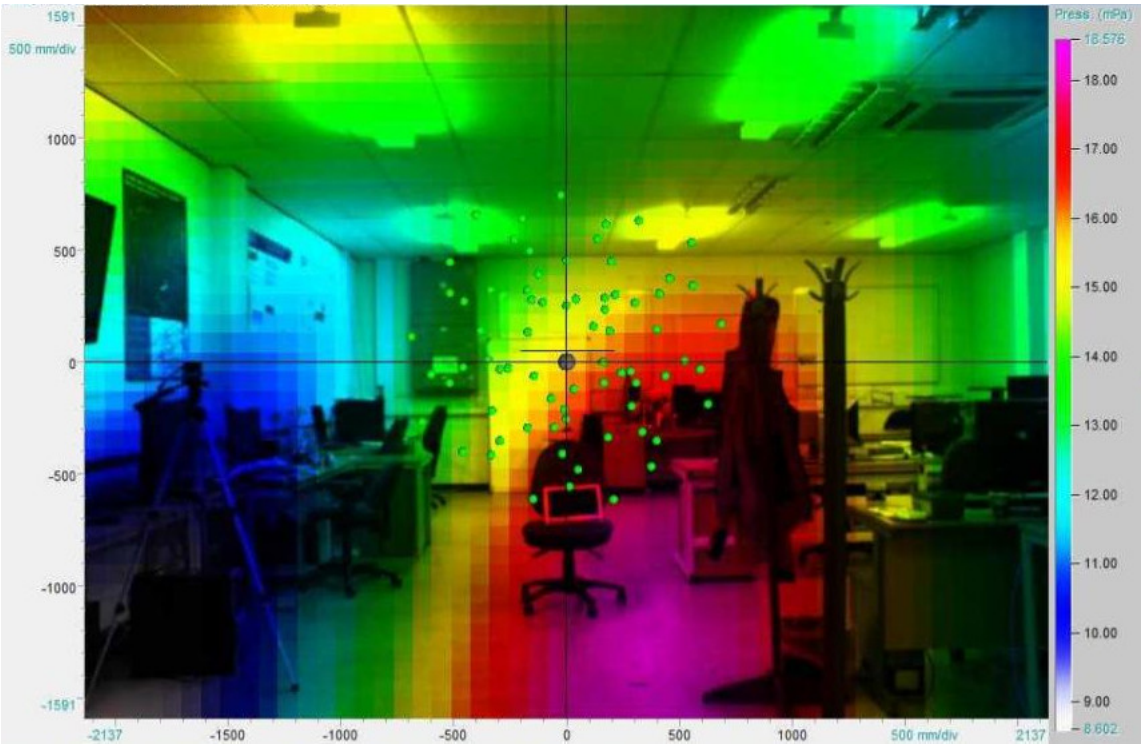
(e) Focus set to 1 metre but sound source is located at 8 metres distance



(f) Focus set to the same distance (8 metres) as the sound source

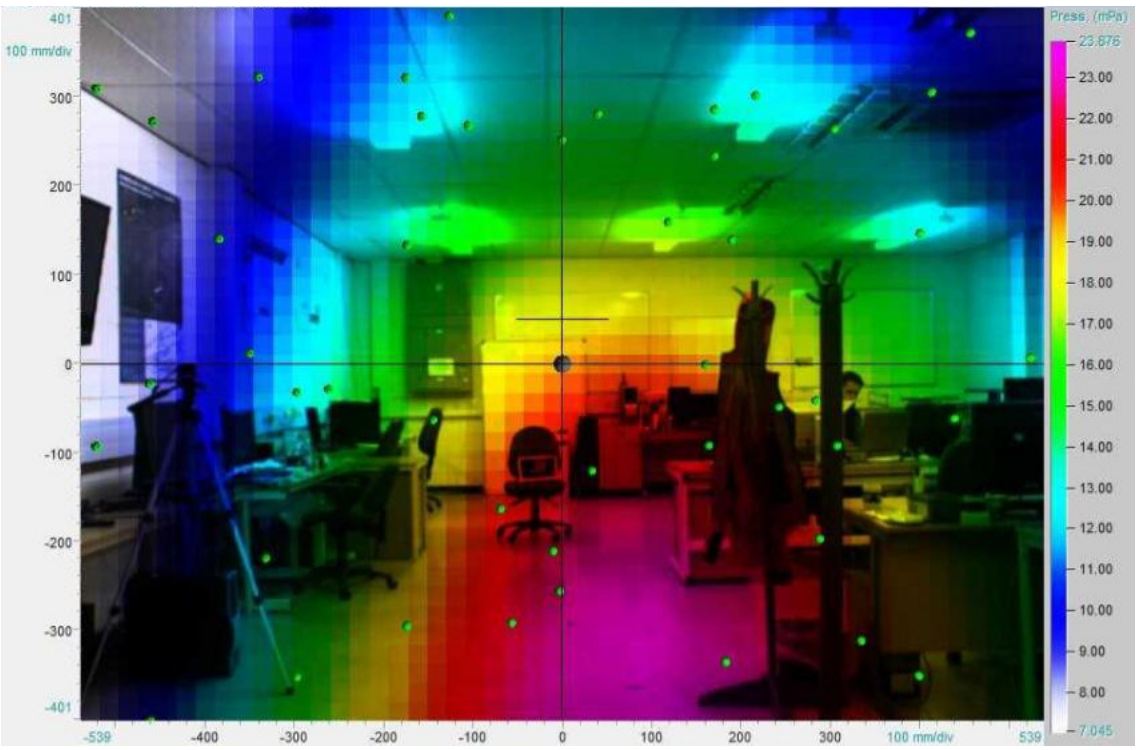
Figure 4.3: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a ring microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera (contd.).

(a) Focus set to 1 metre but sound source is located at 4 metres distance

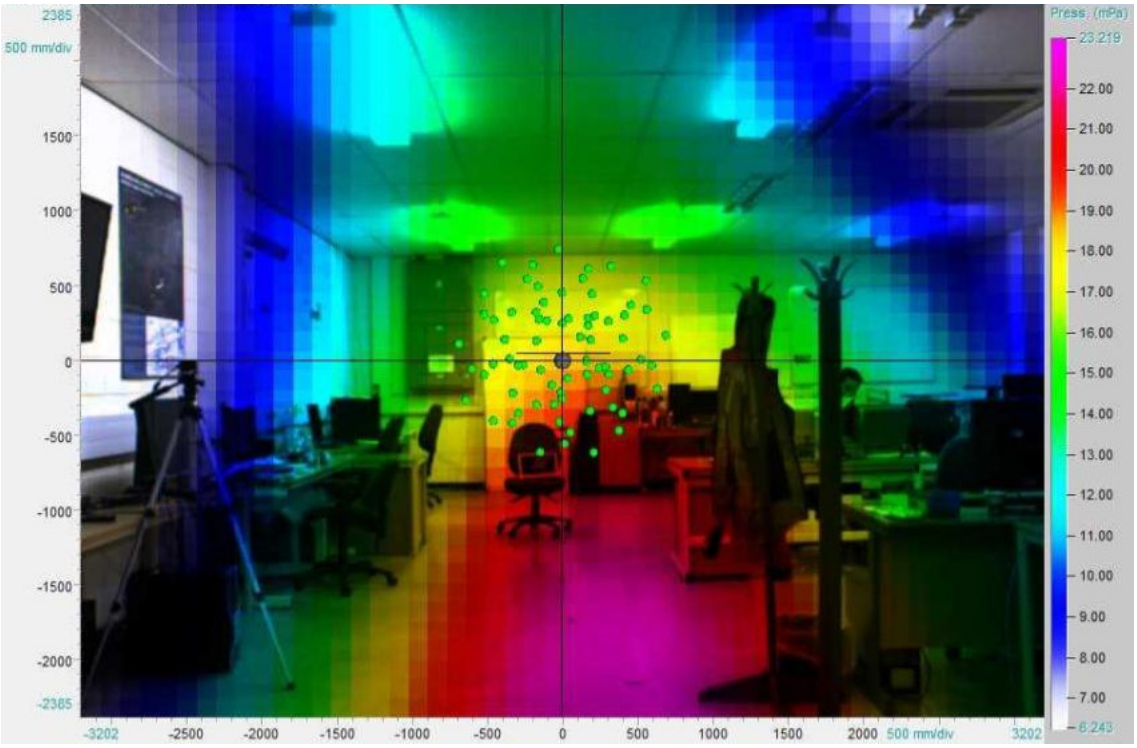


(b) Focus set to the same distance (4 metres) as the sound source

Figure 4.4: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a spiral microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera.

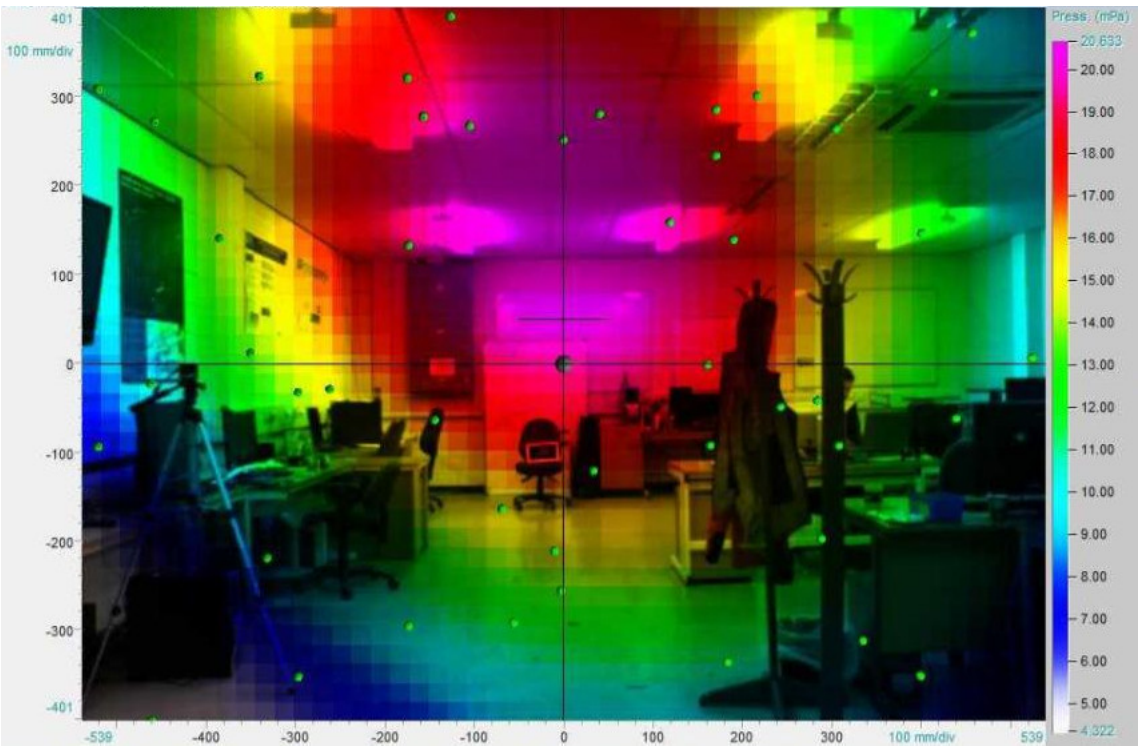


(c) Focus set to 1 metre but sound source is located at 6 metres distance

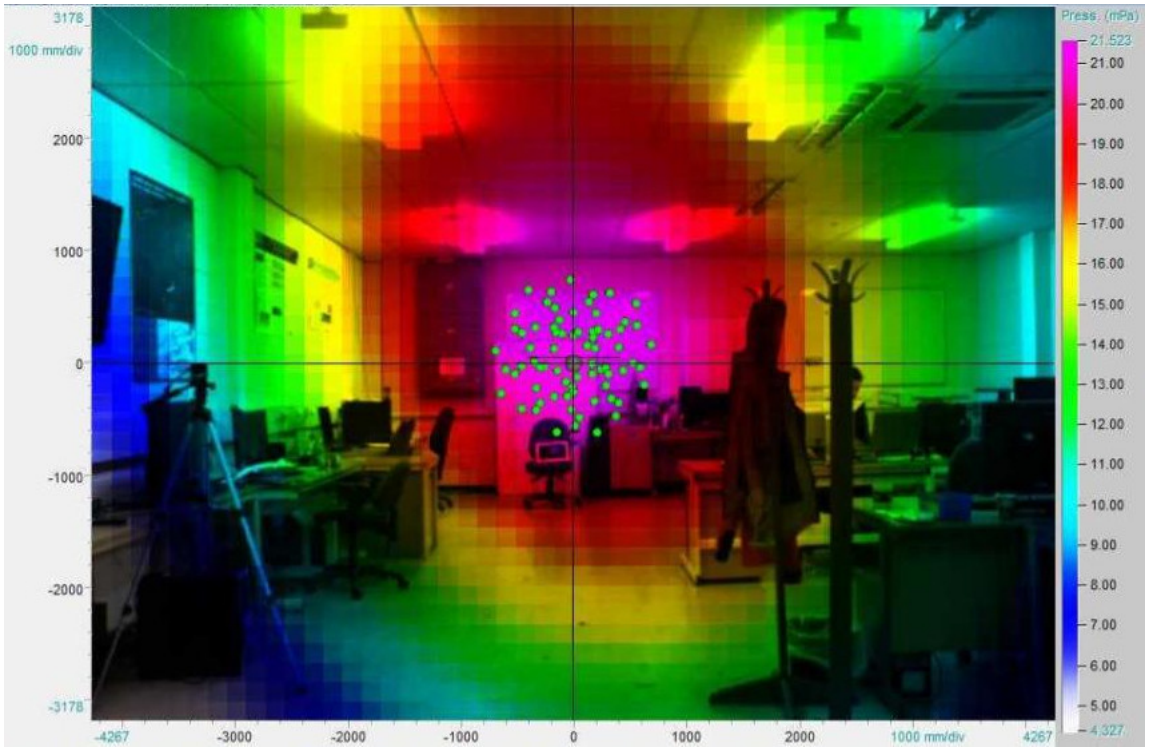


(d) Focus set to the same distance (6 metres) as the sound source

Figure 4.4: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a spiral microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera (contd.).

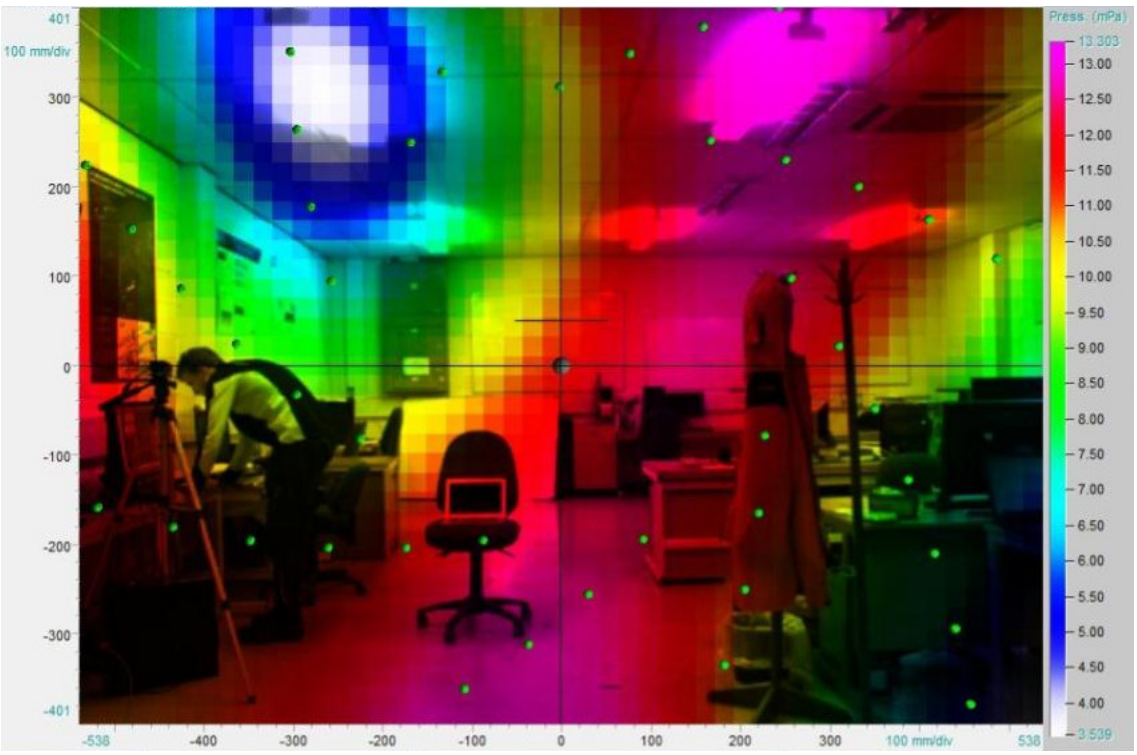


(e) Focus set to 1 metre but sound source is located at 8 metres distance

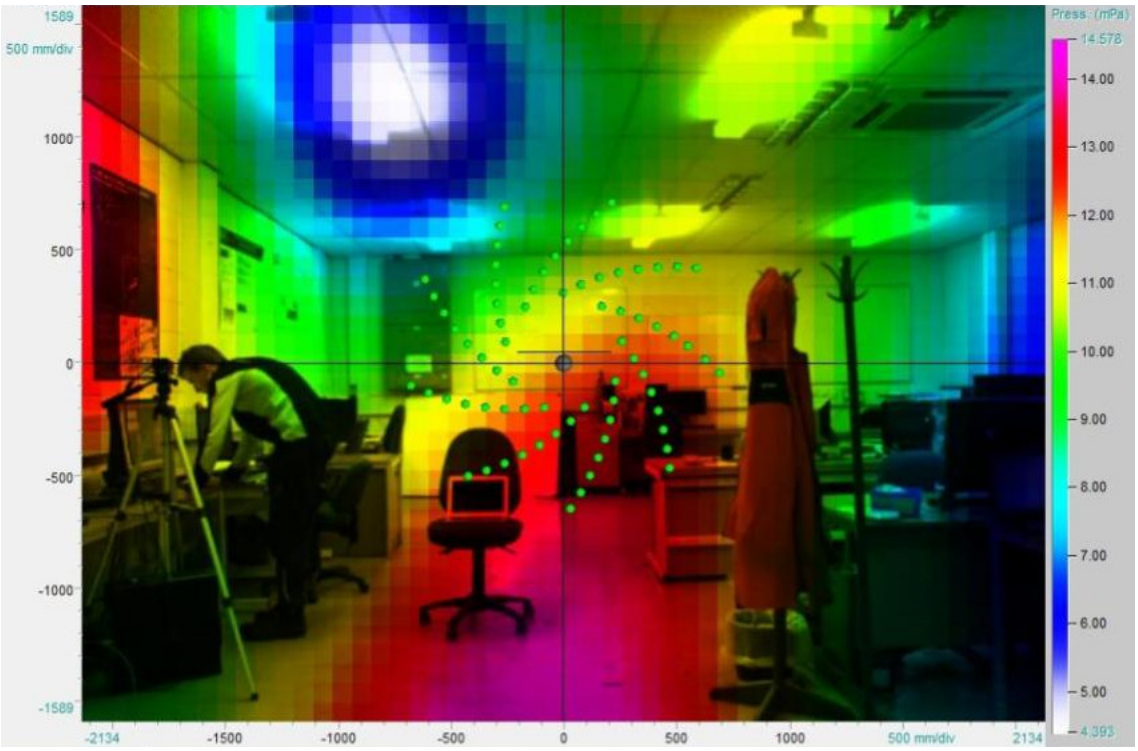


(f) Focus set to the same distance (8 metres) as the sound source

Figure 4.4: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a spiral microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera (contd.).

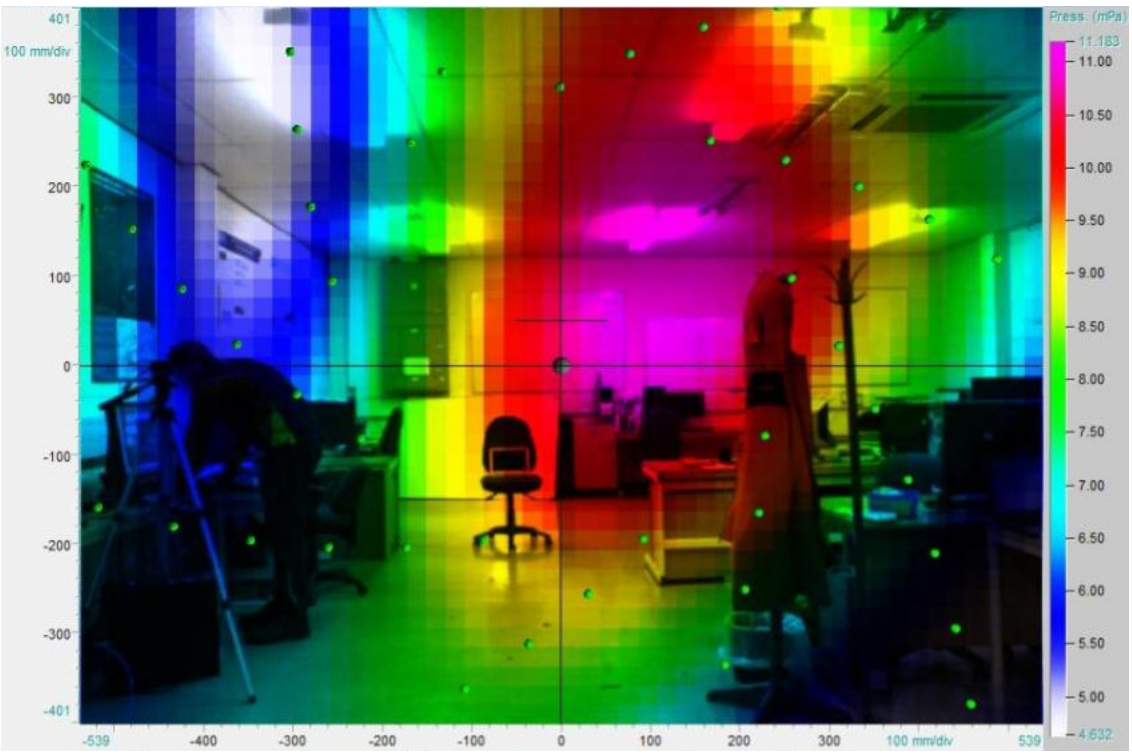


(a) Focus set to 1 metre but sound source is located at 4 metres distance

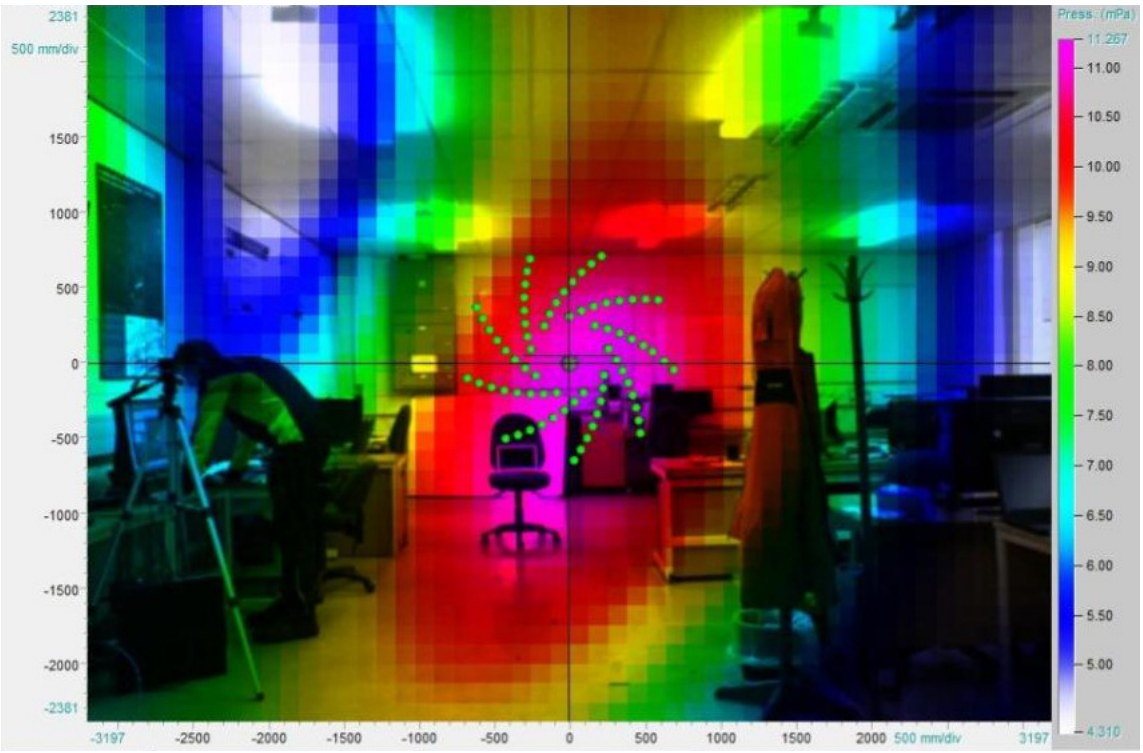


(b) Focus set to the same distance (4 metres) as the sound source

Figure 4.5: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a wheel microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera.

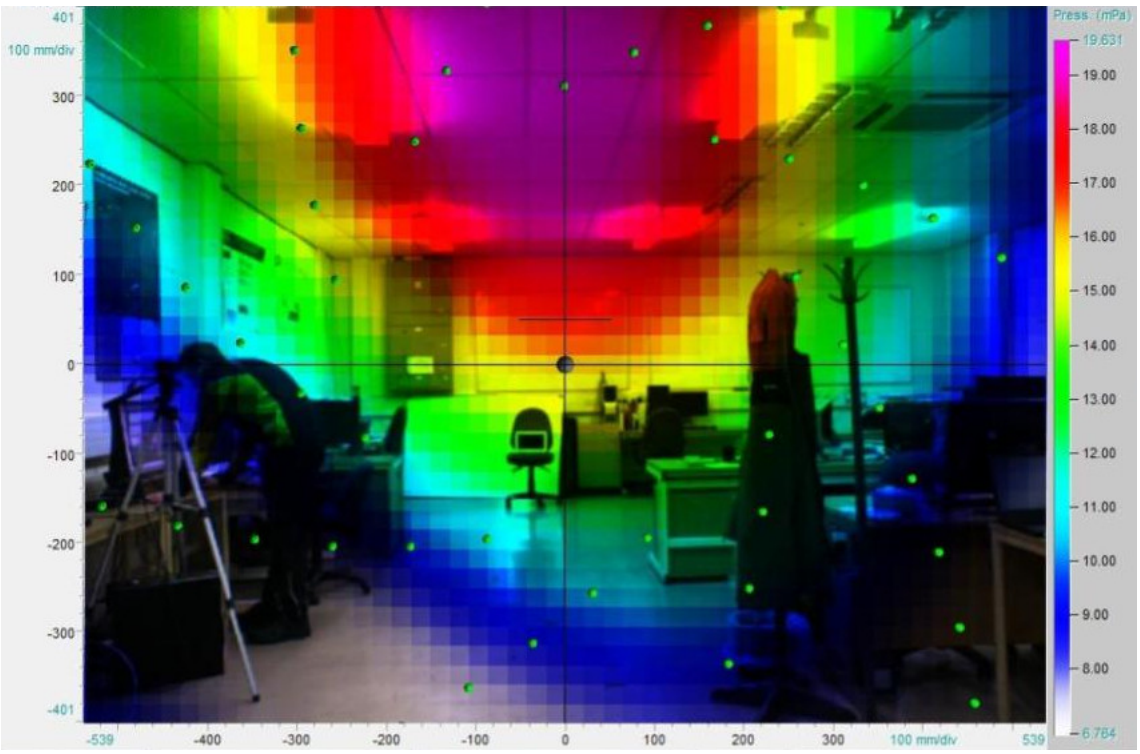


(c) Focus set to 1 metre but sound source is located at 6 metres distance

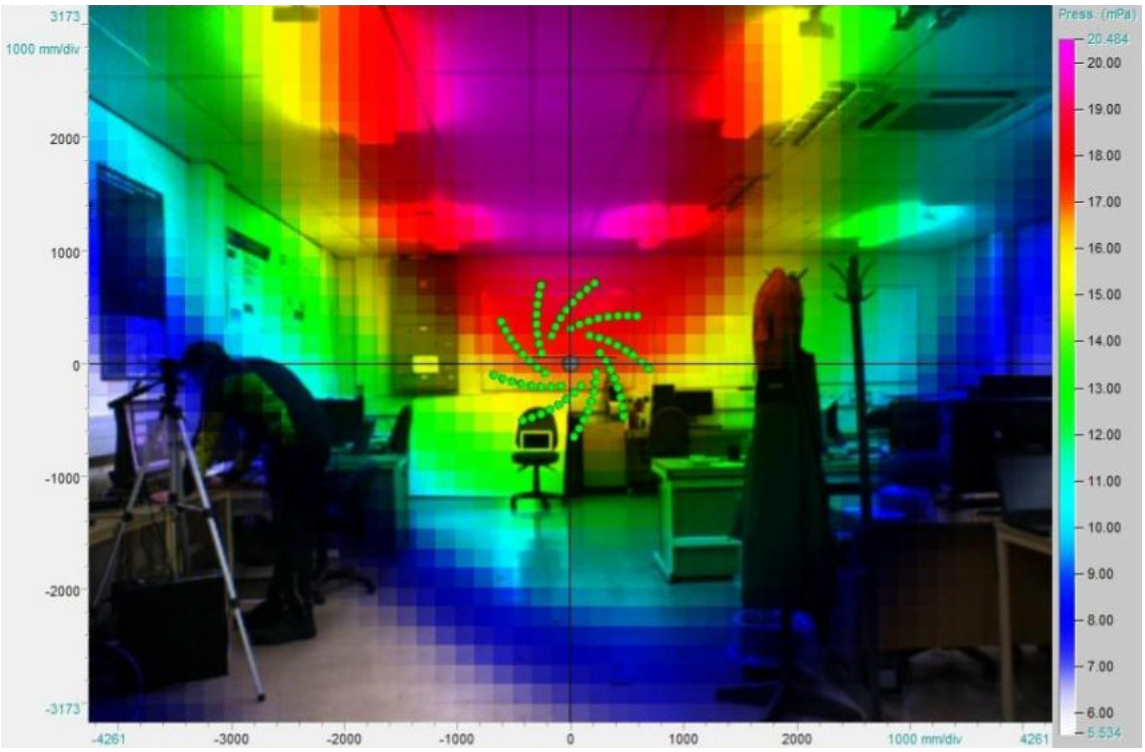


(d) Focus set to the same distance (6 metres) as the sound source

Figure 4.5: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a wheel microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera (contd.).



(e) Focus set to 1 metre but sound source is located at 8 metres distance



(f) Focus set to the same distance (8 metres) as the sound source

Figure 4.5: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a wheel microphone configuration. With the pink blob being the highest detected sound pressure down to the blue areas with the lowest detected sound pressure and the clear areas being the locations without a detectable sound pressure. The green circles represent the microphone configuration and the black dot represents the video camera (contd.).

The AC is particularly sensitive to reverberation and noise. For instance in Figure 4.5a where the air-conditioning unit in the ceiling makes a noise which is picked up by the acoustic camera and has the same loudness as the sound coming from the computer. The noise is localised as a single source and classified as being louder than the primary sound source. Reverberation, caused by the size and emptiness of the room, interferes with these algorithms and is localised as multiple sources in the output of the beam-forming algorithm (see Figures 4.6 and 4.7).

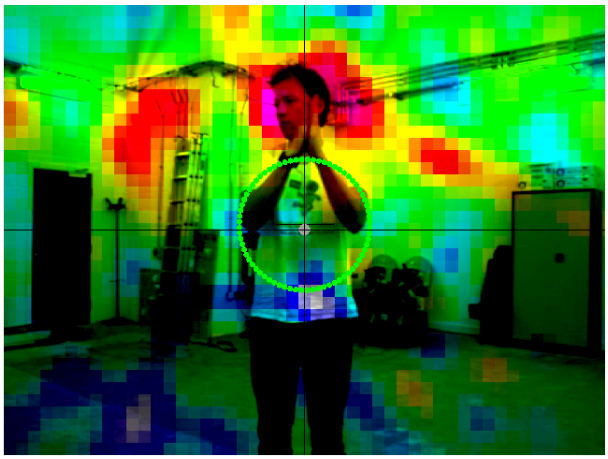


Figure 4.6: Pressure map showing the origin of the sound (pink blobs) and reverberation (red blobs) with reverberation coming from the back wall (rightside and leftside of the head). The green circles represent the microphone configuration and the black dot represents the video camera.

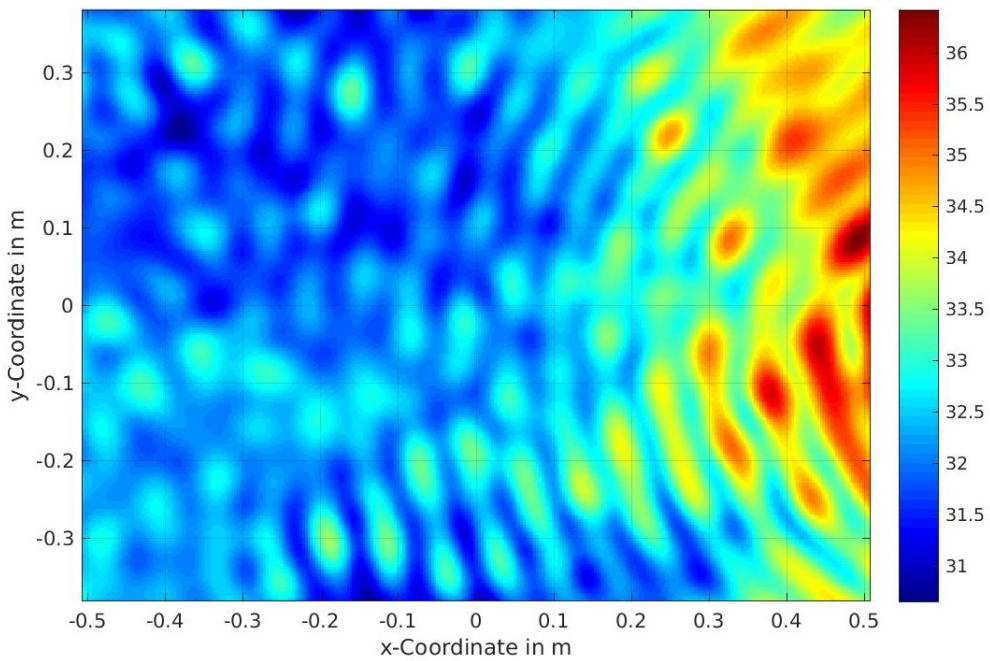
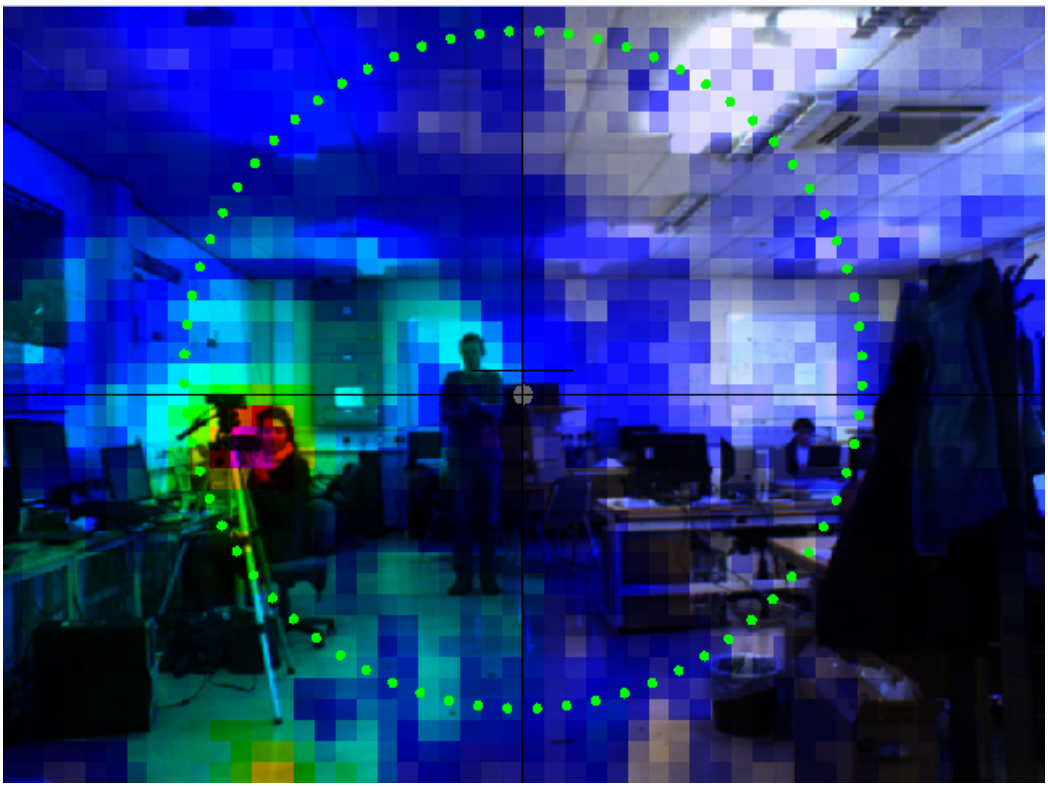


Figure 4.7: A sound source located on the edge of the field of view of the microphone array. The dark red blobs and the ripples show the main reverberation of the sound. The scale shows the sound pressure (dB)

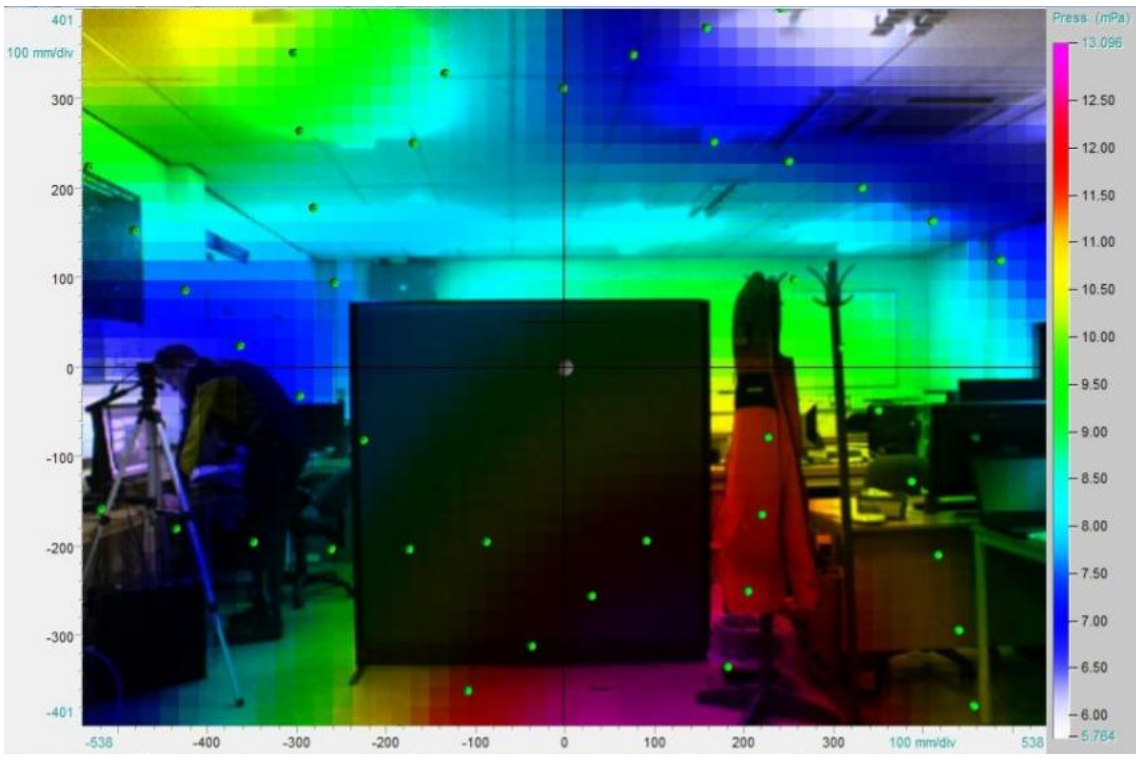
The AC is able to locate sounds on the edge of and outside the view of the camera (see Figure 4.7) and can also distinguish between two sources only when they do not overlap in their 2D location. This provides us with more information than having either video and audio separately. For example, looking at the audio and video separately, is not always evident where sound is coming from (see Figure 4.8).

(a) Camera view

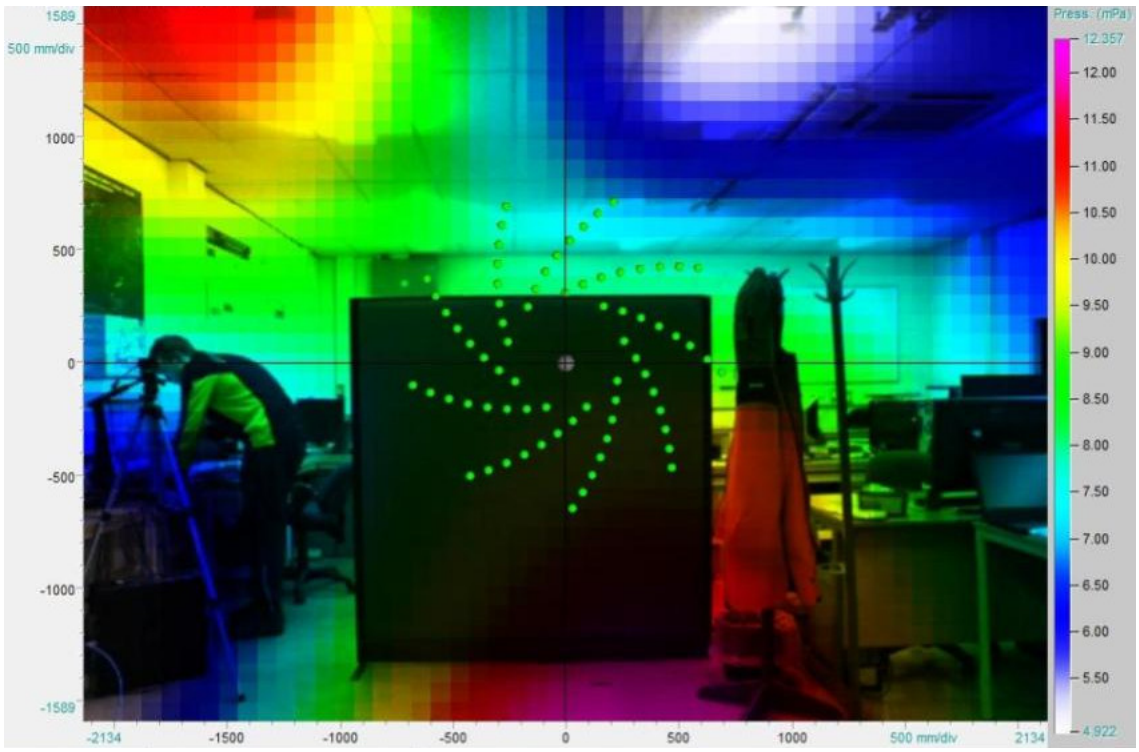


(b) Beamforming view. The red blob shows the sound source, the blue is the reverberation of the sound. The green circles represent the microphone configuration and the black dot represents the video camera.

Figure 4.8: A second person clapping their hands as seen on the video (a) and as seen by beamforming (b).

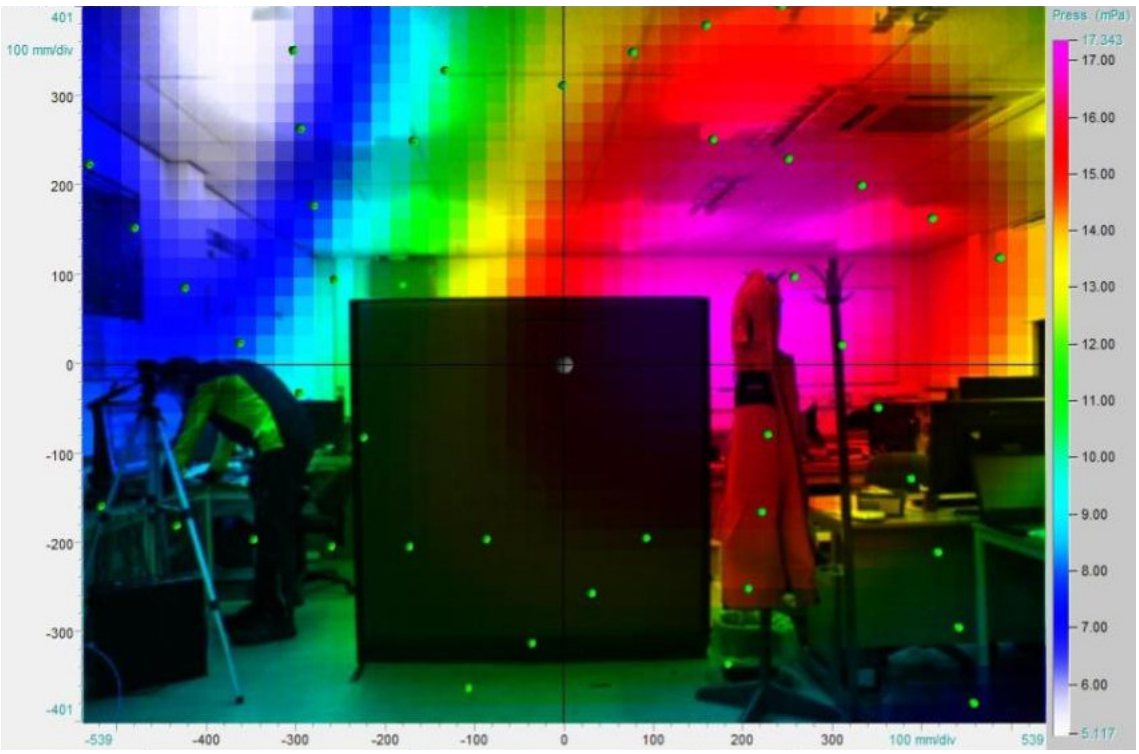


(a) Focus set to 1 metre but sound source is located at 4 metres distance directly behind the wall

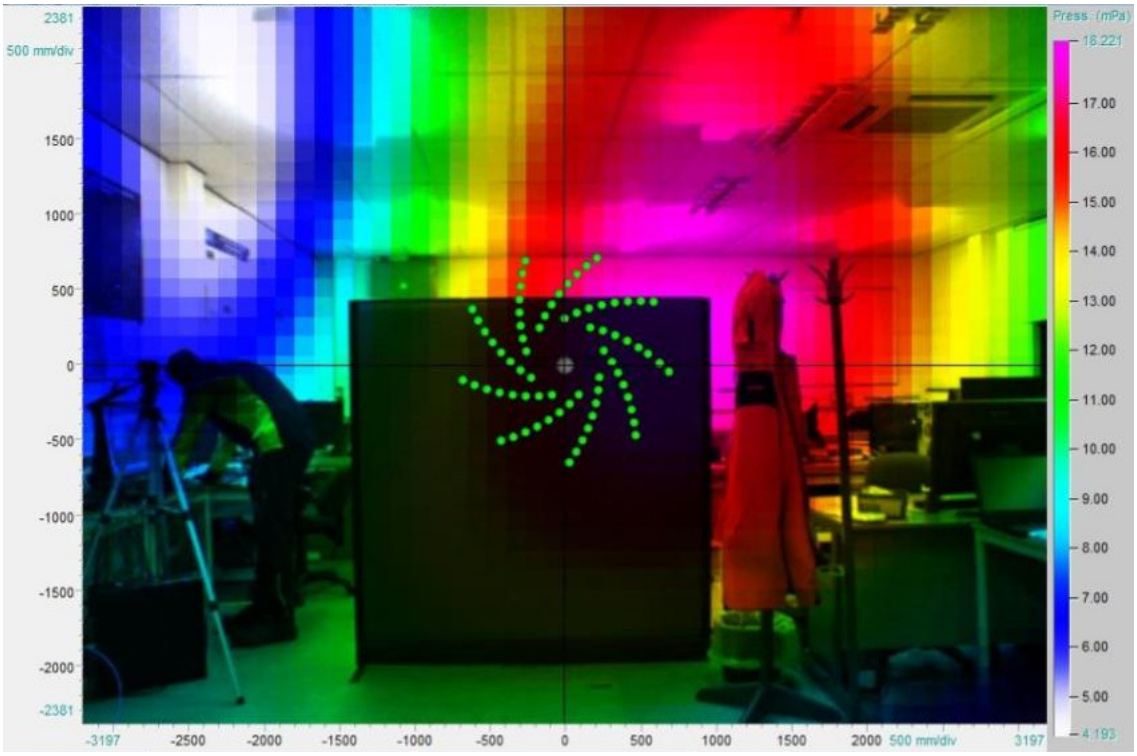


(b) Focus set to the same distance (4 metres) as the sound source directly behind the wall

Figure 4.9: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a wheel microphone configuration with the sound source behind a section of wall. The green circles represent the microphone configuration and the black dot represents the video camera.



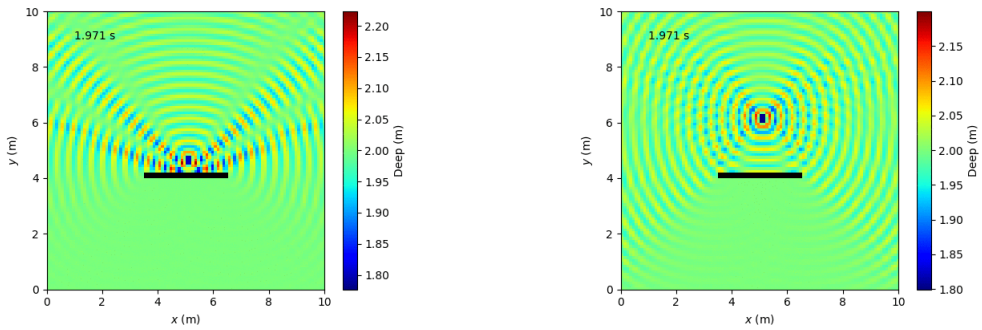
(c) Focus set to 1 metre but sound source is located at 6 metres distance directly behind the wall



(d) Focus set to the same distance (6 metres) as the sound source directly behind the wall

Figure 4.9: The influence of setting the correct focus on the accuracy of the beamforming algorithm using a wheel microphone configuration with the sound source behind a section of wall. The green circles represent the microphone configuration and the black dot represents the video camera (contd.).

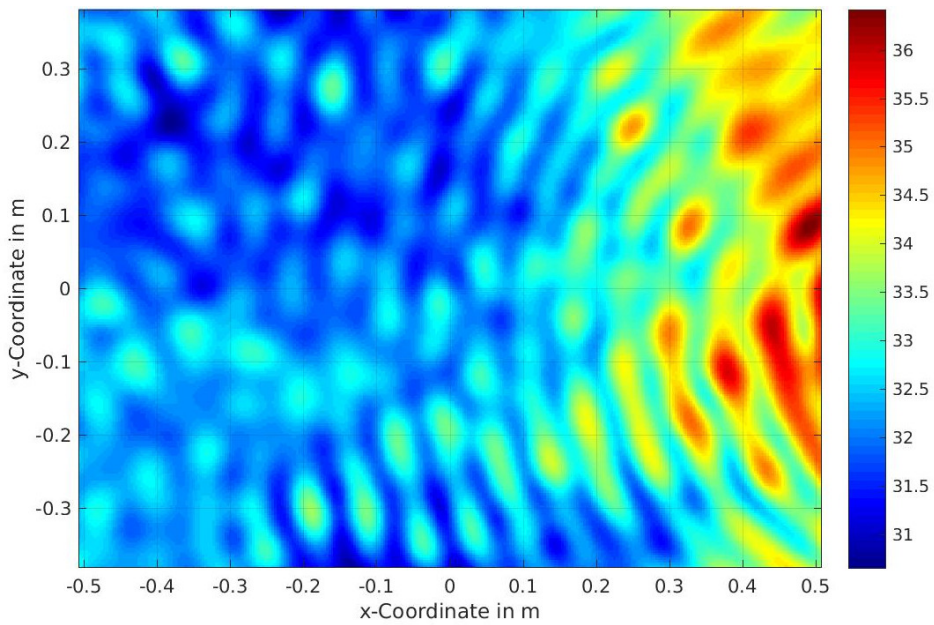
When there is an obstruction between the sound source and the AC it becomes more difficult for the beam-forming algorithms to determine where the sound is coming from (see Figure 4.9). However, this is also dependent on the distance from obstruction to sound source. When the distance between sound source and obstruction is large the beam-forming algorithms are more accurate because the sound is able to travel around the obstruction more easily. Whereas, when the sound source is close to the obstruction the sound has more difficulties in reaching the microphones (see Figure 4.10).



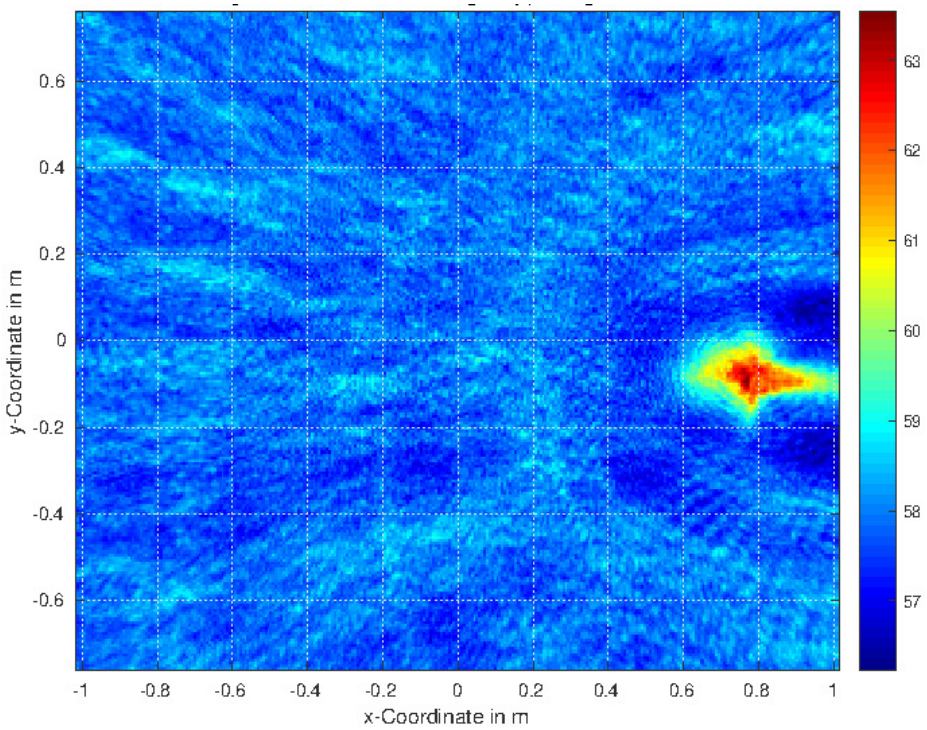
(a) Sound source close to obstruction (b) Sound source away from obstruction

Figure 4.10: A comparison between obstruction when it is close to the sound source and further away.

Applying beam-forming algorithms in the frequency domain provides greater accuracy in where the sound source is located than when we apply a time-domain beam-forming algorithm. The time-domain beam-forming algorithms are more influenced by secondary sound sources (noise or reverberation). Resulting in a source showing up in a different location than it actually is. In frequency-domain beam-forming algorithms these secondary sources are localised as having a lower intensity than the main source and show up as blobs coming from the location of the main source (see Figure 4.11). This gives a greater accuracy of where the primary source is located.



(a) Frequency-domain beam-forming using GCC-PHAT



(b) Time-domain beam-forming using delay-and-sum

Figure 4.11: A comparison between frequency beam-forming and time beam-forming. The sound source is located on the right outside the view of the camera

4.3 Recordings

For the recordings, the speakers were given a book to read from[1]. Pages 1-7 were enough for the speakers to read in the timeframe it took to make a recording. The recorded speech was mainly English by non-native English speakers[2]. For all recordings, the exact time at which speaker started and finished was recorded. The recorded text is added to the corpus in a transcript file. In addition to the ground truth text, a single video and recordings from all the 72 microphones in individual files were obtained.

The speakers were instructed to first stand still and then walk around. At the start of the recordings, the minimum distance between the AC and the speaker was 6 metres. To allow for free movement through the room this was reduced to 3 metres during the walking stage of the recording.

	Separation recordings	**Tracking recordings**
single speaker	5	12
multi-speaker	5	0
subjects	3 (2 women, 1 man)	16 (4 women, 12 men)
distance microphone to source	$>$ 5 metres	$>$ 3 metres

Table 4.1: Overview of the recordings.

4.3.1 Separation recordings

To use for speaker separation, 10 recordings were made (see Table 4.1). These recordings were made in a room 13 metres long, 8 metres wide and 3 metres high. The room contained several air-conditioning units, which add noise to the recordings. This room represents a realistic office space with furnishing (12 tables and 48 chairs). There was no specific acoustic sound proofing in the room. The absence of the acoustic sound proofing adds reverberation to the recording.

These recordings contain speech of one, two or three speakers. The single speaker recordings can be used to create new multi-speaker recordings by randomly mixing two recordings together. For these recordings; the speakers were instructed to stand

[1] "Away in the Wilderness" by R. M. Ballantyne, pages 1-7
[2] There are also 5 recordings of Dutch speech

still for 60 seconds, after which they got a signal to start walking through the room keeping a minimum distance of 6 metres away from the AC (indicated by tape on the floor).

4.3.2 Tracking recordings

17 recordings of single speakers were made with additional data from the Microsoft KinectV2 in a room 8 metres long, 8 metres wide and 5 metres high. This room is similar to a large workshop or small factory environment. There is no sound proofing installed in the room. The roof is made of metal sheets which amplifies exterior and interior sounds. Reverberation is an unavoidable part of the recordings made in this room because of its size and insulation properties. These noises and reverberation can be heard in the recordings.

During these recordings the speakers were instructed to stand still at 6 metres from the AC for 45 seconds. After a signal, they could start walking around the room. The speakers were instructed to keep a minimum distance of 3 metres from the AC (indicated by markings on the floor). These recordings contain the audio coming from the AC, the video from the AC and the manually added ground truth of the spoken text. In addition to this, there are audio recordings made by the Kinect microphones and the infrared and depth videos made by the Kinect. The depth video gives a x, y, z, location of the speakers when they are in the field of view of the Kinect.

To synchronize between the two different devices every recording starts with a clap, which is later removed in the post processing.

N.B. these recordings can also be used for speakers separation as described in the previous section.

4.3.3 Post processing

All of these recordings contain noise and reverberation. Removing these will simplify the speaker separation problem. However, to be able to test the performance of the algorithms in the real world, either or both should be present in the original dataset. Therefore, the data is processed to create three other versions of the dataset (no_noise, no_echo, original and nonoise_echo), each adding a new level of complexity

to the speaker separation problem (see Table 4.2). These complexities are compared to the original recordings to see how the signal to noise ratio of the dataset improves or degrades when that property is removed. Due to the nature of the removal algorithms, the datasets where the reverberation is removed also degrades the audio signal, whereas when only the noise is removed the audio signal is improved.

This post processing has to happen offline because the noise between the recordings varies. It is therefore more difficult to create a model that can remove all of the noise in real time. This is something seen in real life too, when recordings are too short, then noise reduction algorithms have performance issues. For the noise reduction, spectral noise gating is used. This technique uses a quiet segment of the audio as a noise sound fingerprint. This is then removed from the audio as a whole, which was done for each speaker and each microphone separately. In this case, reverberation is the lengthening of speech by reflections. As these reflections are often lower in amplitude, the associated frequency levels can be compressed, allowing the persistence of the sound to be controlled. A multiband compressor is used to select the frequency levels to compress. For both techniques, the implementations in Audacity® v2.1.2 [182] are used.

dataset	tracking recordings	separation recordings
no_noise	1.73	2.53
no_echo	-1.35	-1.39
nonoise_echo	-2.34	-2.34

Table 4.2: Overview of average change in signal to noise ratio (in dB) of the datasets compared to the original recordings.

4.3.4 Corpus organisation

The corpus is divided into folders representing the number of speakers present in the recordings. These can be used in different situations. For the recordings with multiple people are no single speaker ground truth files available. Each of the folders containing files with one, two or three speakers is divided per recording room (LC and LR). Each of the rooms contains the different speakers recorded in this room and their session number (for example, of speaker T1 there are two sessions T1_1

and T1_2). Within the speaker folder there, is a further division based on whether the recording contains noise, reverberation, both or neither (respectively no_noise, no_echo, nonoise_echo and original). Next to these four folders, the video recordings from the AC camera and Kinect (if present) can be found in this folder. Each of the four dataset folders (no_noise, no_echo, nonoise_echo and original) contain the recordings of the 72 microphones at 192 kHz.

4.3.5 Use cases

The size and properties of the acoustic camera do not allow it to be mounted on a robot. However, the corpus can be used for building models that solve problems in robotics and these models can be used with different microphone arrays which have similar properties to those used for creating the models. For example, the corpus can be used for dereverberation, therefore the data has been passed through a multiband compressor can be used as a baseline. The influence of realistic noise on the algorithm can be tested using the original data. For noise cancelling, a similar approach can be adopted. A total of 72 microphones can be used for a multi-channel approach to improve the accuracy of the algorithm. The multi-speaker recordings can be used to increase the complexity of the problem for the algorithm.

For speaker separation, the data can be mixed in several different ways. Different microphones can be used to simulate a distance between speakers or the same microphone can be used to simulate two speakers in the same location. However, the mixing of single speaker files does have a disadvantage. When the mixing includes the reverberant signal of both speakers, it will be less realistic because these have been recorded at different times. Meaning that there is a difference in the reverberation of two speakers artificially mixed together and two speakers speaking at the same time. For the multi-channel algorithms the data from the Kinect can be used to determine the exact location of the speakers.

The fourth usage of the data considered here is to do speaker tracking. The array setup allows for different configurations to be used: from a linear stereo array (by selecting two oppositely located microphones) to using all 72 microphones for localisation. In addition to this there is location data from the Kinect which can be used as ground truth data. However, there are instances in the recordings where the

speaker walks out of view of the cameras but still can be heard. This is an added level of complexity to the data.

Another usage for the corpus is onset of speech detection. The AC's high quality recordings allow for accurate detection of when the speaker starts to speak. For this, the recordings of all microphones can be used and the algorithm can be tested in the four different settings.

4.4 Conclusion

Within this chapter, a novel corpus that can be used to evaluate algorithms for speaker separation was presented. This corpus contains realistic data that can be used to train machine learning models for speaker separation in noisy and reverberant environments (see for example Chapters 6 and 7). However, this is not the only usage of the corpus. It can be used for speaker localisation and tracking and noise and reverberation removal or suppression. The tracking algorithms need to work with the reverberation present in the recordings but can also use the other three datasets where some of the properties have been removed. Noise and reverberation removal can be compared to the sets where that property has been removed. These sets can be used as a baseline.

The microphone array used for recording this corpus allow for a precise localisation and recording of different sound source. However, its size and ability to only record 90 seconds make it unsuitable for real-time recordings or placement on a robot. The recordings can be used for training a robot to respond to the correct speaker and allow for the creating of a smaller microphone array with a similar configuration to the one used for training the robot (or algorithm).

Chapter 5

Dereverberation

5.1 Introduction

With the introduction of artificial assistants and robots into our homes and work environments, speech recognition has moved from controlled conditions to unpredictable conditions. Now, instead of working in a small clean lab environment, speech recognition has to work in different environments varying in size from, for example, a living room with minimal furnishing to a large factory hall. These various environments introduce reverberation from the walls and ceiling making it difficult for a robot or artificial assistant to understand the speaker and execute critical tasks.

Reverberation is similar to echo, however in the case of reverberation the reflections arrive within 50ms, whereas with echo the reflection arrives after 50ms. Reverberation is described as the persistence of sound (in this case speech) after the sound has been produced. This leads to a richer and warmer sound which is useful for music but also to errors in speech recognisers if they cannot determine when a phoneme (phonetic description of combinations of letters) has ended, introducing additional letters in the transcription which should not be there. This in turn creates an issue with the speech recogniser and with language understanding for artificial assistants and robots further down the line. For robotics, a working speech recogniser is important to understand what the speaker wants and what the robot needs to do. With reverberation, it is difficult for a robot to distinguish between actions spoken by the speaker, which may affect the robot's choice of executing a critical task. This can result in breaking an object, for example when the robot is

about to push a mug off a table and it does not recognise the word "STOP" spoken by the speaker.

This chapter concentrates on investigating the performance of existing algorithms and novel algorithms for the removal of reverberation. There are two ways of removing the reverberation from a sound signal:

- dereverberation (supervised dereverberation), where the sound signal is compared with a ground truth (the signal without reverberation) and a mask is build using the reverberant signal and the ground truth.
- blind dereverberation (unsupervised dereverberation), where multiple channels of the reverberant sound signal are compared to build a mask for removing the reverberation. This does not use the ground truth signal.

Both ways assume that the reverberation can be approximated by calculating the cross-correlation between signals. Supervised dereverberation uses the non-reverberant signal (Y) and the reverberant signal (X). For the blind dereverberation the cross-correlation is calculated between multiple microphones that receive the reverberant signal. This cross-correlation is assumed to describe the reverberation in the frequency domain and therefore the frequency response function (FRF).

Both dereverberation and blind dereverberation methods of removing the reverberation are investigated in this chapter. In order to do this, two different categories of algorithms are evaluated; learning and non-learning algorithms. The learning dereverberation algorithms build a mask and are able to adapt this mask over a number of iterations to match the reverberation better, thus learning the frequency response function. For the learning correlation based algorithms the existing H_1 NTF, WPE and MIMO WPE algorithms are compared with the novel H_s NTF, Cauchy WPE and Cauchy MIMO WPE algorithms. These six learning based algorithms are divided in two categories (supervised and unsupervised). The supervised algorithms (H_1 NTF and the novel H_s NTF) calculate the correlation between the ground truth and the microphone as the reverberation. On the other hand, the unsupervised algorithms (Cauchy WPE, WPE, Cauchy MIMO WPE and MIMO WPE) only use the information coming from the microphones as input and calculate the correlation between two microphones.

The non-learning algorithms build a mask but do not adapt this mask meaning

that the mask is approximated once. The non-learning correlation based algorithms are the existing H_1, H_2 and H_s correlation measurements. The H_1 method (see Section 2.3.2) assumes that there is noise in the reverberant signal and produces an underestimate of the FRF when the noise is in the non-reverberant signal instead. The H_2 method (see Section 2.3.2) assumes that there is noise in the non-reverberant signal and produces an overestimate of the FRF when the noise is in the reverberant signal instead. Finally, the H_s method (see Section 2.3.2) tries to find the balance between the H_1 and H_2 methods by scaling the influence of each of the correlation techniques. This particular implementation by Leclere [32] has not been used before for doing speech dereverberation. These algorithms are used for both dereverberation and blind dereverberation.

These algorithms are used because they are explainable, tractable and easily expanded. Other algorithms such as Hidden Markov models (HMM), autoencoders and generative adversarial networks (GAN) do not share all of these properties. For instance, autoencoders and GAN are easily expanded but not tractable or explainable whereas HMMs are explainable but difficult to expand.

There are four main stages in this chapter:

- Comparing the performance of the three non-learning algorithms (H_1, H_2 and H_s) in both a dereverberation and blind dereverberation setting. This is to see the influence of the reverberation on the different correlation measurements and to determine which is best at determining the frequency response function.
- Comparing the performance of the novel H_s NTF algorithm with the result of the H_1 NTF algorithm. The H_s NTF should give an improved performance because it will be able to account for noise being present in the approximation as well as in the input.
- Comparing the performance of two novel variants of the WPE and MIMO WPE methods to the original WPE and MIMO WPE algorithms. In their original form, WPE and MIMO WPE use the Gaussian distribution to calculate the reverberation. The two novel variants are based on the Cauchy distribution (see Section 2.2.4) which has a longer tail. This means that the distribution is able to include more frequencies in its calculation.
- Comparing the performance of the WPE and MIMO WPE algorithms as well

as the two novel variants using different window functions. This shows why Drude et al. [108] suggest using a Blackman window function over the normally used Hann window function. Instead of concentrating only on the Blackman and Hann windows, six different window functions (Bartlett-Hann, Bartlett, Blackman, Hamming, Hann and triangular) are compared. This is to see if there is a different window function outside of the two normally used for these experiments.

The performance of these algorithms is measured using five algorithms (PESQ, SDR, SDR_{mir}, SISDR and SNR). These are chosen to be able to compare the results with those of the existing literature. The algorithms are run on the TIMIT dataset and the MIMO WPE algorithm is also run on the Acoustic Camera (AC) corpus. The TIMIT dataset is chosen to make a comparison with existing literature whereas the AC corpus gives the MIMO WPE an environment with realistic reverberation to test the performance on.

5.2 Algorithms

5.2.1 H_s NTF dereverberation method

The H_s method described in the previous section can be combined with work by Ozerov et al. [40] to create a new algorithm which learns the reverberation. In this section the reverberation matrix is being learned, not the separation of sound sources which is the original intention of Ozerov's work and will be described in Chapter 7. The idea behind this is that with updating the approximation of the reverberant signal by multiplying two nonnegative matrices together and working in the frequency domain approximates the frequency response function. This improves the quality of speech and reduces the number of artefacts because the dereverberation mask can be tailored to the input signal by constantly adapting the approximation of the frequency response function.

To make the algorithm concentrate on the reverberation instead of creating the speech signal as well, the non-reverberant signal is used as second input. This means that the algorithm is used in a supervised fashion where both the reverberant and non-reverberant signals are known, giving the algorithm one less matrix to learn

and fully concentrate on calculating the difference between the non-reverberant and reverberant signals.

As explained in Section 2.3.2, the reverberant signal in the frequency domain is the multiplication of the non-reverberant signal with the FRF. The FRF does not change over time (i.e. the influence of the reflections is only described in the frequency domain).

$$x_{fn} = A_{fn} s_{fn} + b_{fn} \tag{5.1}$$

First the reverberant signal (x) is divided into a non-reverberant signal (s), noise (b) and the reverberation (A) (see Equation 5.1). To calculate the reverberation, the H_s method (see Section 2.3.2) is used. This method accounts for noise being present in both the original mixture and the approximation of this mixture.

Next Ω_s is calculated which is the non-reverberant signal per microphone multiplied by the Hermitian transpose of the reverberation and the inverse of the reverberation signal per microphone (see Equation 5.2).

$$\Omega_s = \Sigma_s A^H {\Sigma_x}^{-1} \tag{5.2}$$

This calculation is followed by calculating the cross-correlation of Ω_s and the original input (see Equation 5.3). The original input is defined by Equation 5.4.

$$\hat{\Sigma}_{xs} = \hat{\Sigma}_x {\Omega_s}^H \tag{5.3}$$

$$\hat{\Sigma}_x = xx^H \tag{5.4}$$

Both Ω_s and $\hat{\Sigma}_{xs}$ are used to calculate the reverberation (A), for this, first $\hat{\Sigma}_{xs}$ (the cross-correlation of Ω and the reverberant signal) is multiplied with the inverse of the non-reverberant signal (see Equation 5.5).

$$A = \hat{\Sigma}_{xs} {\hat{\Sigma}_s}^{-1} \tag{5.5}$$

The reverberant signal per microphone is calculated using the non-reverberant signal per microphone multiplied by the reverberation and the Hermitian transpose

of the reverberation and finally noise is added to the signal (see Equation 5.6).

$$\Sigma_x = A\Sigma_s A^H + \Sigma_n \tag{5.6}$$

For approximating noise, Ozerov et al. [38] is followed where the noise is described as Equation 5.7

$$\Sigma_n = \hat{\Sigma}_x - A\hat{\Sigma}_{xs}{}^H - \hat{\Sigma}_{xs}A^H + A\Sigma_s A^H \tag{5.7}$$

Equation 5.5 is replaced by Equation 5.8 to create the H_s NTF version. This is to account for the noise that is present in the mixture (instead of the approximation) or in both the mixture and the approximation. $\hat{\Sigma}_{sx}$ is calculated by interchanging $\hat{\Sigma}_x$ and Ω_s in Equation 5.3 thus forming Equation 5.9.

$$A = U_n V_n{}^{-1} \tag{5.8}$$

$$\hat{\Sigma}_{sx} = \Omega_s \hat{\Sigma}_x{}^H \tag{5.9}$$

In Equation 5.8, Σ_{hs} is the result of taking the n biggest values of the singular value decomposition (SVD) of the correlation matrix (see Equations 5.10 and 5.11). These n values describe at most 99.9999% of the data. This means that the noise in the signal is removed which are the parts with the lowest correlation in the correlation matrix. The correlation matrix is built from 4 matrices namely: the input; the cross correlation between the input and Ω_s; the cross correlation between Ω_s and the non-reverberant signal and the non-reverberant signal. As explained in Section 2.3.2, the left singular vectors of the SVD (see Equation 5.11) are described by both U and V, s describes the singular values and W the right singular vectors.

$$\hat{\Sigma}_{xyxy} = \begin{pmatrix} \hat{\Sigma}_x & \hat{\Sigma}_{xs} \\ \hat{\Sigma}_{sx} & \hat{\Sigma}_s \end{pmatrix} \tag{5.10}$$

$$\hat{\Sigma}_{xyxy} = \begin{bmatrix} U \\ V \end{bmatrix} sW^H \tag{5.11}$$

5.2.2 Cauchy WPE

Weighted prediction error (WPE) (see Section 2.3.3) uses past information to predict the influence of the FRF on the current frame. This method offers a trade-off between a learning algorithm and an algorithm running in real-time. This method is changed to use the Cauchy distribution instead of the Gaussian distribution used in the original version of the algorithm. The original algorithm uses information from the past to improve the dereverberation mask. However, when a source is moving the algorithm uses information that is out-of-date. This creates an error in the mask and does not remove all of the reverberation. Unlike the methods described in Sections 2.3.2 and 5.2.1, WPE assumes that there is limited noise present in the recording (see Equation 5.12). This limits the algorithm to dereverberation in almost noiseless conditions.

$$Y(f) = X(f)FRF(f) \tag{5.12}$$

An important part of the WPE method is the autoregression process. This works by calculating the autocovariance of a signal at the same time step and the cross covariance at a delayed time step.

$$\begin{aligned} \sigma &= \frac{|x_{f,t-D}|^2 + |d_{f,t}|^2}{3 * |d_{f,t}|^2} \\ \Phi &= \sigma * \sigma^T \\ \phi &= \sigma * x^H_{f,t-D} \end{aligned} \tag{5.13}$$

$$\sigma = \frac{2 * \pi * (|x_{f,t-D}|^2 + |d_{f,t}|^2)^{3/2}}{|d_{f,t}|^2}$$

$$\Phi = \sigma * \sigma^T \tag{5.14}$$

$$\phi = \sigma * x^H_{f,t-D}$$

$$\sigma = \frac{|x_{f,t-D}|^2}{3 * |d_{f,t}|^2(|x_{f,t-D}|^2 + |d_{f,t}|^2)^+}$$

$$\Phi = \sigma * \sigma^T \tag{5.15}$$

$$\phi = \sigma * x^H_{f,t-D}$$

Another change to the WPE algorithm is calculating the autocorrelation of the approximated desired signal ($d_{f,t}$) which has been modified with the delayed signal. These changes are based on the Cauchy distribution (see Section 2.2.4) which does not use the mean of the approximated desired signal ($\frac{1}{T}\sum^T |d_{f,t}|^2$), instead the power magnitude of the desired signal is used ($|d_{f,t}|^2$). The cross-correlation (ϕ) was changed to calculate the correlation between the approximated desired signal and the delayed signal ($x_{f,t-D}$) (see Equations 5.13 to 5.15). There are three different modifications made which are called Cauchy v1 (see Equation 5.13), Cauchy v2 (see Equation 5.14) and Cauchy v3 (see Equation 5.15). These modifications remove the need to approximate the mean of the approximated desired signal as is the case with the original version of this algorithm (see Section 2.3.3). This should increase the speed of the algorithm.

5.2.3 Cauchy MIMO WPE

Multiple input multiple output (MIMO) WPE is an extension of WPE that outputs the same number of signals as it gets as input. Also as described in Section 2.3.4, it does not have the limitations that WPE has (limited noise present in the recordings). As MIMO WPE has the same structure as the WPE algorithm, the same three modifications can be applied without changing the overall structure of the algorithm.

It is difficult to improve the speed of the algorithm because it is running near real-time. However, these modifications offer a different way of approximating the desired signal. The main increase in speed is in calculating the filter matrix. Where WPE does this per frequency bin, MIMO WPE calculates this over all the frequency bins at the same time. This does not affect the precision of the algorithm but does increase the speed.

5.3 Corpora

For the dereverberation experiments, two corpora are used; TIMIT and the Acoustic Camera (AC) corpus. The TIMIT corpus [3] contains speech from a single speaker recorded with a close-talk microphone (see Section 3.1). These speech recordings do not contain noise or reverberation. This corpus is used to compare the methods against the existing literature. These recordings of this corpus are simulated in two rooms of different sizes. One of the rooms is of similar size as the one used for the AC corpus (see Section 4.3.2). The other room is of similar size as the one used by Parchami et al. [110] which is smaller than the room used for the AC corpus. This is to compare the results of the WPE and MIMO WPE variants with the six WPE versions described by Parchami et al. [110].

The AC corpus contains recordings of a single speaker in a realistic office and workshop environment (see Chapter 4). These recordings contain noise and reverberation which have been removed to create four different datasets of the same corpus each with their own properties (original, no_noise, no_echo and nonoise_echo).

5.4 Experimental setup

5.4.1 Corpora

For the dereverberation experiments seven different algorithms are applied to the TIMIT dataset. This is to compare the results from the experiments with those presented in existing literature (see Section 3.2). Cauchy MIMO WPE has also been applied to the AC corpus for testing in a realistic environment.

5.4.2 Environment

For the experiments two rooms with a single source and two microphones according to the specifications described in Table 5.1 (see Figure 5.1) are simulated. Room A is a replication of the tracking recording room used for the AC corpus (see Section 4.3.2), whereas room B is a replication of the room used by Parchami et al. [110] (see Figure 5.2). Room A is used to simulate a workshop environment with high levels of reverberation. This room is used for all algorithms.

Room B is used to compare the results with those of Parchami et al. [110] and is therefore only used for the Cauchy WPE and Cauchy MIMO WPE algorithms.

For the simulation of the rooms, the pyroomacoustics [183] library is used. It models the reflections from the wall using the image method [29]. This method only uses those reflections that are within a radius given by the speed of sound and the reverberation time (RT_{60}). Three of the six performance measurements are implemented in a library call mir_eval [184], PESQ is implemented in pypesq[1] and for SNR and SISDR own implementations[2] are used. The WPE and MIMO WPE algorithms are based on the implementations[2] by respectively Nakatani et al. [26] and Drude et al. [185]. The algorithms are run on a OpenSuse Linux computer with an Intel i3 processor with 4GB RAM.

As input to the algorithms, recordings from the TIMIT datasets are used as well as recording from the AC corpus. These recordings are chosen at random with a uniform distribution.

	x	**y**	**z**
Dimensions	8m	8m	4m
Source	4	8	1.5
Microphone 1	4.75	2	1
Microphone 2	5.5	2	1

Table 5.1: Dimensions of Room A and locations of microphones and the sound source

[1] https://github.com/vBaiCai/python-pesq
[2] https://github.com/TeunKrikke/dereverb

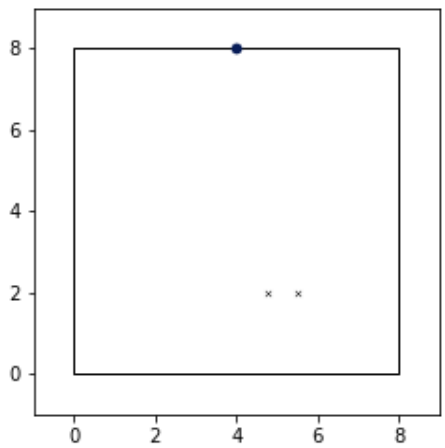


Figure 5.1: Top view of room A where the stars represent the microphones and the dot the speaker

	x	**y**	**z**
Dimensions	4m	5m	2m
Source	1.5	3	1.5
Microphone 1	4.75	2	1
Microphone 2	5.5	2	1

Table 5.2: Dimensions of Room B and locations of microphones and the sound source

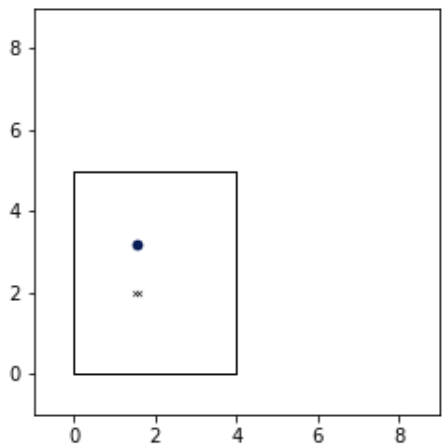


Figure 5.2: Top view of room B where the stars represent the microphones and the dot the speaker as described by Parchami et al. [110]

Using the Sabine equation (see Equation 2.24) the RT_{60} of a room can be calculated, where V is the volume of the room, S the surface area and $\bar{\alpha}$ the absorption coefficient. For example, the volume of room A is $320m^3$, the surface area is $288m^2$ (four wall of 8 x 5 and the floor and ceiling of 8 x 8), the 0.161 stands for the number of sound unit seconds travelled per metre i.e. the inverse of 320 m/s (the speed of sound). The walls of room A have an absorption coefficient of 0.35 (corresponding to a wall surfaced with 12.5mm thick acoustic plaster). This results in a RT_{60} of 0.51 seconds for room A. Similarly, room B with the same walls has a RT_{60} of 0.23 seconds.

5.4.3 Parameters

The STFT of the recordings from the TIMIT [3] is used as input with a window size of 100 frames, an overlap of 128 frames between the frames and with 1024 FFT units. In Equation 5.16 F_0 is the base frequency that needs to be detected corresponding to that of a male voice. All frequencies above this will be detected by this window size. F_s is the sample rate of the audio recordings. The recordings are sampled at 16kHz, these are downsampled where needed. Downsampling allows for an even comparison between the different corpora, however this does remove information from the recordings. This is multiplied by five, which is the size of the main lobe in the Hann window function.

$$\begin{aligned} \text{windowsize} &= \text{window lobe size} \times \frac{F_s}{\text{lowest detectable frequency}} \\ \text{windowsize} &= 5(F_s/F_0) \end{aligned} \tag{5.16}$$

The H_1, H_2 and H_s algorithms are non-learning algorithms (see Section 2.3.2) where the first two require no extra parameters. The H_s algorithm, on the other hand, is tested using 95%, 99.99% and 99.9999% of the data corresponding to λ values (see Equation 2.30) of 0.05, 0.0001 and 0.000001 (see Table 5.3). Therefore assuming that the majority of the so-called noise is described by respectively 5%, 0.01% and 0.0001% of the data. Next to this the scaling factors of s_x and s_y need to be determined. This is done on 10 randomly selected files of the TIMIT dataset.

For this process the s_y parameter was set to 1 and s_x ran on values between 1 and 100. Based on the performance of the algorithm in terms of the signal-to-artifact, signal-to-distortion and signal-to-interference ratios, the value of 31 for s_x which was then fixed and s_y was run on values between 1 and 100. This resulted in a value of 65 for the s_y parameter. These parameters are not used for H_s NTF because of the learning capabilities of the algorithm.

λ	0.05	0.0001	0.000001
s_x	0.31	0.31	0.31
s_y	65	65	65

Table 5.3: Parameters for the H_s algorithm

The remaining four algorithms (H_1 NTF, H_s NTF, Cauchy WPE and Cauchy MIMO WPE) are learning algorithms, the first two use nonnegative tensor factorisation to determine the dereverberation mask, the latter two use weighted prediction error for this.

5.4.4 Performance metrics

The performance of the seven algorithms is measured using six different measurements: the signal-to-artifact ratio (SAR), the signal-to-distortion ratio (SDR), the mir_eval signal-to-distortion ratio (SDR) (see Section 2.5), the scale invariant signal-to-distortion ratio (SISDR) [60], the signal-to-interference ratio (SIR), the signal-to-noise ratio (SNR) and the perceptual evaluation of speech quality (PESQ) [61]. The first two measurements give an indication of the noise present in the outcome of the algorithms whereas the latter is used to compare the performance of the algorithms against those in the literature.

5.4.5 Experiments

All algorithms are run on the TIMIT corpus simulated in room A. This is to compare the results of the algorithms with existing literature. Next to the original reverberation time of the room, the H_1, H_2 and H_s algorithms are also tested with an artificial reverberation time of 0.4, 1 and 1.5 seconds. This is to test the performance of the

algorithms in increasing difficulty. The longer reverberation time makes it more difficult for the algorithm to distinguish between speech and reverberation. However, this particular point where the algorithm cannot distinguish between speech and reverberation is different for each algorithm, using these reverberation times will give an indication of where this point is for these three algorithms. These tests run in a supervised and unsupervised setting in room A.

Both NTF algorithms (H_1 and H_s NTF) are tested with the room reverberation time of room A. The different WPE and MIMO WPE algorithms are also run on the TIMIT corpus simulated in room B. This is to compare the results with those described by Parchami et al. [110], who test six variants of the WPE algorithm. These are tested in room B with increasing RT_{60} times from 0.1 to 1 second (with increments of 0.1 seconds).

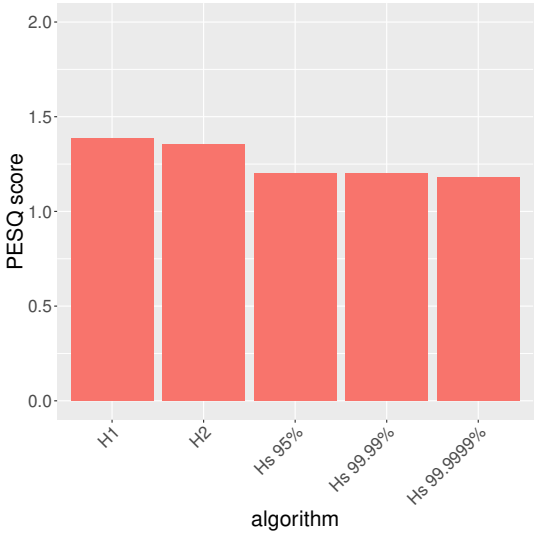
In addition to the room reverberation time, the WPE and MIMO WPE algorithms are also tested with a reverberation time of 0.1 to 1 second increasing the reverberation time by 0.1 seconds. These tests run in both rooms A and B. The WPE and MIMO WPE algorithms are also tested with different window functions to investigate the influence of these. Drude et al. [185] report using a Blackman window function for STFT. However, within the field of dereverberation this is not always explicitly mentioned. Therefore, WPE and MIMO WPE are tested with six different window functions (Bartlett-Hann, Bartlett, Blackman, Hamming, Hann and triangular) to see which performs best. The different MIMO WPE algorithms are run on the AC corpus to investigate how the algorithms perform in a realistic workshop environment.

5.5 Results

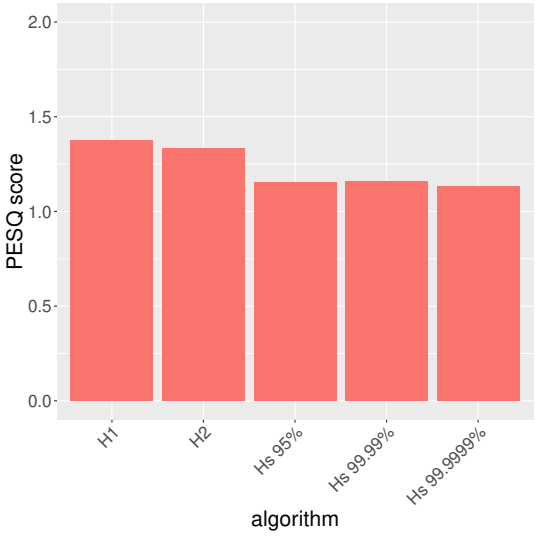
The seven algorithms are evaluated using room A where audio files from the TIMIT corpus are played. The performance of these algorithms is displayed in terms of the PESQ, SAR, SDR, SISDR, SIR and SNR. For all these measurements, it holds that the higher the value the better the algorithms perform. For the WPE algorithms the time it takes for the algorithm to run is also presented.

5.5.1 H_1, H_2 and H_s algorithms

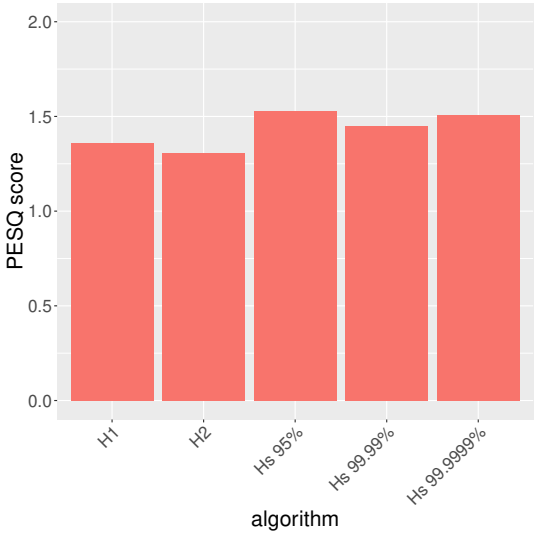
The H_s algorithm performs worse in terms of speech quality when a higher percentage of the data is being used. This is noticeable with the lower RT_{60} times (see Figures 5.3a and 5.3b) but this difference is less noticeable in the higher RT_{60} times. In general when a higher percentage of data is used then it is more likely that it contains noise thus degrading the overall speech quality. However, in terms of distortion the performance is similar to that of the H_1 and H_2 algorithms. When the RT_{60} time is increased to 1 second, the H_s algorithm starts to outperform the H_1 and H_2 algorithm (see Figures 5.3c and 5.4c) according to the PESQ score but this stops when the RT_{60} is increased to 1.5 seconds (see Figures 5.3d and 5.4d). This shows that the H_s algorithm produces a better speech quality with higher reverberation times than the H_1 and H_2 algorithms. However with a reverberation time of 1.5 seconds, the performance is similar due to the lack of noise present in the recordings.



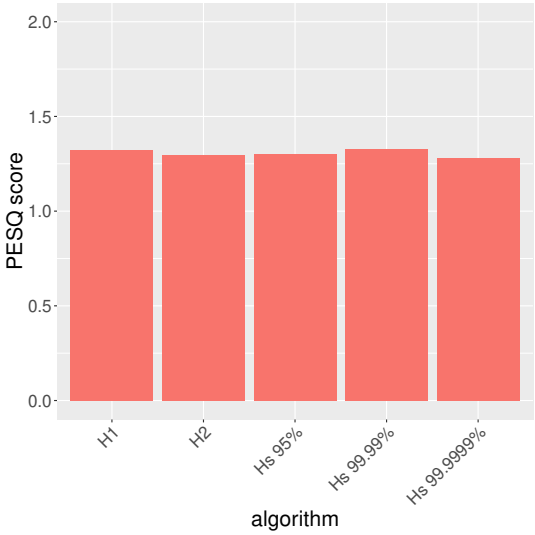
(a) Room A's own reverberation time (0.51 seconds)



(b) Reverberation time of 0.4 seconds

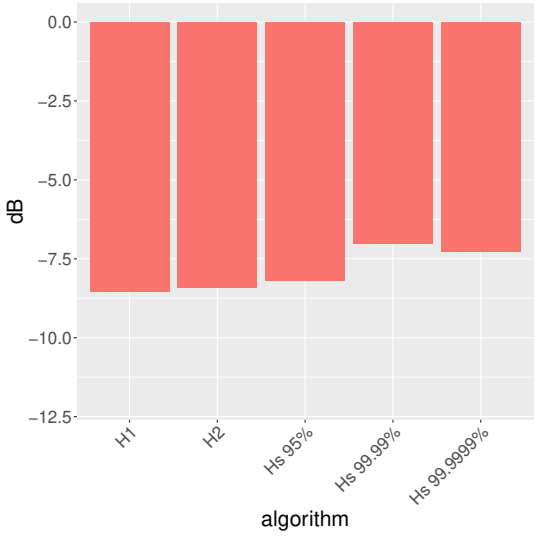


(c) Reverberation time of 1 second

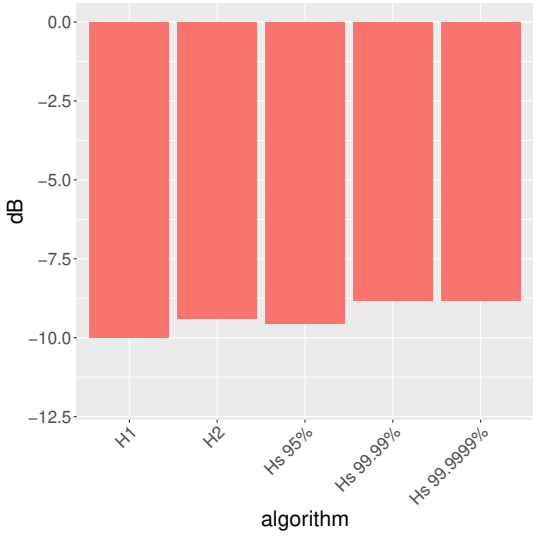


(d) Reverberation time of 1.5 seconds

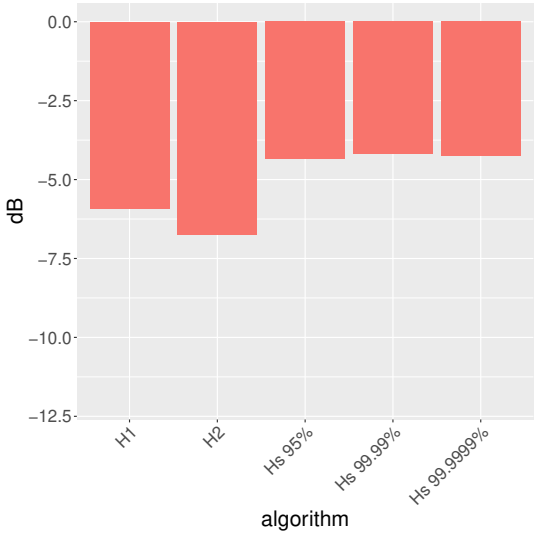
Figure 5.3: A comparison of the PESQ results of the supervised non-learning correlation algorithms with different reverberation times applied to room A



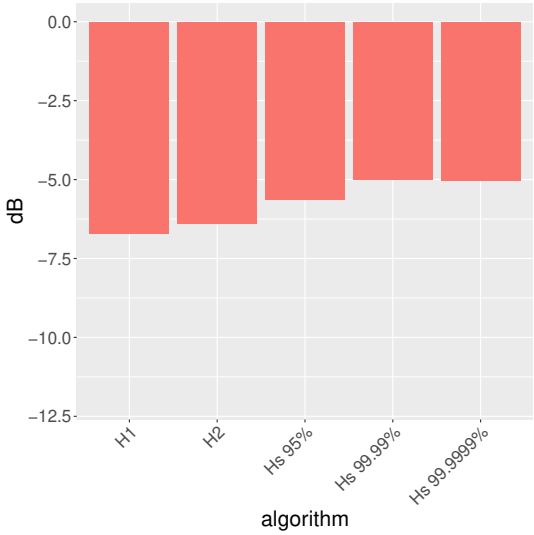
(a) Room A's own reverberation time (0.51 seconds)



(b) Reverberation time of 0.4 seconds

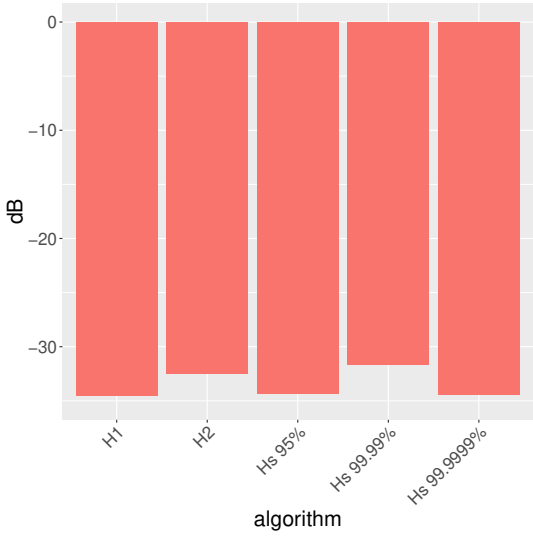


(c) Reverberation time of 1 second

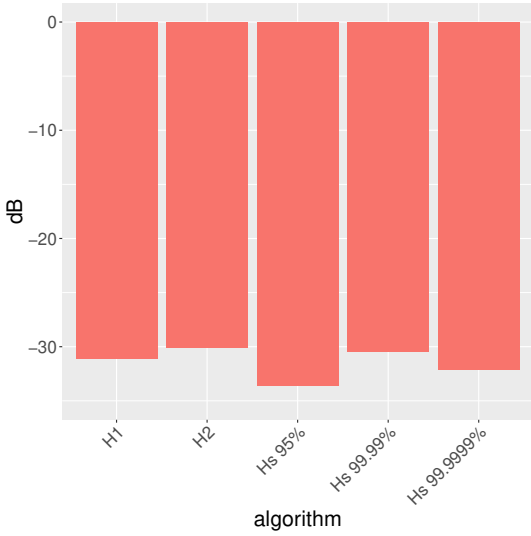


(d) Reverberation time of 1.5 seconds

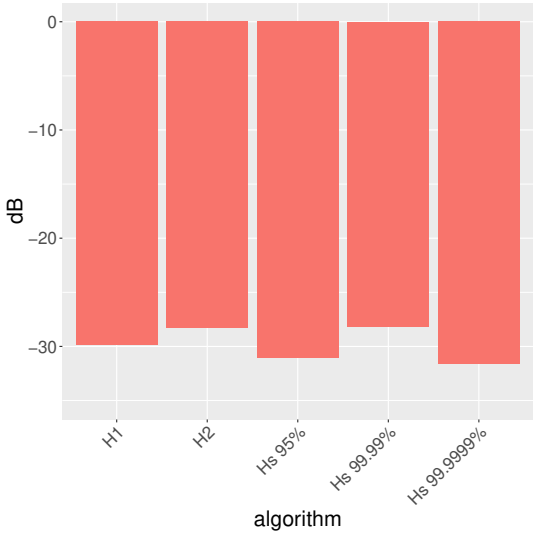
Figure 5.4: A comparison of the signal-to-distortion results of the supervised non-learning correlation algorithms with different reverberation times applied to room A



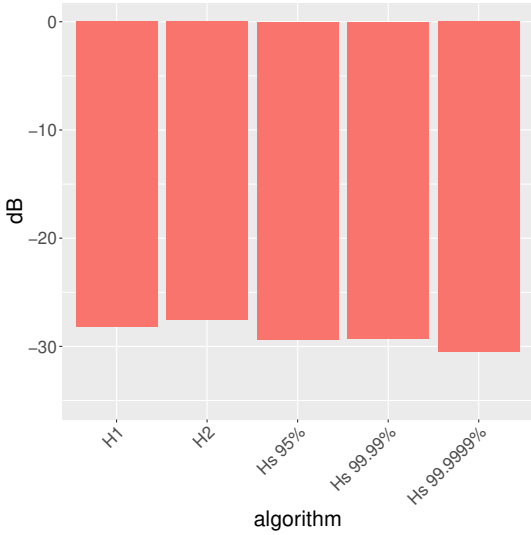
(a) Room A's own reverberation time (0.51 seconds)



(b) Reverberation time of 0.4 seconds



(c) Reverberation time of 1 second

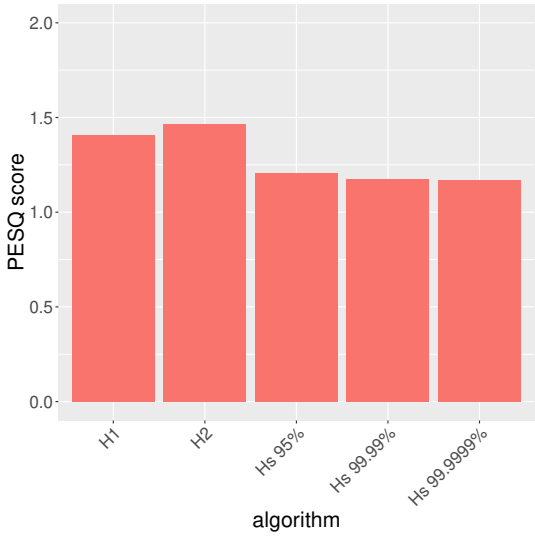


(d) Reverberation time of 1.5 seconds

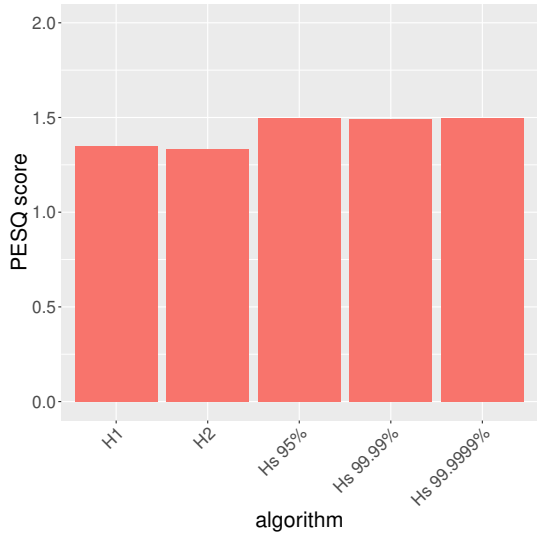
Figure 5.5: A comparison of the scale-invariant signal-to-distortion results of the supervised non-learning correlation algorithms with different reverberation times applied to room A

In the blind dereverberation case, which does not have a reference signal (see Figure 5.6), it does not make a difference if more data is included. The H_s algorithm shows equal performance when 95% of the data is used as when 99.9999% of the data is being used. However when the RT_{60} time is increased to 0.4 seconds, then adding more data improves the resulting signal (see Figure 5.6b). This pattern is not noticeable with an RT_{60} time of 1 second (see Figures 5.6c, 5.7c and 5.8c) but the pattern returns when the RT_{60} time is set to 1.5 seconds (see Figures 5.6d and 5.8d). The results show that when the reverberation time increases the difference

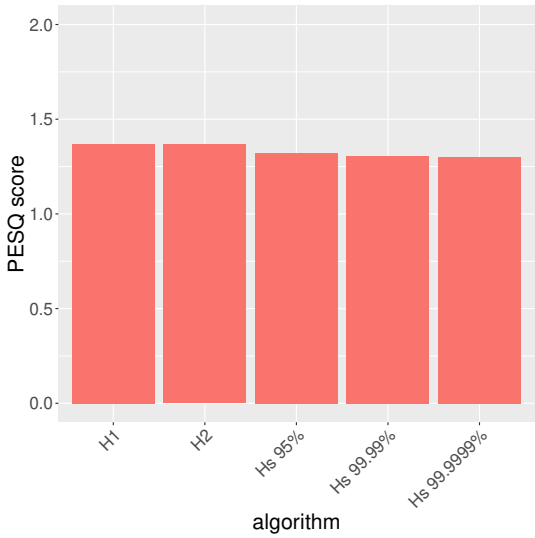
correlation between the microphones is higher. Also, it is more difficult to find a correlation between a reverberant and non-reverberant signal.



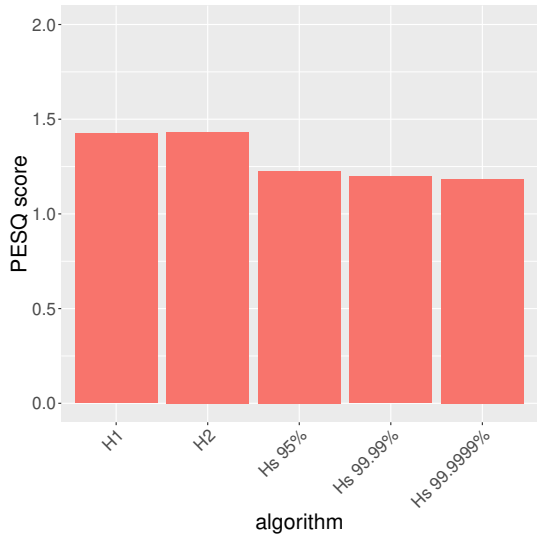
(a) Room A's own reverberation time (0.51 seconds)



(b) Reverberation time of 0.4 seconds

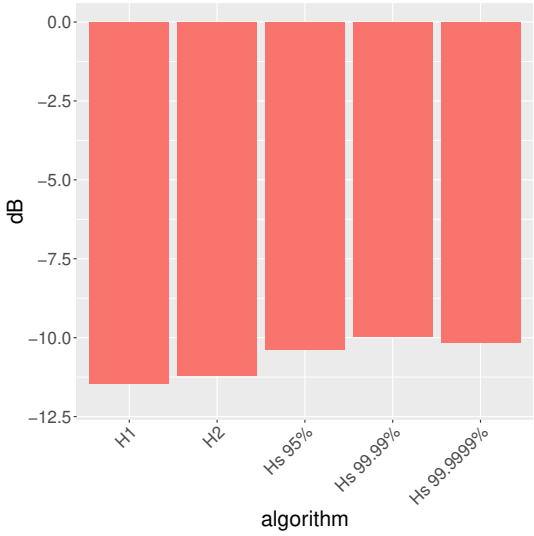


(c) Reverberation time of 1 second

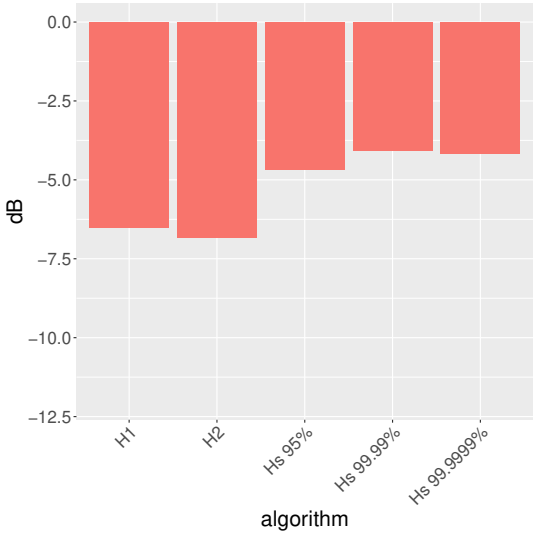


(d) Reverberation time of 1.5 seconds

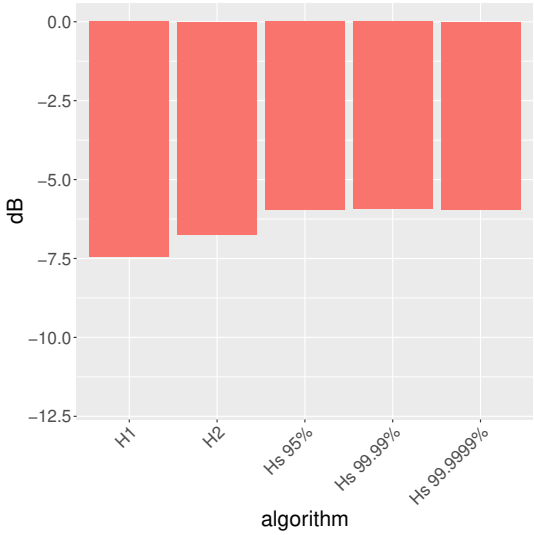
Figure 5.6: A comparison of the PESQ results of the unsupervised non-learning correlation algorithms with different reverberation times applied to room A



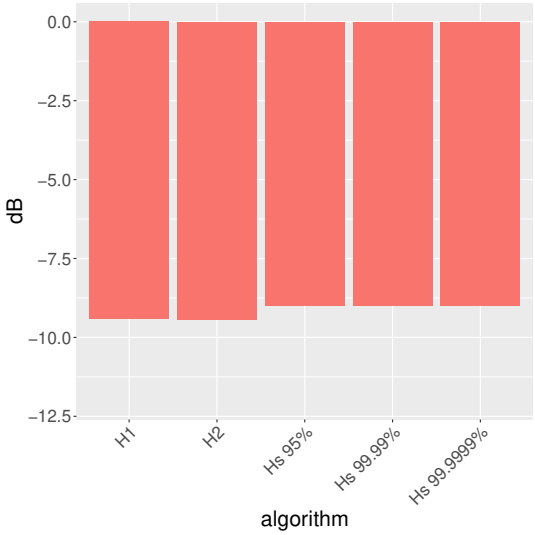
(a) Room A's own reverberation time (0.51 seconds)



(b) Reverberation time of 0.4 seconds

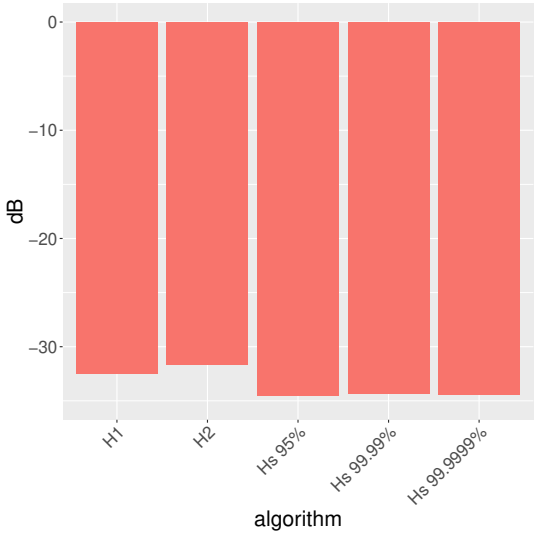


(c) Reverberation time of 1 second

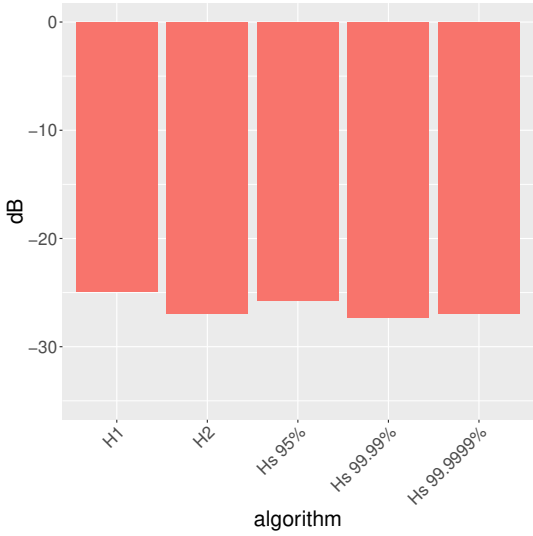


(d) Reverberation time of 1.5 seconds

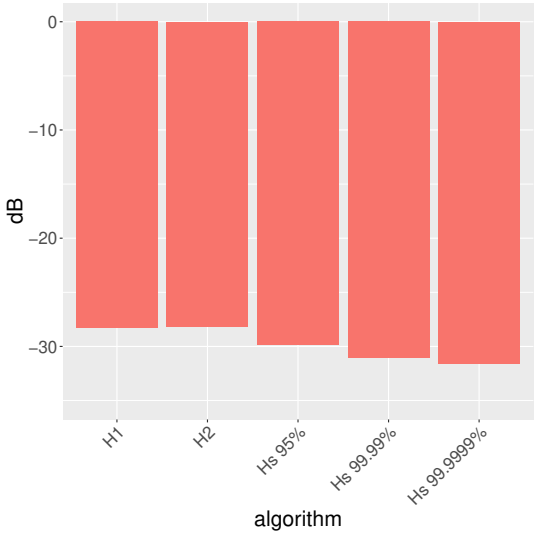
Figure 5.7: A comparison of the signal-to-distortion results of the unsupervised non-learning correlation algorithms with different reverberation times applied to room A



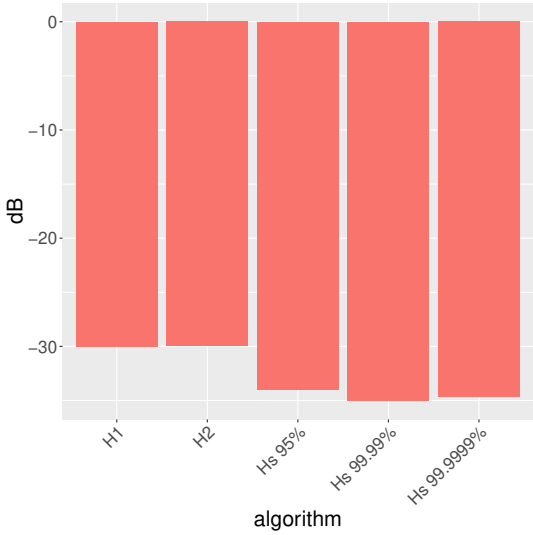
(a) Room A's own reverberation time (0.51 seconds)



(b) Reverberation time of 0.4 seconds



(c) Reverberation time of 1 second



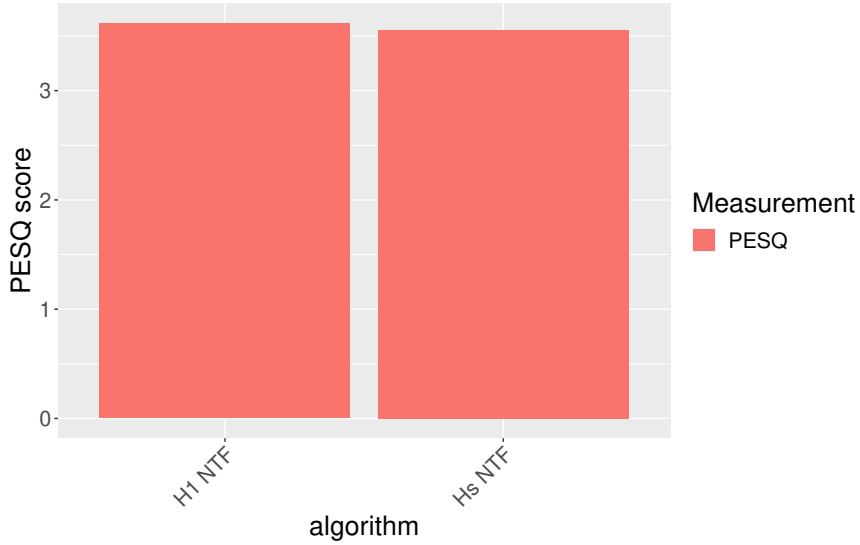
(d) Reverberation time of 1.5 seconds

Figure 5.8: A comparison of the scale-invariant signal-to-distortion results of the unsupervised non-learning correlation algorithms with different reverberation times applied to room A

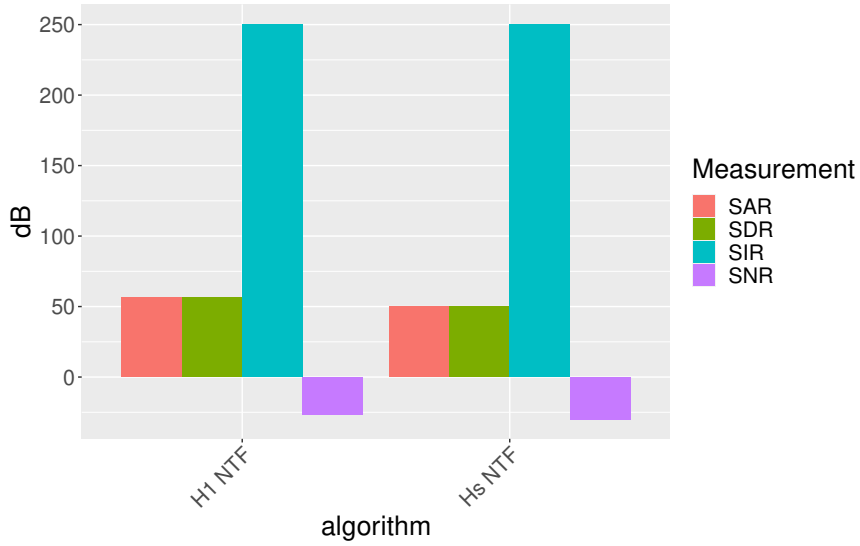
5.5.2 H_1 NTF and H_s NTF algorithms

The algorithm ran only in a supervised method with the difference between the H_1 NTF and the H_s NTF algorithm being minimal. This shows that there is no improvement in using a different way of calculating the correlation between microphones. Having the H_s NTF algorithm use 99.9999% of the data is giving a similar performance to the H_1 NTF algorithm showing that in this case both algorithms are able to remove the reverberation and produce comparative results. Showing that

the H_s NTF is able to remove the reverberation with similar performance as the existing H_1 NTF technique.



(a) The PESQ value of both NTF algorithms



(b) The SAR, SDR, SIR and SNR values of both NTF algorithms

Figure 5.9: The performance of the correlation-based dereverberation algorithm on the TIMIT recordings in room A with a RT_{60} of 0.51 seconds

5.5.3 WPE and MIMO WPE

Evaluating six window functions

Parchami et al. [110] and Nakatani et al. [26] do not explicitly describe the windowing function used for WPE. Because the Hann window function is the most popular window function used with speech, the assumption is that Parchimi et al. use the Hann window function. However, MIMO WPE method uses a Blackmann

window function (see Drude et al. [185]). This window function should be able to deal better with smearing of frequencies, thus creating a more defined spectrogram (see Section 2.2.3). Therefore, six different window functions (Blackman, Hann, Hamming, Bartlett-Hann, Bartlett and triangular) are compared on PESQ, SISDR, SDR, SDR_{mir} and time. The results from these measurements are based on 10 randomly selected files from the TIMIT corpus and have been run in room B to create a comparison with Parchami et al. [110] to see which performs best and is the fastest.

When the different windows are compared, the results show that the Blackman window is the worst performing for the MIMO WPE algorithm (see Figure 5.10a). The Bartlett-Hann, Bartlett and triangular windows are the best performing.

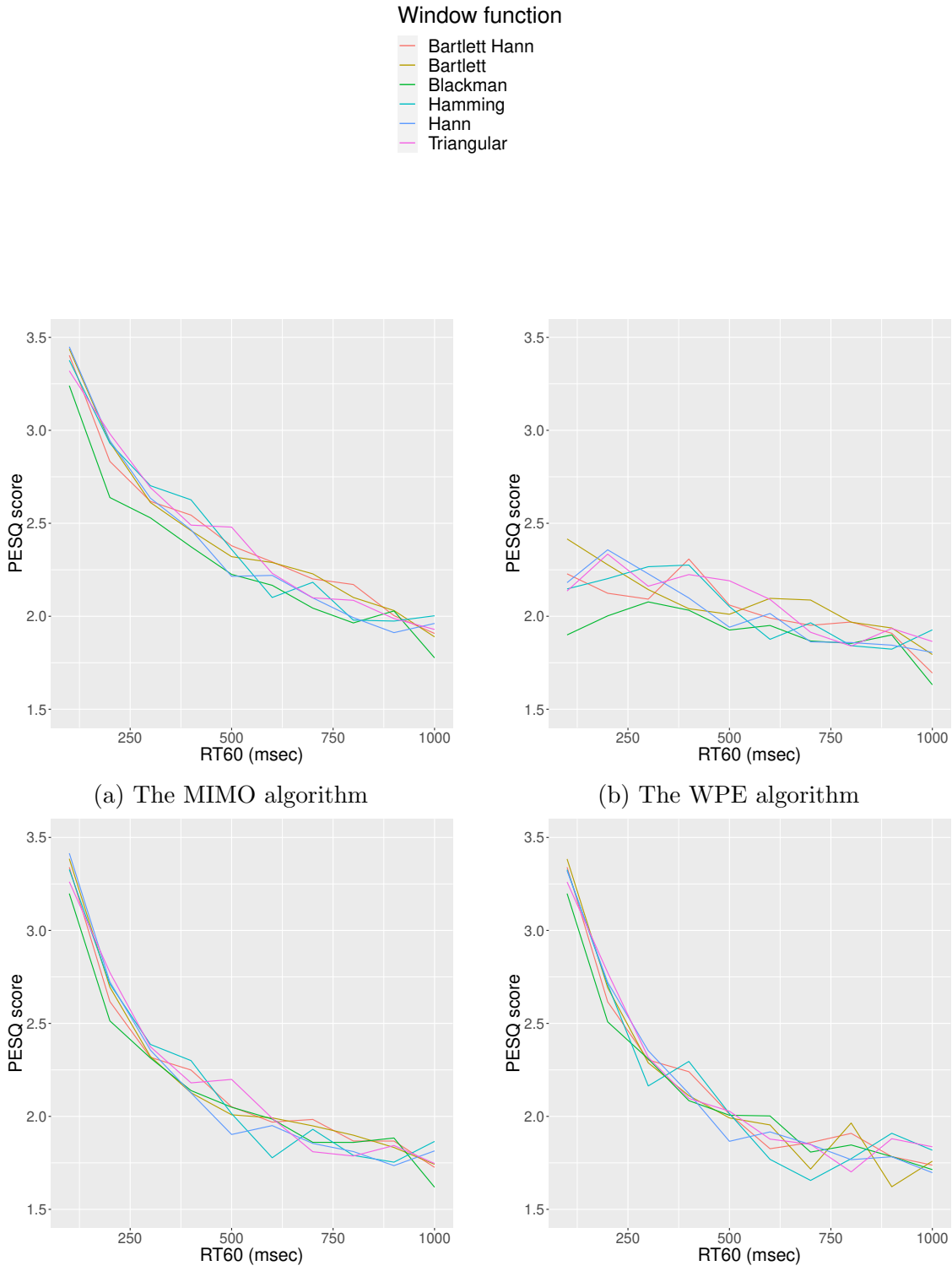
Looking at the different windows for WPE a similar trend is seen (see Figure 5.10b). However, with the short reverberant times (200 and 300 ms specifically) the Hann window performs better than the Bartlett-Hann window. A similar pattern can be seen for the two modified versions (see Figures 5.10c and 5.10d).

The SDR_{mir} shows a similar pattern for WPE and MIMO WPE (see Figure 5.11b and 5.11a). However, the Cauchy MIMO v2 algorithm performs better with a Blackman window on the middle RT_{60} times (between 500 and 700 ms) (see Figure 5.11c).

The other two measurements (SISDR and SDR) do not give conclusive results (see Figures 5.12 and 5.13). Looking at the time it takes to run the algorithm, a Blackman window is quicker, closely followed by the Hamming and Hann windows. In some cases, the Bartlett and Bartlett-Hann windows are the slowest window functions used (see Figure 5.14).

Comparing Blackman and Hann window functions

Comparing the two original algorithms (WPE and MIMO) based on the two windows used in the papers, the results show that the MIMO algorithm outperforms WPE and that the Hann window outperforms Blackman window in PESQ performance (see Figure 5.15a). The same pattern is seen in the time it takes for the algorithm to run and the SDR_{mir} measurement (see Figures 5.15d and 5.15e). Again the SDR and the SISDR do not show a clear distinction between the windows or the algorithms (see Figures 5.15b and 5.15c).



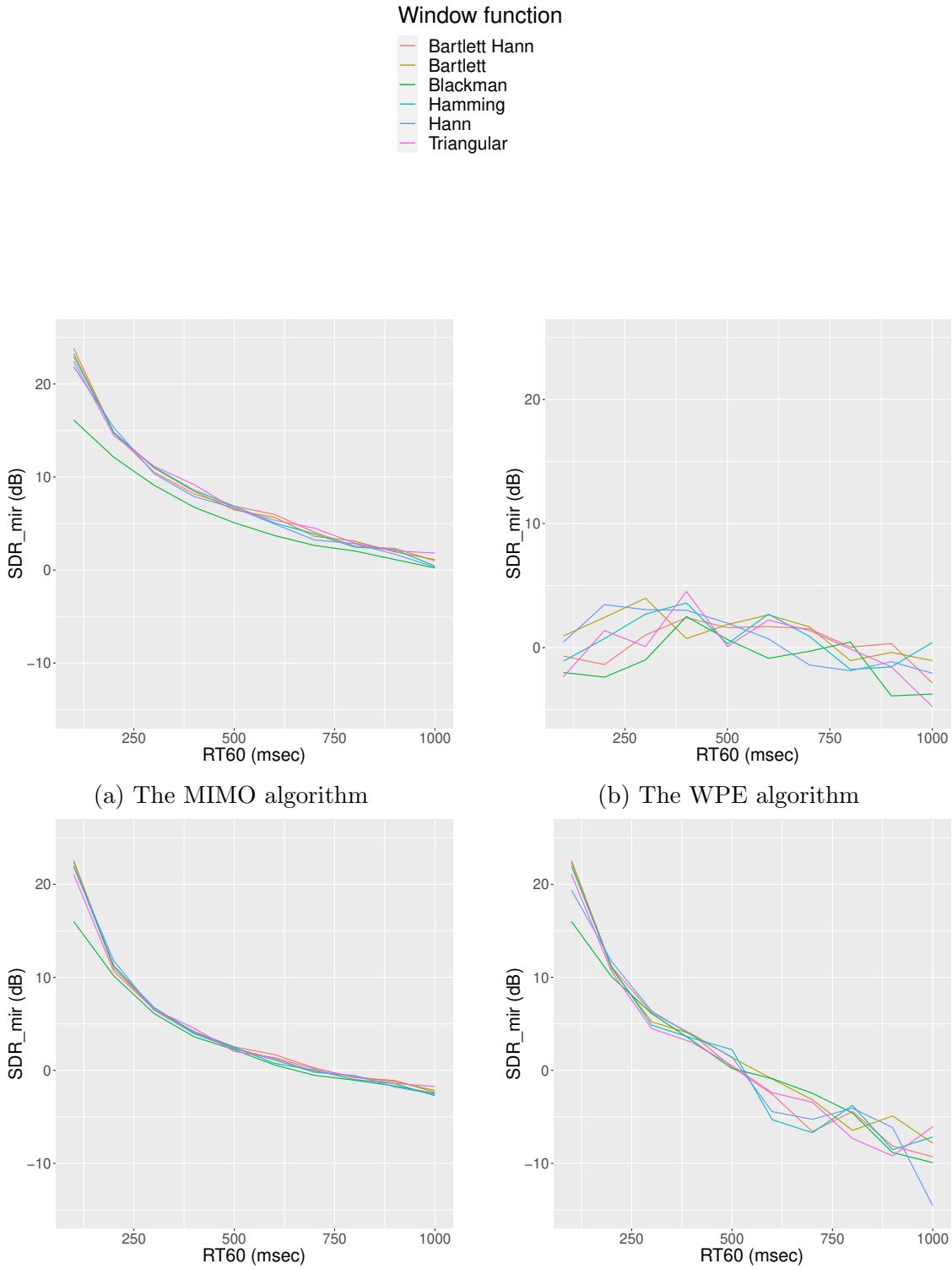
(a) The MIMO algorithm

(b) The WPE algorithm

(c) The Cauchy v2 MIMO WPE algorithm

(d) The Cauchy v2 WPE algorithm

Figure 5.10: The PESQ performance of the different MIMO and WPE algorithms using different window functions evaluated on room B



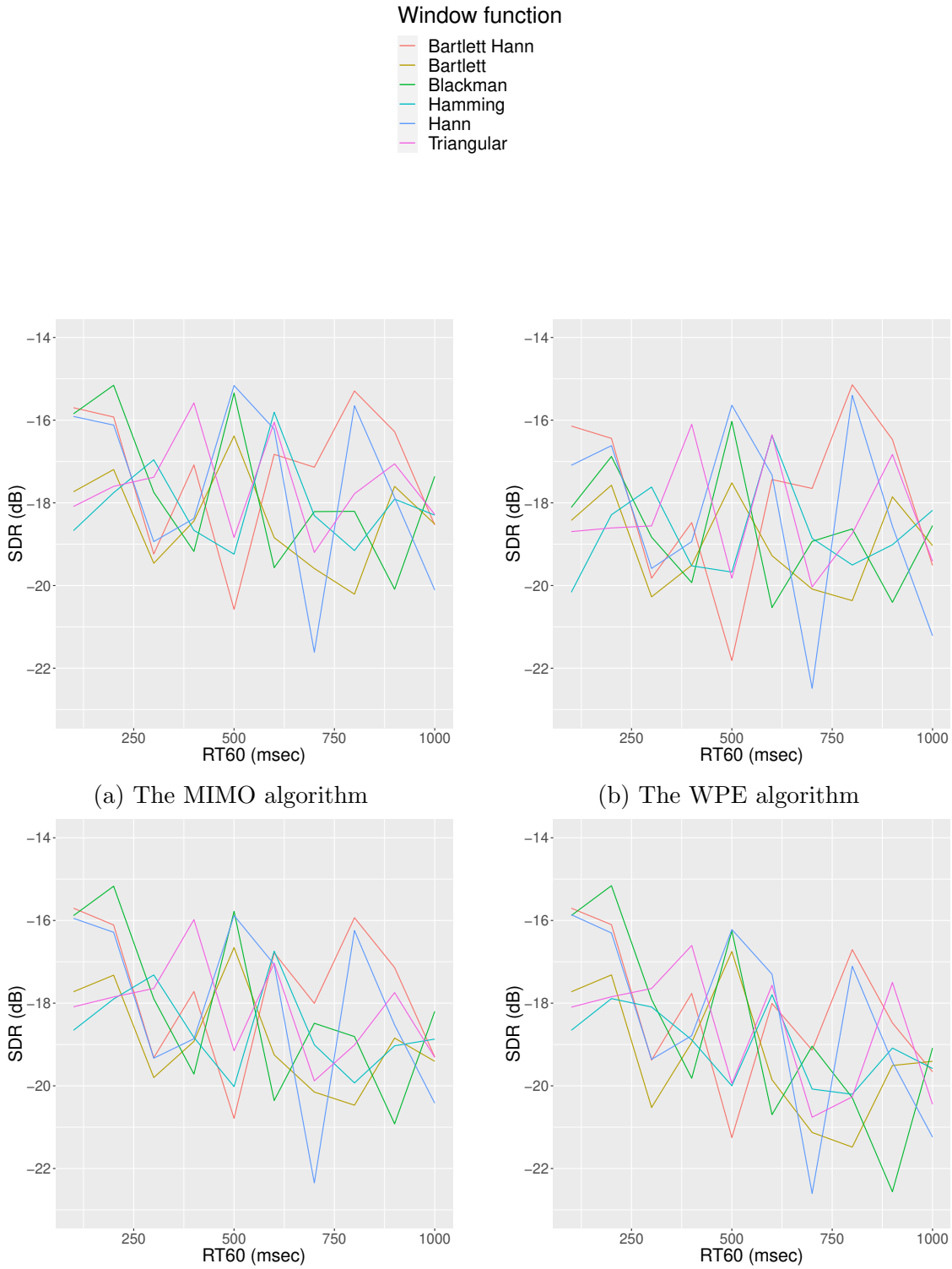
(a) The MIMO algorithm

(b) The WPE algorithm

(c) The Cauchy v2 MIMO WPE algorithm

(d) The Cauchy v2 WPE algorithm

Figure 5.11: The SDR_{mir} performance of the different MIMO and WPE algorithms using different window functions evaluated on room B



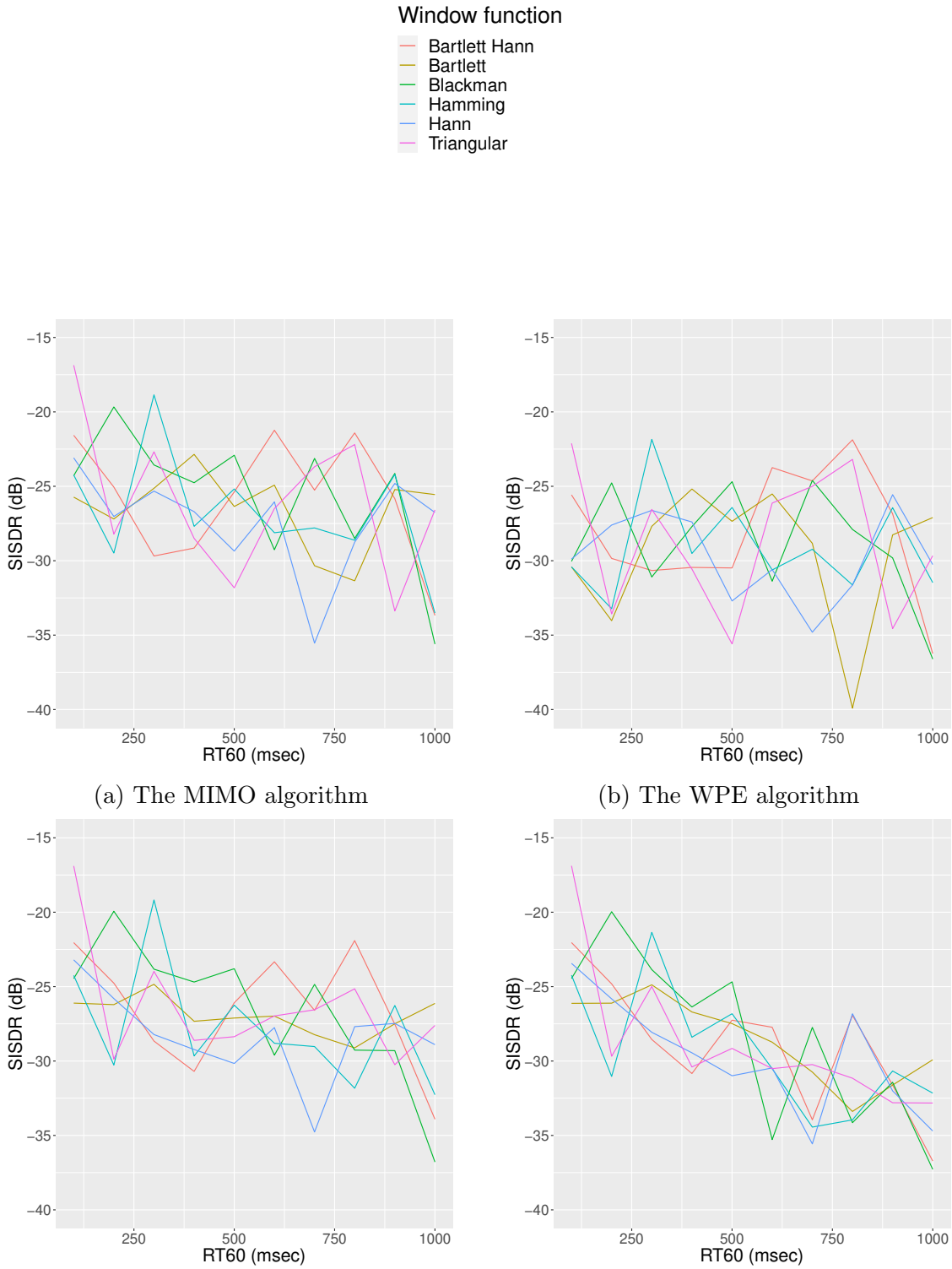
(a) The MIMO algorithm

(b) The WPE algorithm

(c) The Cauchy v2 MIMO WPE algorithm

(d) The Cauchy v2 WPE algorithm

Figure 5.12: The SDR performance of the different MIMO and WPE algorithms using different window functions evaluated on room B



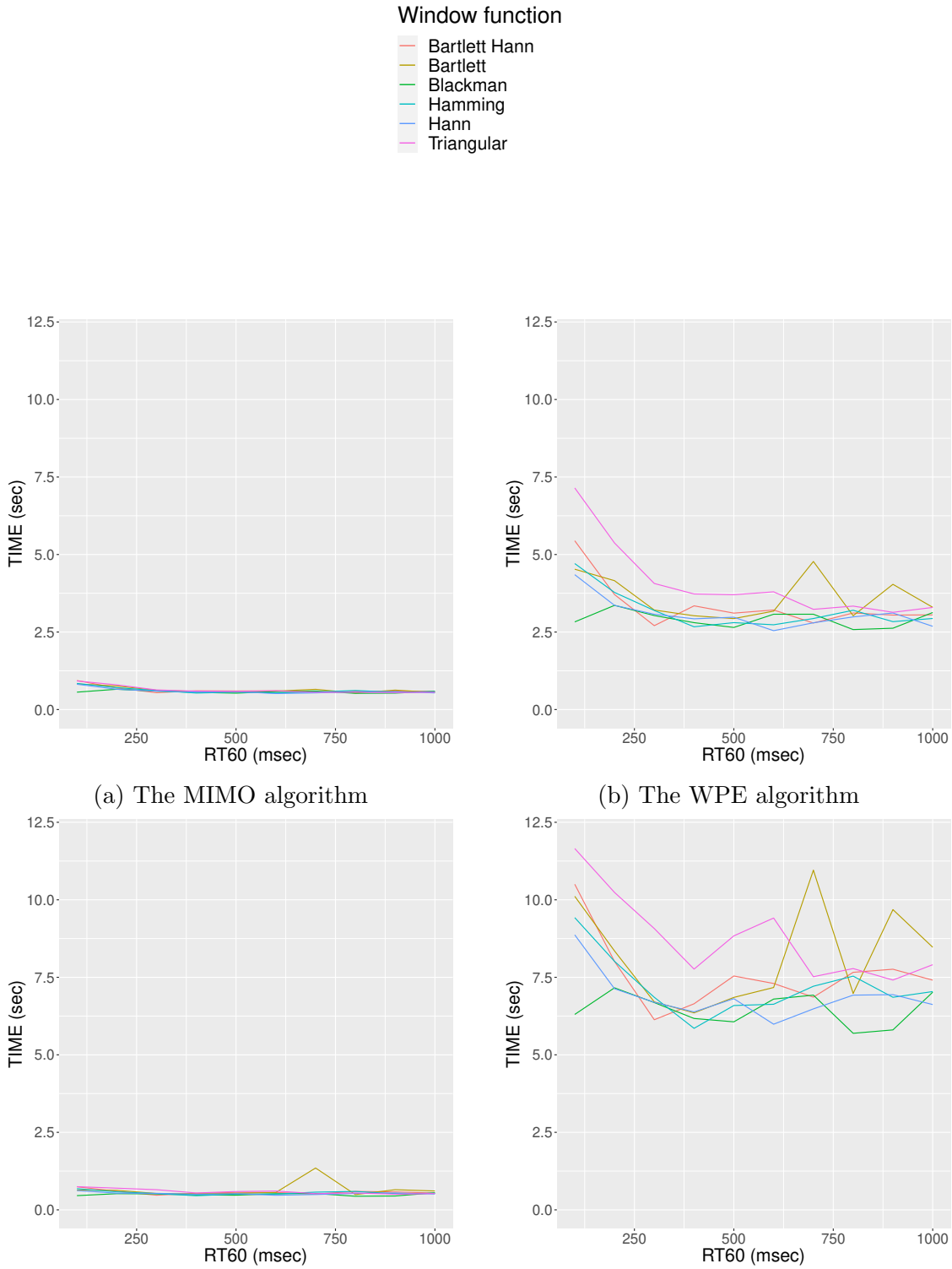
(a) The MIMO algorithm

(b) The WPE algorithm

(c) The Cauchy v2 MIMO WPE algorithm

(d) The Cauchy v2 WPE algorithm

Figure 5.13: The SISDR performance of the different MIMO and WPE algorithms using different window functions evaluated on room B



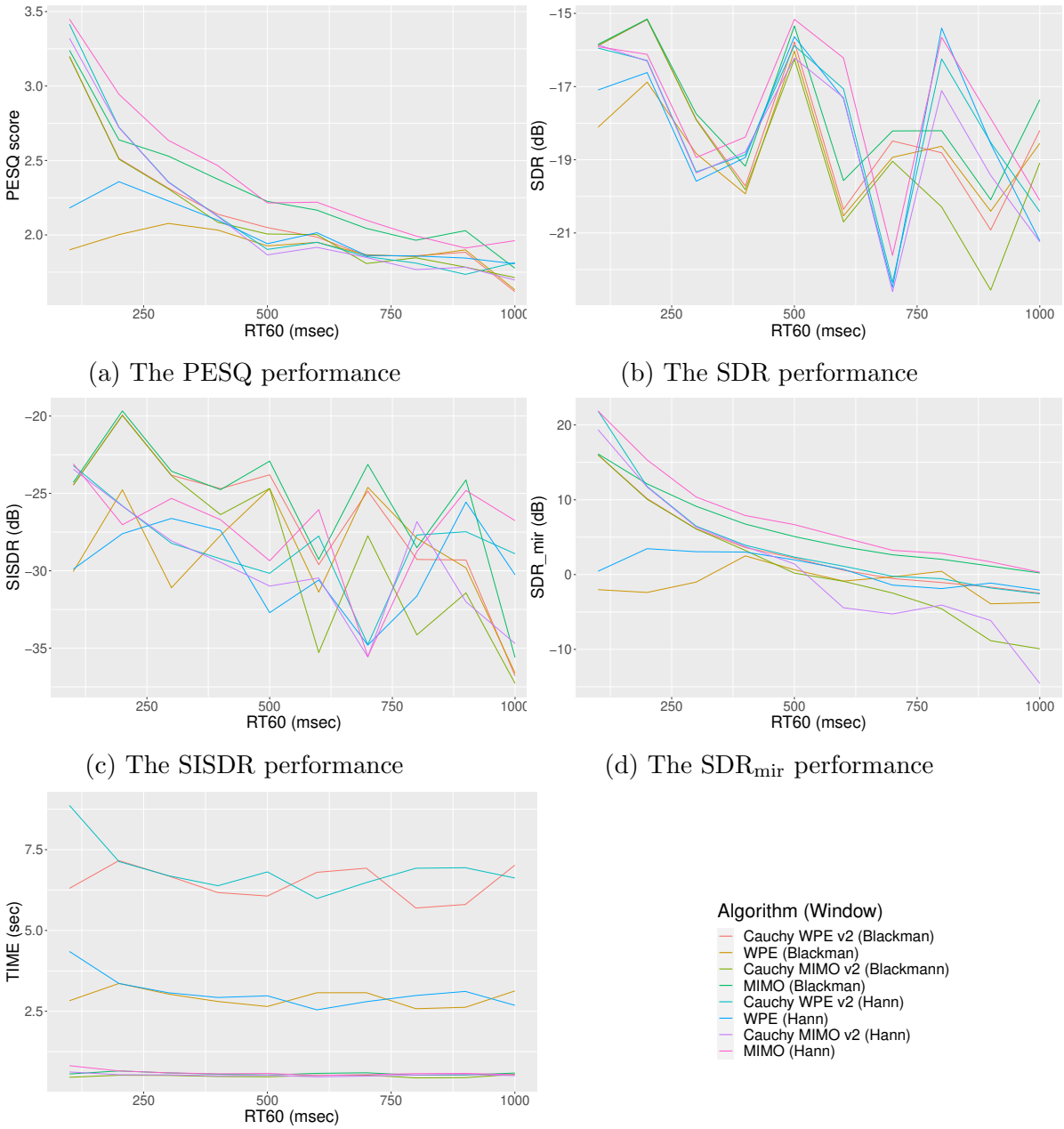
(a) The MIMO algorithm

(b) The WPE algorithm

(c) The Cauchy v2 WPE algorithm

(d) The Cauchy v2 MIMO WPE algorithm

Figure 5.14: The running time of the different MIMO and WPE algorithms using different window functions evaluated on room B



(a) The PESQ performance

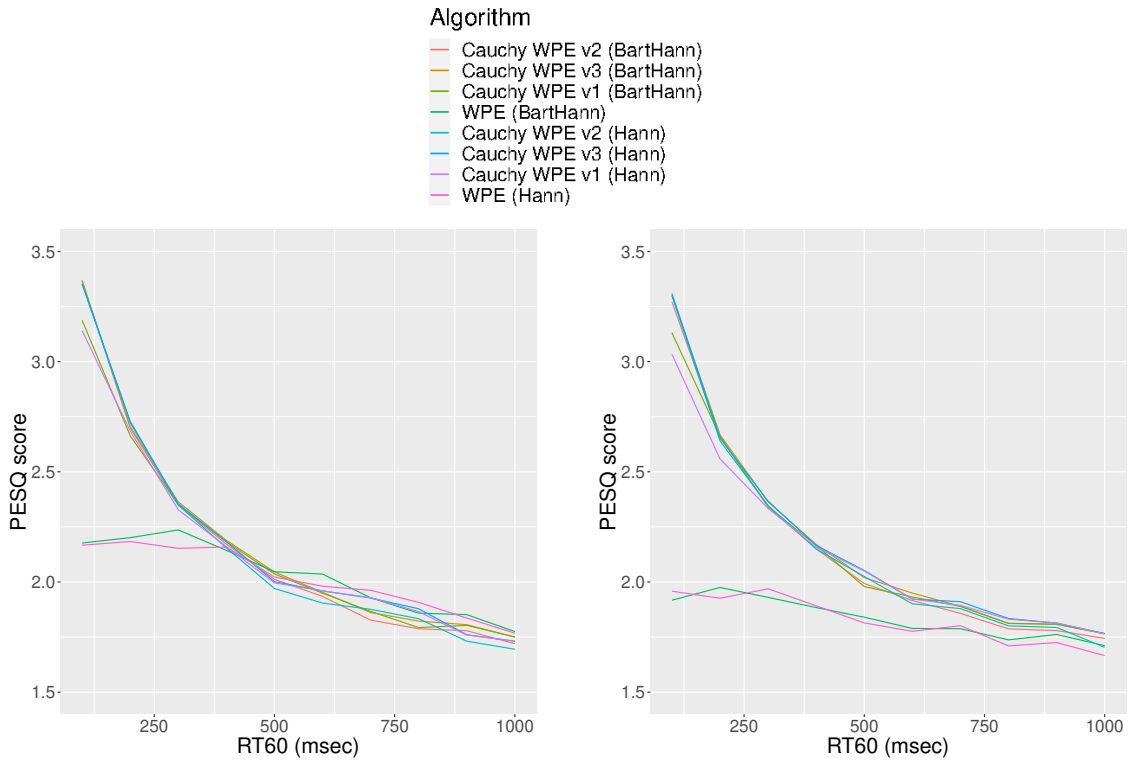
(b) The SDR performance

(c) The SISDR performance

(d) The $\mathrm{SDR}_{\mathrm{mir}}$ performance

(e) The running time

Figure 5.15: The performance of the different Hann and Blackman windowing functions on the different WPE, MIMO WPE, Cauchy v2 WPE and Cauchy v2 WPE algorithms evaluated on room B



(a) Applied to room A

(b) applied to room B

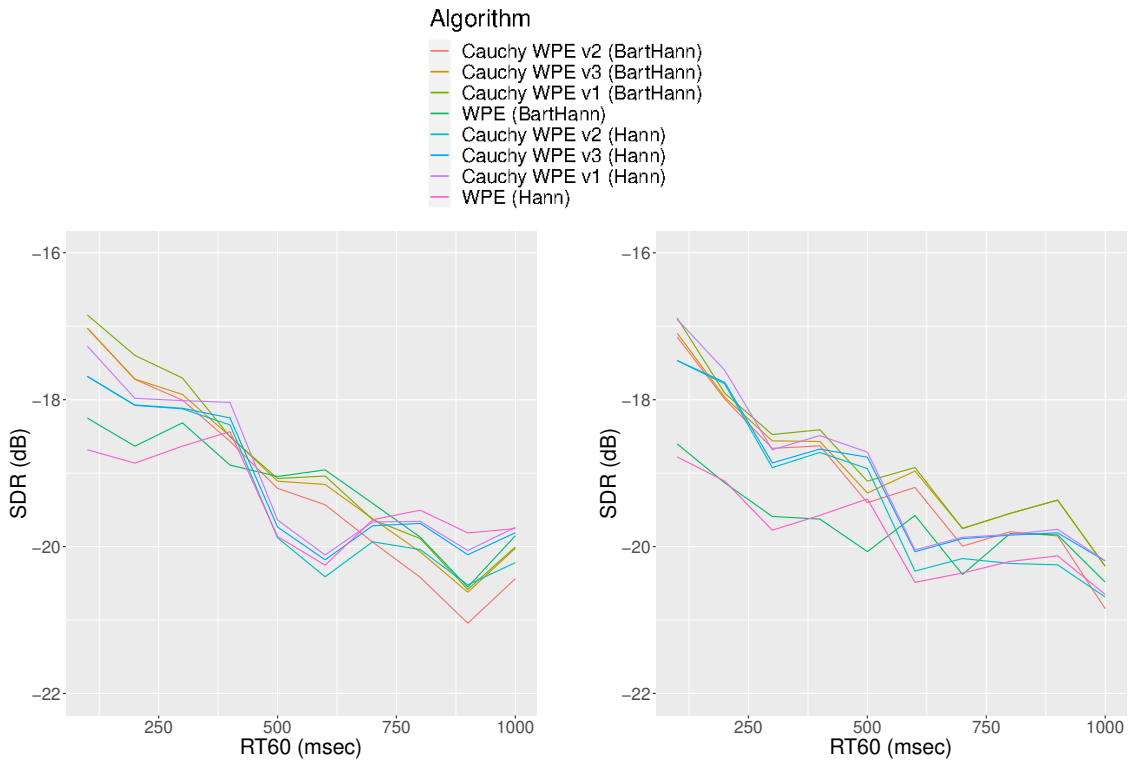
Figure 5.16: The PESQ performance of the Cauchy WPE and WPE algorithms using the Bartlett-Hann and Hann window functions evaluated on room A and B

Evaluating Bartlett-Hann and Hann window functions

When concentrating on the Bartlett-Hann and Hann on the WPE algorithm, the results show that the modifications outperform the original WPE algorithm especially at the lower RT_{60} times (see Figures 5.16 to 5.20). These results show the same pattern for both window functions. Looking specifically at the running time, the original algorithms outperform the modifications by a number of seconds with Cauchy v3 being the quickest. There is also a difference between the performance on the SDR and SISDR scale, which is more noticeable on the low RT_{60} times and only for Cauchy v1 (see Figures 5.17 and 5.18) which has the best result on the SDR scale and the worst of the three modifications on the SISDR scale.

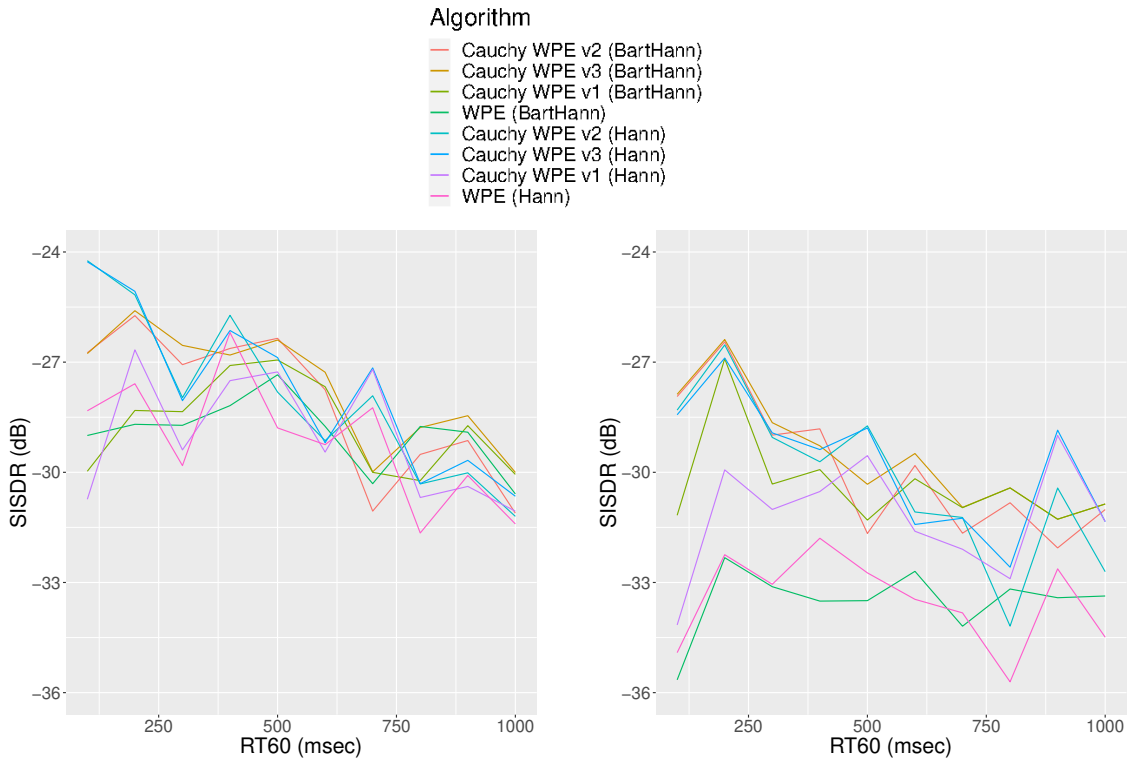
Evaluating the MIMO WPE algorithms on the TIMIT and AC corpora

When looking at how the four MIMO algorithms perform on 100 random files in room A, there is little difference compared to the 10 random files (see Figure 5.21).



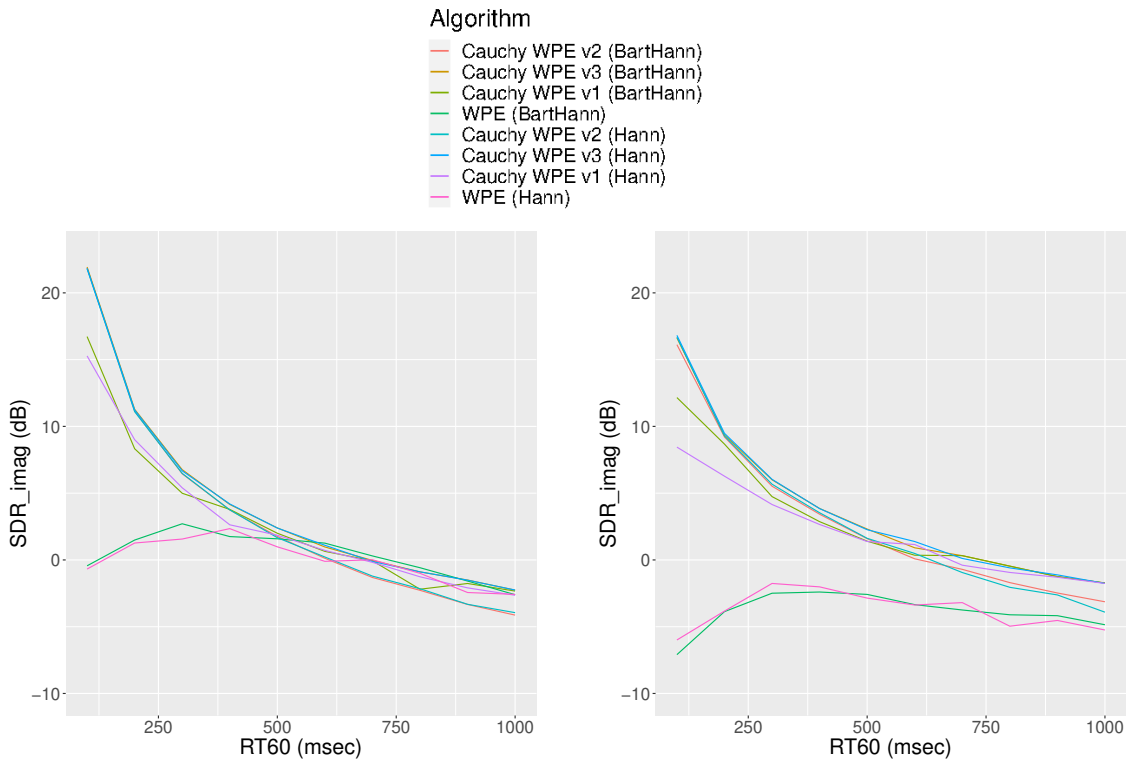
(a) applied to room A (b) applied to room B

Figure 5.17: The SDR performance of the Cauchy WPE and WPE algorithms using the Bartlett-Hann and Hann window functions evaluated on room A and B



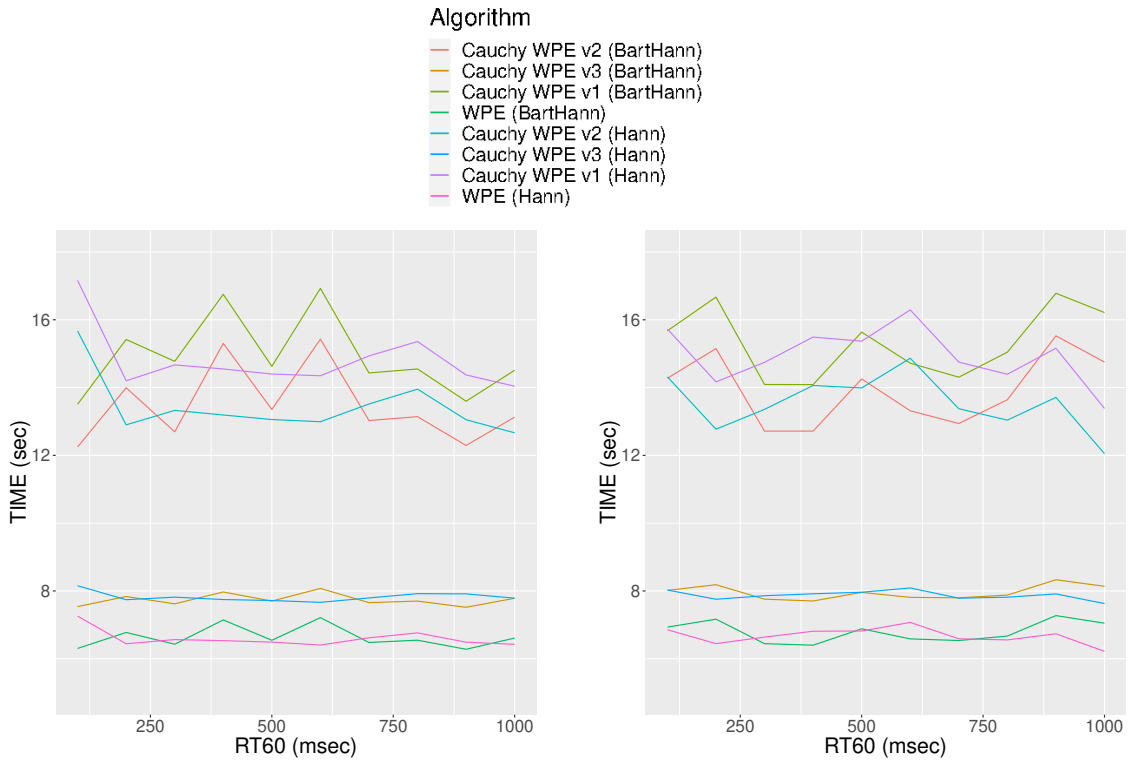
(a) applied to room A (b) applied to room B

Figure 5.18: The SISDR performance of the Cauchy WPE and WPE algorithms using the Bartlett-Hann and Hann window functions evaluated on room A and B



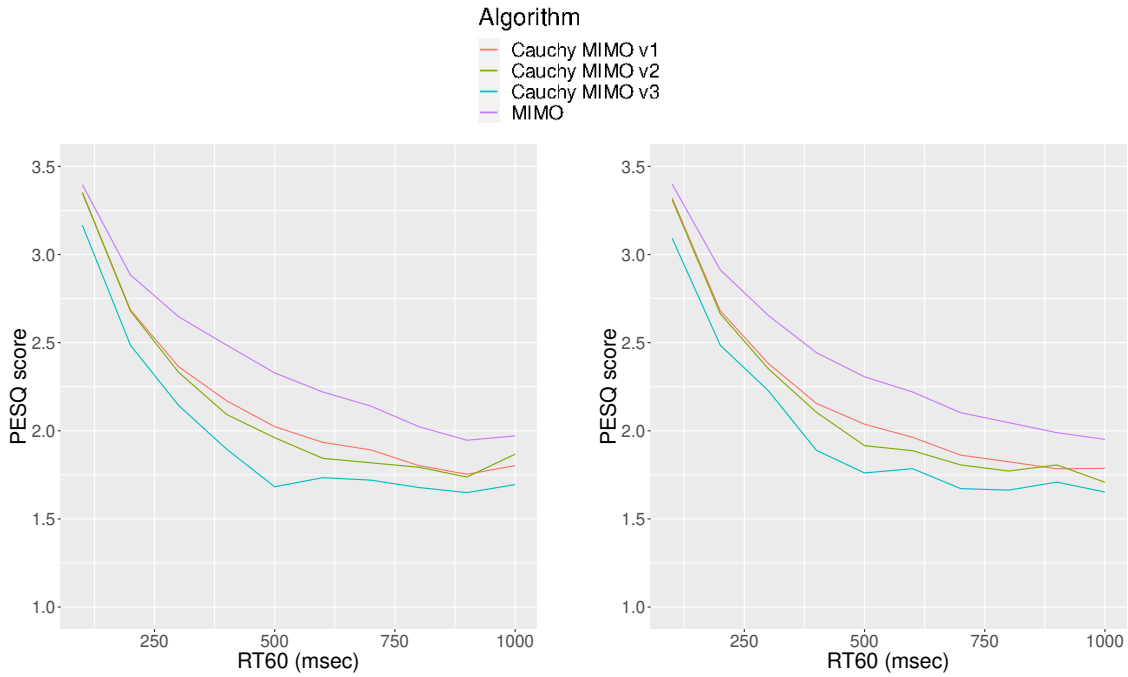
(a) applied to room A (b) applied to room B

Figure 5.19: The $\mathrm{SDR}_{\mathrm{mir}}$ performance of the Cauchy WPE and WPE algorithms using the Bartlett-Hann and Hann window functions evaluated on room A and B



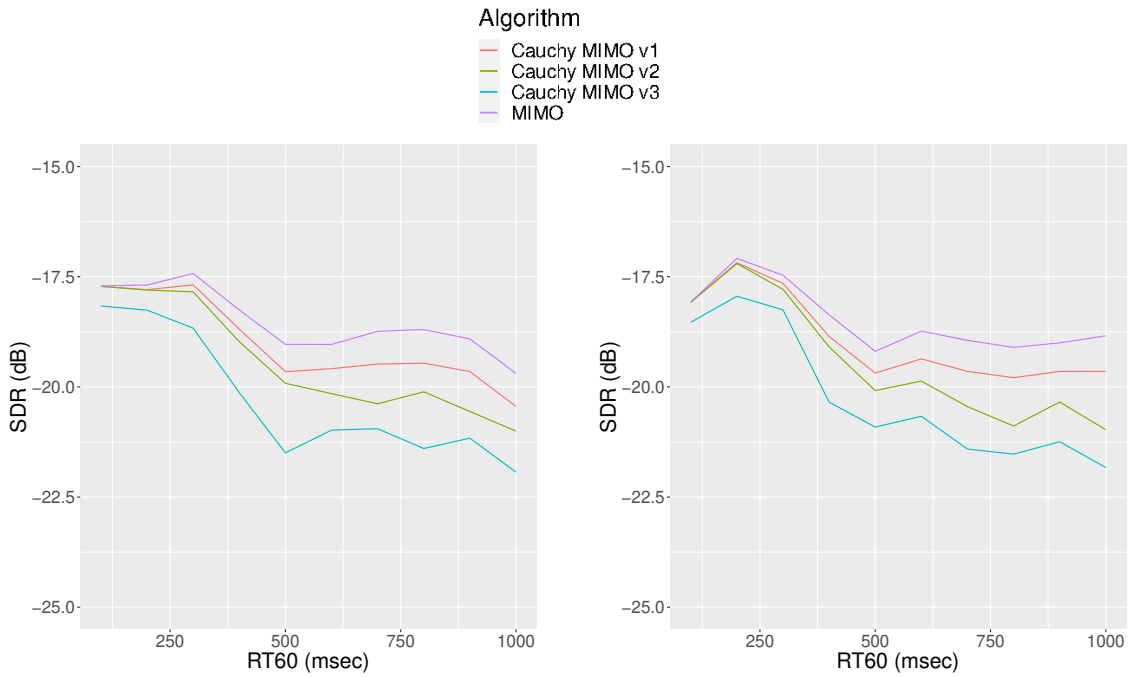
(a) applied to room A (b) applied to room B

Figure 5.20: The running time of the Cauchy WPE and WPE algorithms using the Bartlett-Hann and Hann window functions evaluated on room A and B



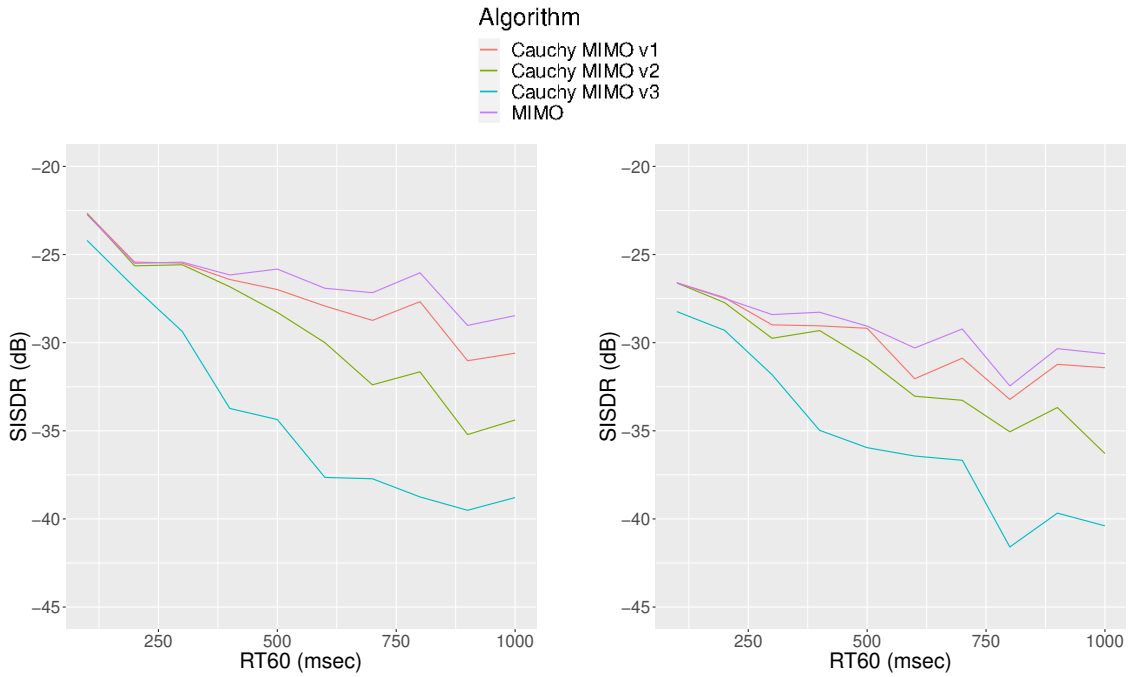
(a) applied to room A (b) Applied to room B

Figure 5.21: The PESQ performance of the Cauchy MIMO WPE and MIMO WPE algorithms using the Hann window function evaluated on room A and B



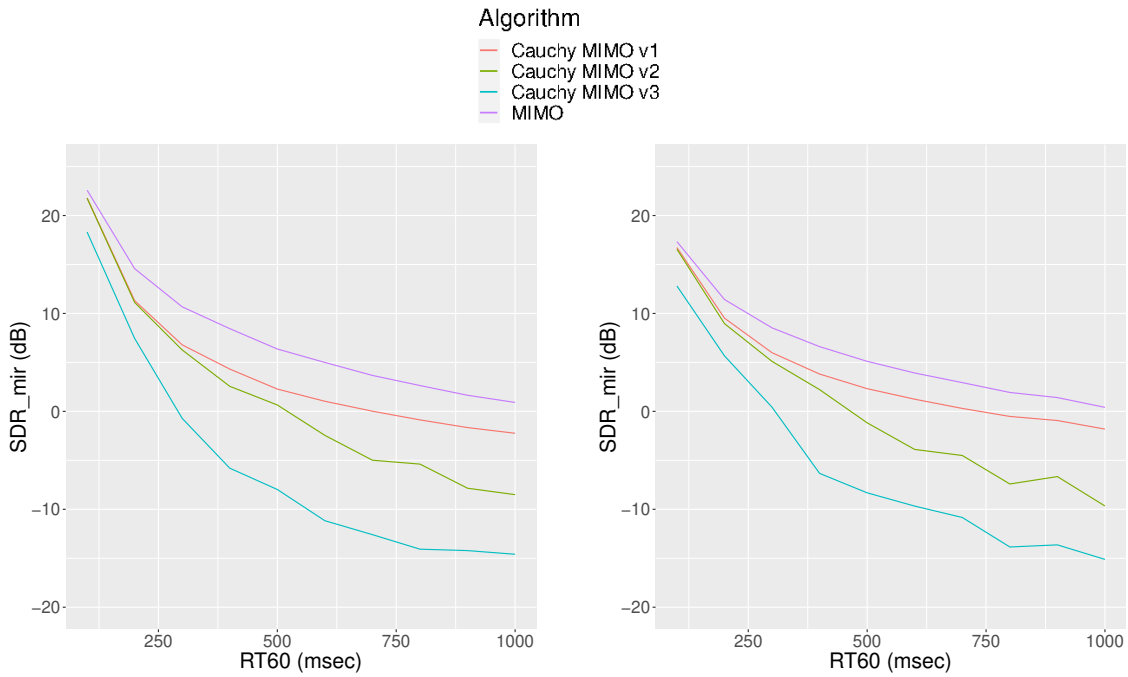
(a) applied to room A (b) Applied to room B

Figure 5.22: The SDR performance of the Cauchy MIMO WPE and MIMO WPE algorithms using the Hann window function evaluated on room A and B



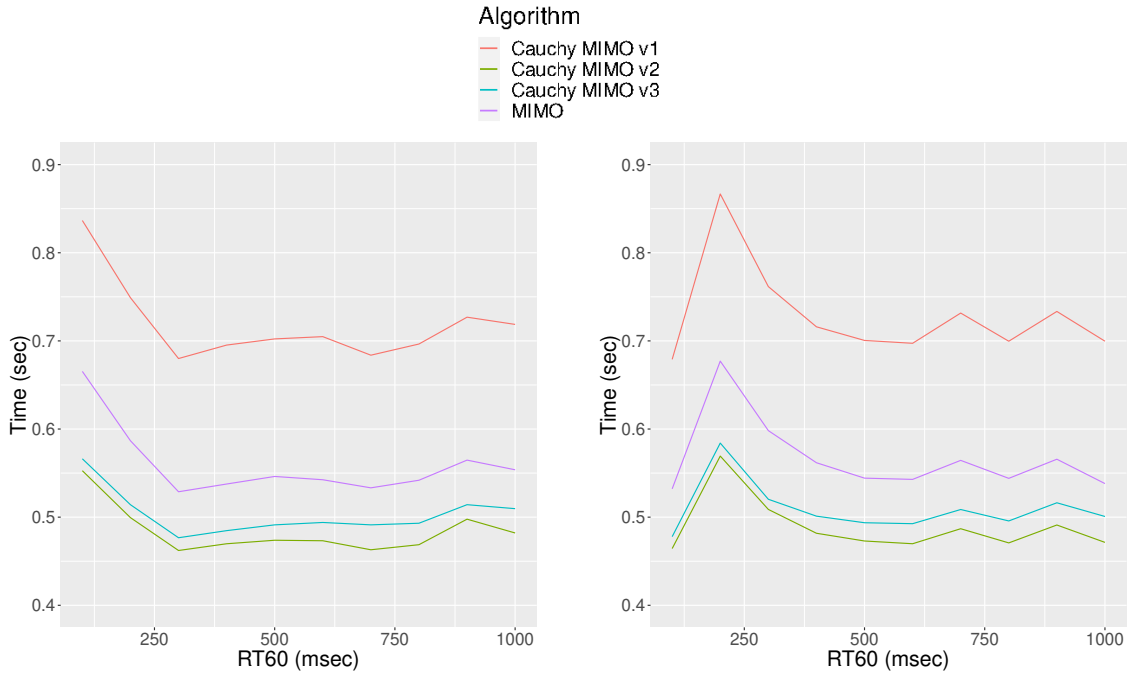
(a) applied to room A (b) Applied to room B

Figure 5.23: The SISDR performance of the Cauchy MIMO WPE and MIMO WPE algorithms using the Hann window function evaluated on room A and B



(a) applied to room A (b) Applied to room B

Figure 5.24: The $\mathrm{SDR}_{\mathrm{mir}}$ performance of the Cauchy MIMO WPE and MIMO WPE algorithms using the Hann window function evaluated on room A and B

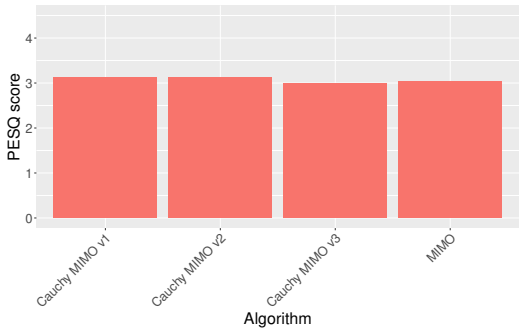


(a) applied to room A

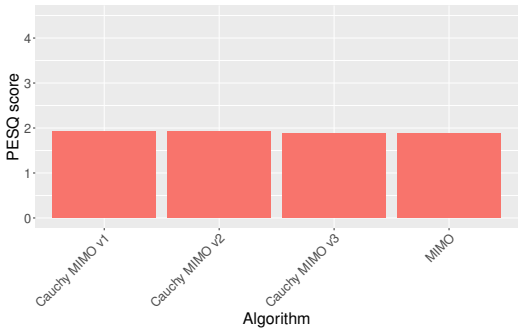
(b) Applied to room B

Figure 5.25: The running time of the Cauchy MIMO WPE and MIMO WPE algorithms using the Hann window function evaluated on room A and B

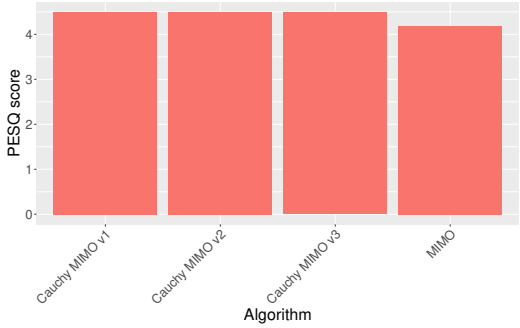
Also, the original MIMO algorithm outperforms the modifications on all the measurements except for the time (see Figure 5.25). When switching to room B there is little improvement meaning that the size of the room or the distance between source and microphone do not impact the performance of the algorithm (see Figure 5.22).



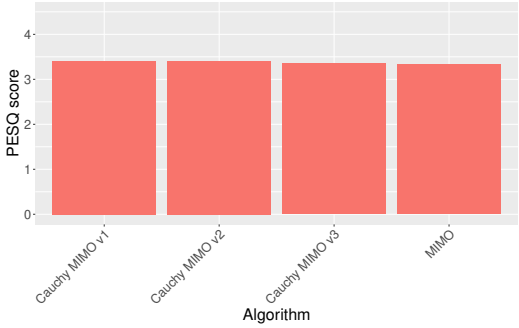
(a) The original dataset



(b) The no_noise dataset

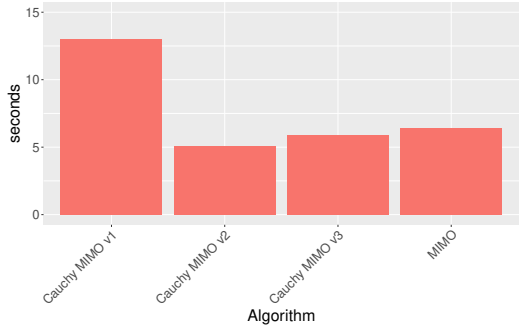


(c) The no_ echo dataset

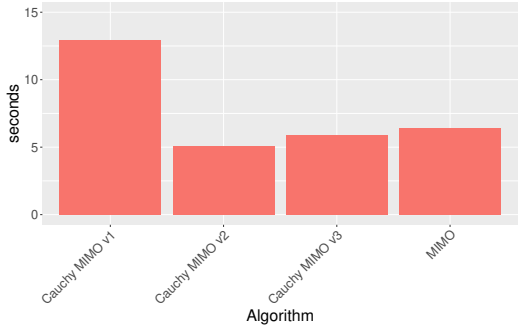


(d) The nonoise_echo dataset

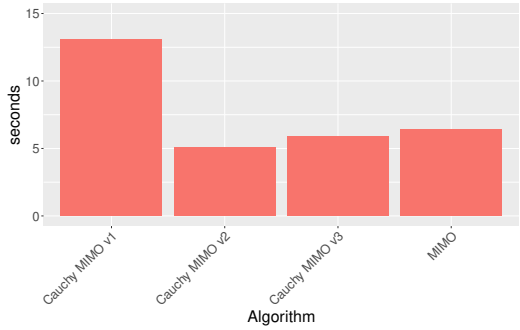
Figure 5.26: The PESQ performance of the four MIMO algorithms on the four datasets of the AC corpus.



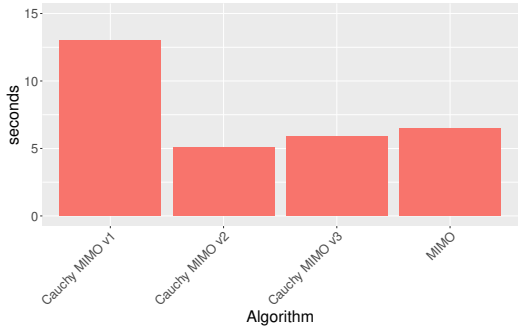
(a) The original dataset



(b) The no_noise dataset

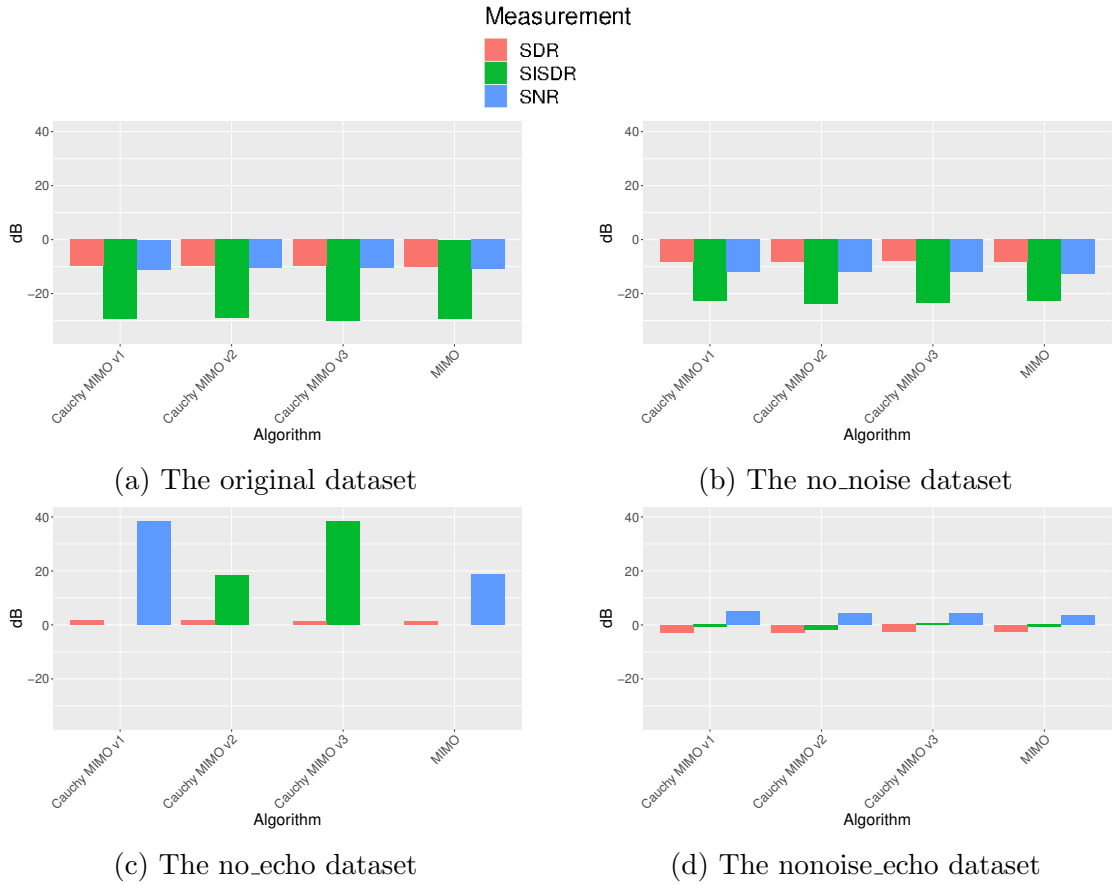


(c) The no_echo dataset



(d) The nonoise_echo dataset

Figure 5.28: The running time of the four MIMO algorithms on the four datasets of the AC corpus.



(a) The original dataset

(b) The no_noise dataset

(c) The no_echo dataset

(d) The nonoise_echo dataset

Figure 5.27: The SDR, SISDR and SNR performance of the four MIMO algorithms on the four datasets of the AC corpus.

When the MIMO WPE algorithm is applied to the AC corpus, where the no_echo dataset is used as the ground truth, there is little difference seen between Cauchy MIMO WPE and the original MIMO WPE algorithm. The Cauchy MIMO WPE show little improvement on the original and no_noise datasets in terms of the PESQ measurement (see Figure 5.26). For the no_echo dataset (see Figure 5.26c), there is an improvement in reverberation removal compared to the original no_echo files, showing that the degraded files are improved.

On the nonoise_echo dataset, the PESQ shows limited improvement which is similar to the original and no_noise datasets. However, on the nonoise_echo, there is an improvement regarding SNR, where Cauchy v1 outperforms the rest (see Figure 5.27d). In this case, the MIMO WPE algorithm is the worst performing. This pattern is also seen in the no_echo dataset (see Figure 5.27c), showing that this particular technique of removing the reverberation (by using a multiband compressor) has a negative influence on the performance of the MIMO WPE algorithm in terms of SNR. However, in terms of SDR there is no difference between the Cauchy v1 MIMO WPE and the MIMO WPE algorithms. Only Cauchy v2 and Cauchy v3 are performing worse here.

Evaluating Cauchy v3 MIMO WPE and the original MIMO WPE algorithms

Comparing the original MIMO algorithm and the worst performing modification (see Equation 5.15) and looking at the performance on individual sentences, the results show a bigger spread in sentences on which the original MIMO algorithm performs better (values greater than 0) than the other way round (see Figure 5.29). This information allows for a closer look at the outliers of each sentence to see whether there is a specific dialect or sentence type to which this performance difference can be attributed. A sentence is considered to be an outlier when the difference between the algorithms is bigger than 1.5 times the difference between the 25th and 75th quartile for that specific category.

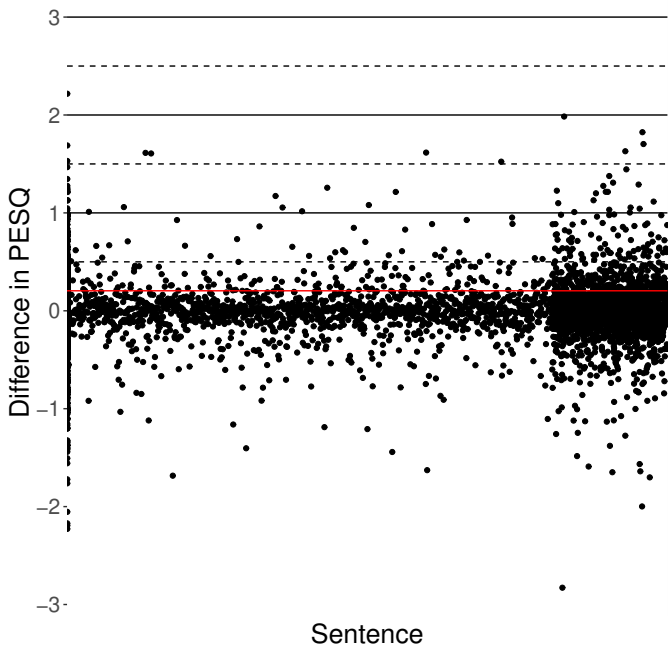


Figure 5.29: The difference in PESQ performance of the MIMO algorithm and the worst performing modification (Equation 5.15). These algorithms ran in room A. Each dot represents a spoken sentence ranging from SA1 on the left to SX on the right.

Within TIMIT there is not an even distribution of how often a sentence is spoken. For example, out of the 1718 sentences in the TIMIT training dataset, 1386 sentences are spoken once whereas there are two sentences that are spoken 462 times. This makes it difficult to say that a particular sentence has a high impact on the performance.

When looking at the occurrences of eight dialects (respectively New England, Northern, North Midland, South Midland, Southern, New York City, Western and Army Brat) or three sentences types (respectively dialect, diverse and compact), the dialect sentences (SA) are used to expose the dialect variations of the speakers, the phonetic diverse (SI) and compact (SX) are used for their coverage of the phonemes. The results show that the SX (compact) sentence types are spoken most (2309 times whereas SI, diverse, is 1386 times spoken and SA, dialect, is 924 time spoken). For the dialects, the numbers 1 (New England), 6 (New York City) and 8 (Army Brat) are are spoken least (380, 350 and 220 times respectively) whereas the others are spoken between 700 to 770 times.

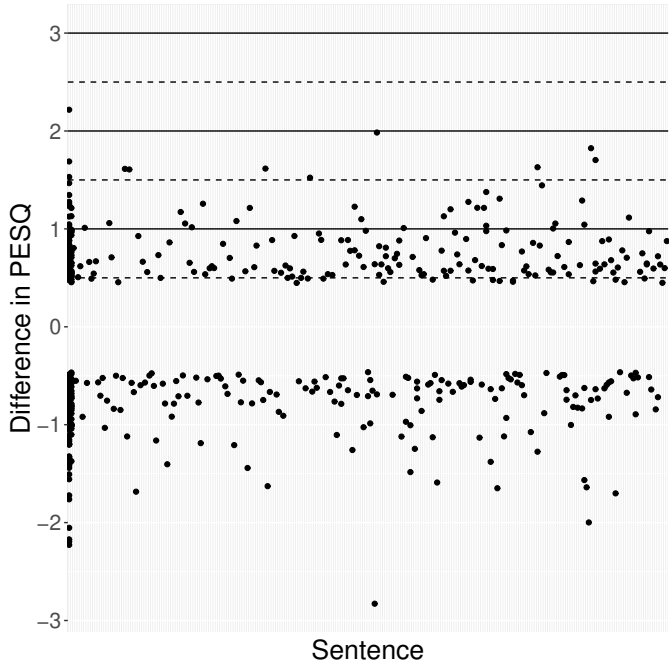


Figure 5.30: Outliers based on the sentence type. Each dot represents a spoken sentence ranging from SA1 on the left to SX on the right.

For the outliers of the three sentences types, the results between -0.5 and 0.5 in difference have been removed, leaving only the results outside of this area (see Figure 5.30). The results within the -0.5 to 0.5 area are not considered outliers as such because they are too close in similarity. In 7% of the outlier cases, the difference in sentence type SA was in favour of the modification where the original has less than 5% of the outliers. The other two types have a smaller difference (see Figure 5.31).

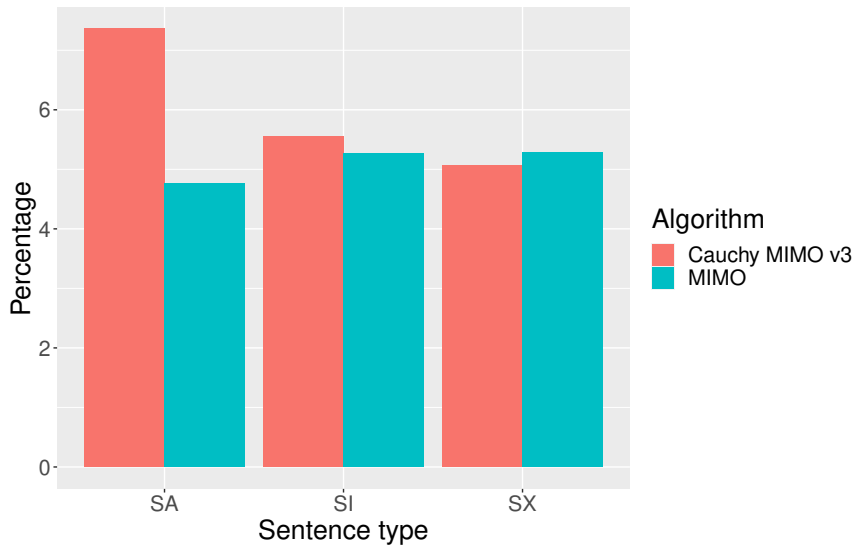


Figure 5.31: Percentage of outliers based on the sentence type.

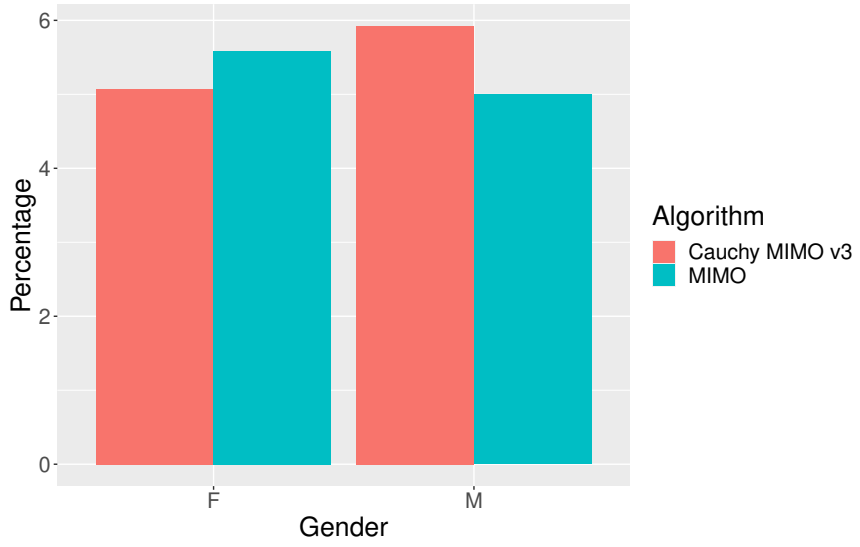


Figure 5.32: Percentage of female and male outliers based on the sentence type.

The gender of the speaker can have an influence on the performance of the algorithms when these are more susceptible to higher or lower frequencies. However, there is no clear difference in gender of the speakers when specifically looking at the gender difference in sentence types for the outliers (see Figure 5.32).

Longer sentences can make it easier for an algorithm to build a dereverberation mask. Also, the pronunciation of words or the word order can have an influence on the performance on an algorithm. When words are spoken in rapid succession with little pauses in between, then it becomes more difficult for the algorithm to determine where the reverberant starts. This influence can be present in certain sentences types but also be a distinguishing feature in certain dialects. When the dialects spoken in these sentence types are plotted, there is a clear performance gain for the MIMO algorithm in the 8th dialect. The opposite holds but with a smaller difference for the 3rd dialect. However, all these differences account for a very small percentage of the spoken sentences (see Figure 5.33).

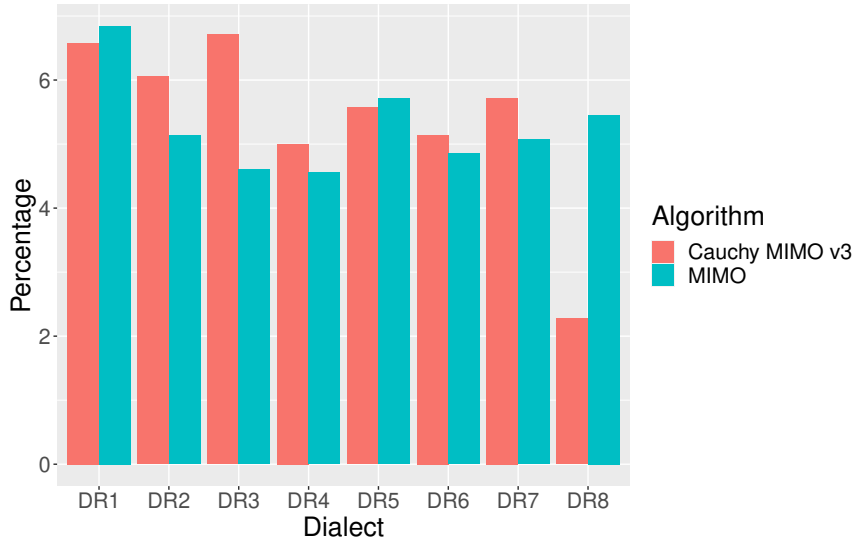


Figure 5.33: Percentage of different outlier dialects used based on the sentence type.

The same is done looking at the outliers of dialects (see Figure 5.34). However, there is a smaller difference between the different dialects, the gender of the speaker or the sentence type when the outliers are based on the dialects (see Figures 5.34, 5.35 and 5.36). This means it is not possible to say whether one dialect performs better or worse with a specific algorithm.

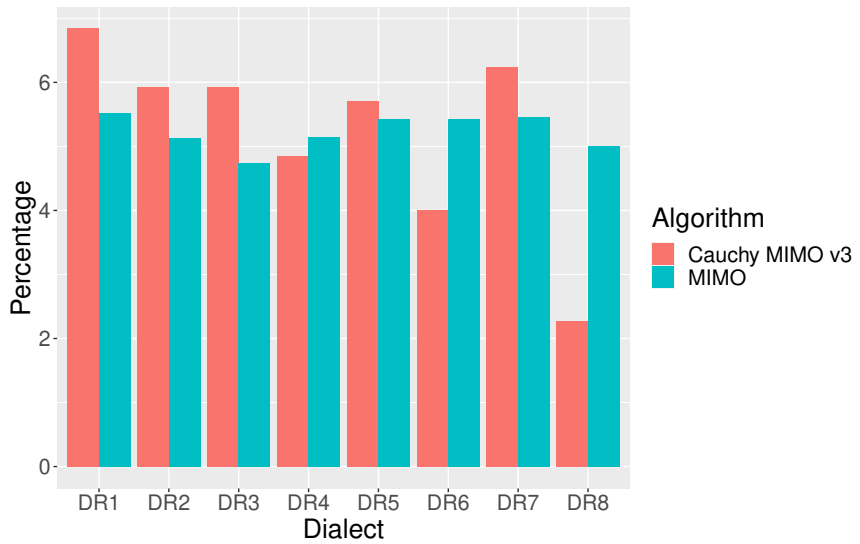


Figure 5.34: Percentage of dialect outliers.

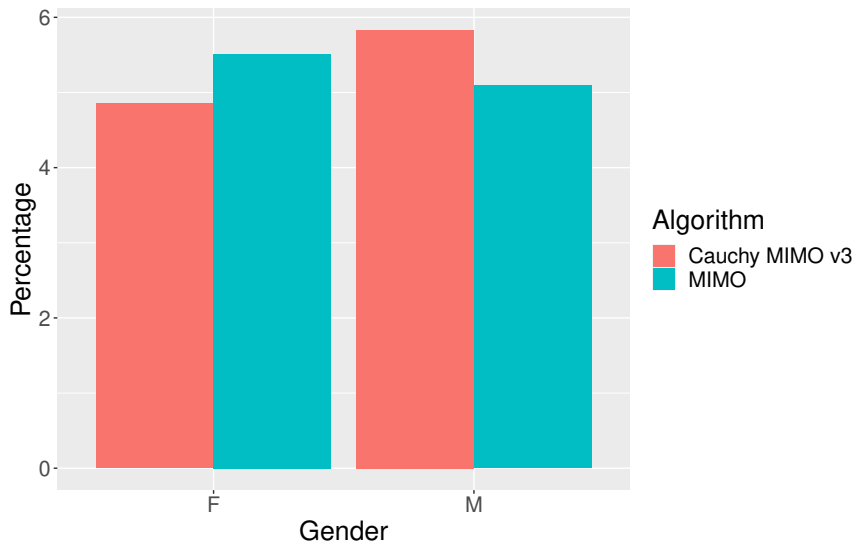


Figure 5.35: Percentage of female and male outliers based on the dialect.

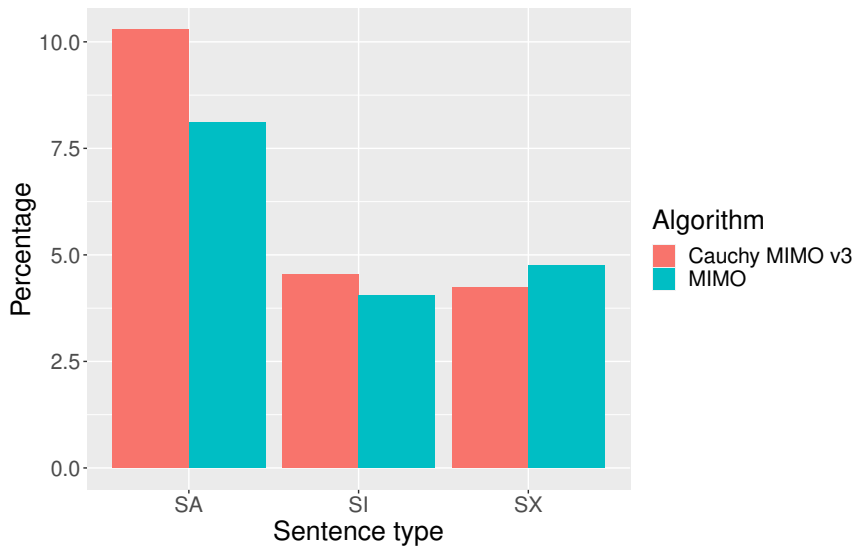


Figure 5.36: Percentage of different outlier sentence types based on the dialect.

5.6 Conclusion

The H_s correlation algorithm that has not been used for the speech dereverberation before has been presented. This correlation algorithm also formed the basis for the novel H_s NTF algorithm. Both have been compared with existing techniques. The H_s algorithm outperforms the H_1 and H_2 correlation on the PESQ, but only in the case of a 0.4s RT_{60}. However, in terms of SDR, the H_s algorithm show a better performance than the H_1 and H_2. This shows that even though the speech quality (as measured by PESQ) is not as good as that of H_1 and H_2, the algorithm introduces

fewer distortions (as measured by SDR) than the H_1 and H_2. This algorithm in its current form is no match for non-negative matrix factorisation. However, the basis of this algorithm can be used within a neural network or in addition to non-negative matrix factorisation.

When the H_s algorithm is applied in combination with NTF creating the H_s NTF algorithm, the difference between H_1 NTF and H_s NTF does not exist. However, both versions of the NTF algorithm do outperform their equivalent algorithms in the non-learning case. The H_1 learning correlation algorithm is used as part of the multichannel speaker separation algorithm (see Chapter 7).

Within this chapter, both WPE and MIMO WPE algorithms have been compared to modifications based on the Cauchy distribution. When looking at the WPE and MIMO WPE algorithms and the Cauchy WPE and MIMO WPE algorithms presented in Section 5.2.2, it is shown that the Cauchy distribution improves the performance of WPE algorithm in terms of PESQ but not of MIMO WPE. The latter still outperforms WPE, Cauchy WPE and Cauchy MIMO WPE on a simulated dataset. When MIMO WPE is applied to a realistic dataset with moving speakers, there is limited improvement made by the Cauchy MIMO WPE. In general, there is a small advantage in the execution time of Cauchy WPE but only for version 1 and 2. When these are compared to the loss in PESQ, the difference stays minimal. Within these results, there is no clear characteristic found that explains why the best performing Cauchy MIMO WPE algorithm performs worse than the original algorithm. The improvement on the original WPE algorithm is greater in terms of PESQ. However, the modifications still share the limitations of the original WPE algorithm.

N.B. there is no test for statistical significance applied to these results nor is this reported. The files are chosen on a random basis meaning that when the experiment is executed again the results will be similar if not the same.

Dereverberation makes it easier for a separation algorithm to determine which speaker is speaking. The process also helps with determining the location of the speakers that can then be used for the separation process (see Chapters 6 and 7).

Chapter 6

Single channel speaker separation

6.1 Introduction

For robots, it is important to be able to distinguish between speakers in order to execute the right command at the right time and not do something by accident that was overheard in a different conversation. Therefore, it is important to be able to distinguish between speakers and to separate these. There are different ways of doing this which can be defined into two classes:

- Single channel speaker separation.
- Multichannel speaker separation.

With single channel speaker separation only one microphone is used to separate the speakers, whereas with multichannel speaker separation multiple microphones are used. This chapter concentrates on single channel speaker separation. Given that the separation algorithms only use one microphone for the separation, the information they can use is limited. When the speech is degraded by artefacts from noise or reverberation, it becomes difficult for the separation algorithms to separate the speakers. By combining different techniques, the separation algorithms are able to adapt to these artefacts and distinguish between speakers.

This chapter concentrates on the separation of speakers using one microphone in a near- and far-field setting (see Section 2.6.1). For this problem two different styles of algorithms are used: supervised and unsupervised (see Section 2.2.2). In the case of supervised speaker separation the algorithm uses a ground truth recording from the speakers present in the mixture. Unsupervised speaker separation (also called

blind speaker separation) does not use a ground truth but instead tries to cluster similar frequencies together to build a mask for separating the different speakers.

For the supervised algorithms, this chapter concentrates on applying recurrent neural networks and convolution neural networks to this problem. These have the advantage that once trained they can be applied to unseen data with small adaptations in the environment. An often discussed downside of these algorithms is the large amounts of training data that are needed before the algorithm converges to a solution. However, when there is a lack of training data, a model can be shared and improved by different parties using so-called federated learning. Once trained, the model can be retrained to suit different environments or deal with more severe cases of noise and reverberation (this is in the form of transfer learning).

The unsupervised algorithm used here is non-negative matrix factorization (NMF). This algorithm needs to be trained on each mixture individually and therefore has a shorter training time. NMF has been applied successfully on near field speech and premixed audio files [7, 39, 186]. However, single channel NMF has difficulties with distinguishing between speakers at a longer distance or when noise and reverberation are present. The noise makes it difficult to distinguish between speakers because their voice quality is impaired. Whereas reverberation creates overlap between speakers which makes it difficult to tell where one speaker starts and the other finishes. On the other hand, when the deep learning algorithm is applied to a similar setting (i.e. same corpus) as it has been trained on, it can unmix the speakers in real-time. However, if the setting changes, the algorithm will have to be trained again on hours of data representing this new setting.

These two different styles of algorithms are compared against a baseline produced by an ideal binary filter (see Section 2.4.1). Three corpora (vocalization corpus, map task corpus and acoustic-camera corpus) are used in this thesis to evaluate different algorithms (see Table 6.1) for single channel speaker separation (the corpora are introduced in Chapters 3 and 4). These three corpora have different recording distances (i.e. distance between speaker and microphone) to test the performance of the sparse, convolution and direction of arrival NMF techniques under different conditions. This is to see how the algorithms perform when the recording contains noise and reverberation and to see how the distance between the microphones and

the speakers influences the performance of the algorithms.

Supervised Ideal binary filter
IBF
Supervised Deep Learning
biLSTM
CNN
DNN
LSTM
RCNN
Unsupervised Non-negative matrix factorisation
Sparse Euclidean NMF
Convolution Euclidean NMF
IS NMF
Sparse IS NMF
Convolution IS NMF
Sparse KL NMF
Convolution KL NMF
Direction of Arrival NMF
Direction of Arrival NTF

Table 6.1: Overview of the algorithms that are being evaluated

6.2 Algorithms

6.2.1 Non-negative matrix factorisation

Non-negative matrix factorisation (NMF) approximates the mixture by iteratively updating two matrices (see Section 2.4.2). When a subset of each of these matrices is selected and multiplied together, it gives an approximation of the speakers. To measure the difference between the mixture and the approximation produced by NMF, a cost function is used. NMF has been adapted with different cost functions each with their own characteristics (see Sections 2.2.4 and 2.4.2). Therefore, it is important to choose the cost function that works best with the specific problem.

This chapter concentrates on three different cost functions (described in more detail in Section 2.2.4):

- Kullback-Leibler
- Itakura-Saito
- squared Euclidean

These cost functions are chosen because of their adaptability and successful application to audio source separation. Kullback-Leibler divergence (KL), is the most popular cost function and is often used as a baseline to compare against others [141]. This cost function can easily be adapted to include directional information [39] or used to change NMF to a probabilistic algorithm [7].

The family of beta divergence, to which the KL divergence belongs, contains the two other cost functions (squared Euclidean distance and Itakura-Saito divergence) that are used in this chapter (see Section 2.2.4). This creates a special version of NMF called β-NMF, which has an additional parameter (β) in the update rules for the W and H matrices that determines which cost function is used. The values for β are given in Table 6.3 [141]. The squared Euclidean distance has the same adaptations as the KL divergence, for example sparsity or convolution. A comparison between the squared Euclidean distance and the KL divergence is often drawn to evaluate the performance of different adaptations [142]. The third cost function in the beta divergence is the Itakura-Saito divergence. This cost function is used less often for speaker separation and has not been extended in the same way as the Euclidean distance and KL divergence. All of these cost functions have been applied to different problems from speaker separation and singing voice separation to musical instrument separation.

The choice of cost function is one way to increase the accuracy of the NMF algorithm. Another way to adapt the algorithm is to use one of the following:

- convolution
- directionality (in the form of direction of arrival)
- sparsity

These allow the algorithm to make use of additional features in the data. Convolution (see Section 2.4.2) is useful for overlapping speech because it tries to combine an average of multiple timesteps. In the case of overlap, this means that the al-

gorithm is likely to find a point where there is only one speaker talking and stores this information in the feature matrix (W, see Section 2.4.2). Another advantage of convolution is that it should be able to deal with noise. This is because it averages a signal over multiple timesteps, thus removing small anomalies that can be attributed to noise.

Adding directionality (see Section 2.4.2) to a technique like NMF [7], provides intensity information about the different sources. This combines the knowledge of the possible source locations with information from multiple microphones allowing the algorithm to separate the sources. Direction of arrival NMF and direction of arrival non-negative tensor factorisation, as described by Stein [7], are not single channel techniques. For the estimation of the direction of arrival, there is more than one microphone needed in this case. This technique is used to compare the advantage a multichannel technique offers over a single channel technique. The location of the two sources is assumed to be differentiable, which on a 2D plane is not always the case when sources move around. For example, when the sources are directly behind each other this does not show up on a 2D plane, only in a 3D environment (see Section 2.6.3).

Sparsity (see Section 2.4.2) is useful when there is noise present or overlapping speech in the recordings because the algorithm is better at adapting for missing components. Sparsity ensures that the H matrix takes longer to converge to a solution because the H matrix is constantly being slightly modified. The modification creates W and H matrices that are more diverse and allow for a better separation of the sources than with vanilla NMF. Overlap within the speech signal creates an additional problem to solve for NMF, dividing the signals into their respective dictionaries. The level of sparsity is indicated by the λ parameter, which is added to the update function of the H matrix (Equation 2.42). N.B. sparsity is only enforced when $\lambda > 0$ [36, 37].

The three cost functions are combined with three additions (sparsity, convolution, direction of arrival). Each cost function will be combined with one of these additions and then evaluated on the corpora. These additions were chosen because of their performance on speech. Combined, they will allow for a comparison of the performance of the cost function and the performance of the additional functions.

Due to the implementation of the DoA it is not possible to apply this technique to the other cost functions (see Section 2.4.2). In the case of DoA NMF, the algorithm is changed so that it will output the percentage of the frame used by each speaker instead of building a mask using the power spectral density. In total, nine different NMF techniques were applied to the speaker separation problem (see Table 6.3). Four of these techniques use the Kullback-Leilbler (KL) divergence, while the others use the Itakura-Saito (IS) divergence or the squared Euclidean distance.

6.2.2 Deep Learning

NMF is not the only technique applied to speaker separation - one alternative is the field of deep learning where recurrent neural networks (RNN) and convolution neural networks (CNN) have been successfully applied to supervised speaker separation [187–189] (see Section 3.3.2). One of the advantages of deep learning is that it can learn the unmixing mask and then be applied to unseen data without retraining the algorithm. The main disadvantage is that it needs multiple hours of speech data and, equally, multiple hours of training to build an unmixing mask for the separation of speakers. These are often the main arguments against deep learning. On the other hand, the vast volume of training data also means that deep learning can be adapted to remove noise and reverberation from the mixtures when this is present, assuming that the ground truth files are free of these artefacts. Furthermore, a generalised algorithm is robust to small changes in the environment and does not need to be retrained on each file.

Both RNN and CNN are often applied to this problem in a supervised manner where there are ground truth files of the speakers available. For these techniques it is very important to choose the right parameters and size of the network. Five different deep learning techniques were applied to the speaker separation problem; bi-directional long short-term memory networks (biLSTM), convolution neural networks (CNN), deep neural networks (DNN), long short-term memory networks (LSTM) and recurrent convolution neural networks (RCNN). The LSTM and biLSTM networks have the advantage of looking a number of timesteps back or forward, which allows the network to find a segment where there is only one speaker speaking. This advantage is also used by the RCNN which uses convolutions to learn the

spectrum and a recurrent layer to find similarities between the different timesteps. These similarities correspond to the different speakers and allow the network to build a filter.

6.3 Corpora

For testing the different techniques, two of the 16 corpora described in Section 3.1 as well as the acoustic camera corpus (see Chapter 4) are used. The corpora vary in recording length, the presence of noise and recording distance. The vocalization corpus contains telephone speech, which is close to the mouth, whereas the MapTask corpus contains speech recorded with close-talk microphones. On the other hand, the acoustic camera corpus contains far-field speech with a large distance between microphone and speaker (see Section 2.6.1). Both the vocalization and MapTask corpora are recorded in a noise free environment but contain speech of a background speaker. On the other hand, the acoustic camera corpus is recorded in a realistic office environment and contains background noise but there is no background speaker present. In the case of the vocalization and MapTask corpora, the recordings do not contain reverberation whereas due to the size of the room and the room being unfurnished, the acoustic camera corpus does contain reverberation. The features of these corpora mean that the algorithms need to deal with:

- overlapping speech which has the same or is lower in volume than the main speaker volume
- noise coming from appliances (i.e. air-conditioning units and computers)
- reverberation due to the size of the room.

A concise overview of the three corpora is given in Table 6.2.

The first corpus is the vocalization corpus[1] [87] which contains recorded telephone conversations of 120 different subjects. In the recording there is background speech present of a second speaker. This does not provide a clean ground truth. For this corpus there is no localisation information available.

The MapTask corpus [88] is the second corpus. In this corpus people are wearing headphones and a microphone and explained how to get from A to B on a map. This

[1] http://www.dcs.gla.ac.uk/vincia/?p=378

corpus	vocalization	MapTask	acoustic-camera
subjects	120 (63 women, 57 men)	64 (32 women, 32 men)	16 (12 men, 4 women)
# mics	1 (per file)	1 (per file)	72 (per file)
# files	2763	191	7
file length	0:10	5:00	1:30
2nd speaker	Yes	Yes	No
noise	No	No	Yes
mic-to-source	$<$ 1 metre	$<$ 1 metre	$>$ 6 metres
localisation	No	No	Yes
rec. env.	Lab setting	Lab setting	Workshop
F_s	16kHz	16kHz	192kHz
transcripts	No	No	Yes

Table 6.2: A comparison of the three corpora used for speaker separation

corpus contains speech of 64 subjects. As with the vocalization corpus, the second speaker can be heard in the background because the people were recorded in pairs. This means that this does not provide a clean ground truth nor does this corpus have localisation information available.

The third and last corpus is a small corpus recorded with the Acoustic Camera (AC) (see Table 6.2 and Chapter 4). The room used for these recordings has background noise along with reverberation due to the room size, as is typical of many home and office environments. The high sensitivity of the microphones to noise and echo means that post processing is needed to create a clear approximation of the speaker.

6.4 Experimental setup

6.4.1 Corpora

The vocalization corpus and MapTask corpus were used to determine how well each technique performs on the separation task. In using these two corpora the mixtures of the recordings contain overlapping speech coming from a background speaker. This increases the difficulty of doing a clean separation.

With the AC corpus (see Chapter 4), the performance of the different techniques when there is noise and reverberation in the recording was measured. This means

that the algorithms have to be able to separate the speakers in a natural environment which increases the difficulty of the problem. The recordings of the AC corpus are downsampled to the same sampling frequency as those of the vocalization and MapTask corpora.

6.4.2 Environment

All of the NMF algorithms (except for DoA NMF and DoA NTF [7] algorithms) are implemented in NMFlib a library for MATLAB[2]. For the DoA NMF and DoA NTF the implementation of the author is used. These algorithms run on MATLAB 2013b on a 2013 MacPro with Intel Xeon E5 3.7 GHz and 16 GB of RAM. The deep learning algorithms are implemented in Keras and TensorFlowv1 and are published on GitHub[3]. The deep learning algorithms run on the NVIDIA DGX-1. The results are measured using the BSS_eval library [190] for MATLAB and the mir_eval library for Python [184].

6.4.3 Parameters

As explained in Section 2.4.2, the output of the Short-time Fourier transform (STFT, see Section 2.2.3) was used as input for the NMF algorithms with a windows size of 30 ms and an overlap of 10 ms. These algorithms have two fixed parameters (F and K), while parameter N depends on the length of the file. For F, 513 frequency bins were used, determined in the same way as in Section 5.4.3. For the NMF algorithms K is set to be the desired number of speakers, in this case 2. The algorithms were stopped after 1000 iterations, by which time the cost function has converged.

The sparsity parameter setting was selected independently for each algorithm and each corpus independently. The sparsity parameter started at 0.001 and was increased with increments of 0.001 until the convergence of the algorithm. When the sparsity parameter reached 0.9 the experiment was stopped and the best parameter, corresponding to the highest signal-to-artifact, signal-to-distortion and signal-to-interference ratios, was chosen (in this case a sparsity parameter of 0.001). The results showed that the sparsity parameter was robust over different corpora also

[2] `https://github.com/audiofilter/nmflib`
[3] `https://github.com/TeunKrikke/SourceSepDL`

when noise and reverberation were introduced.

Technique	Cost function	Parameters	
		λ	β
Sparse Euclidean	Euclidean	0.0001	2
Convolution Euclidean	Euclidean	0	2
IS	Itakura-Saito	0	0
Sparse IS	Itakura-Saito	0.0001	0
Convolution IS	Itakura-Saito	0	0
Sparse KL	Kullback-Leibler	0.0001	1
Convolution KL	Kullback-Leibler	0	1
DoA NMF	Kullback-Leibler	0	1
DoA NTF	Kullback-Leibler	0	1

Table 6.3: Overview of the parameters and cost functions used by the evaluated NMF techniques.

To apply DoA NMF and DoA NTF to the vocalization and MapTask corpora, localization information is created artificially because this is not provided by the corpus. To do this, a time delay of one audio frame is used. This means that the artificially created microphones are spaced at a relative distance of 1 audio frame apart, dependent on the frame rate of the recording. For example, when a recording is made at 16 kHz the microphones would be spaced at $\frac{c}{f_s}$ metres or in this case $\frac{340.29}{16000}$ metres which is equal to 0.021 metres.

IBF has been applied with a varying volume for the primary speaker in the range of -10 dB to +10 dB with increments of 5 dB. This means that the two speakers are mixed with different volume levels. For example, -10 dB means that the primary speaker is 10 dB lower in volume than the secondary speaker. On the other hand, +10 dB means that the primary speaker is 10 dB higher in volume than the secondary speaker.

The five deep learning algorithms (see Section 2.4.3) contain a separation layer which contains an additional hidden layer with 512 units and a Wiener filter that is used for the separation. The loss function which optimises the network is the mean squared error between the output of the network, which is two signals, and the ground truth of the sources. Apart from this configuration each algorithm has a specific configuration:

- recurrent neural network (RNN) with two long-short term memory (LSTM) nodes with 512 units [189]

- recurrent neural network (RNN) with two directional long-short term memory (LSTM) nodes with 512 units [189]
- convolution neural network (CNN) with two convolution layers with 64 filters of 3 x 3 [154]
- deep neural network (DNN) with two hidden layers [189] each with 150 units
- recurrent convolution network (RCNN) with two convolution layers (64 filters of 3 x 3) and one recurrent layer (512 units)

6.4.4 Performance measurements

For testing the different techniques, three objective measurements were introduced in [1] (see Section 2.5) namely: signal-to-distortion ratio (SDR); signal-to-interference ratio (SIR) and signal-to-artefact ratio (SAR). Positive values indicate better performance for all measurements.

6.4.5 Experiments

The IBF and nine different NMF algorithms (see Section 6.2.1) were run on the three different corpora (see Section 6.3) to compare their performance in different situations (telephone to workshop environment) and with different file lengths (10s to 5 minutes). The IBF algorithm does not have additional parameters that can be set, only the volume of the primary speaker is adjusted.

Furthermore, the four different deep learning networks have been applied to the vocalization corpus. This corpus contains the most recordings out of the three corpora and is therefore the most suited to train a neural network on.

6.5 Results

The different algorithms are all run on the vocalization corpus with the performance of the algorithms presented as measures of signal-to-distortion ratio (SDR); signal-to-interference ratio (SIR) and signal-to-artefact ratio (SAR). For these measurements, it holds that the higher value is the best performing algorithm. The NMF variants and IBF have also been run on the MapTask and AC corpora.

6.5.1 Vocalization corpus

Ideal binary filter

When the IBF is applied to the vocalization corpus (see Figure 6.1), the results show that when the primary speaker is higher in volume than the secondary speaker (i.e. > 0dB on the algorithm axis) the IBF performs better. This is seen in the SAR and SIR measurements but the biggest improvement is seen in the SDR.

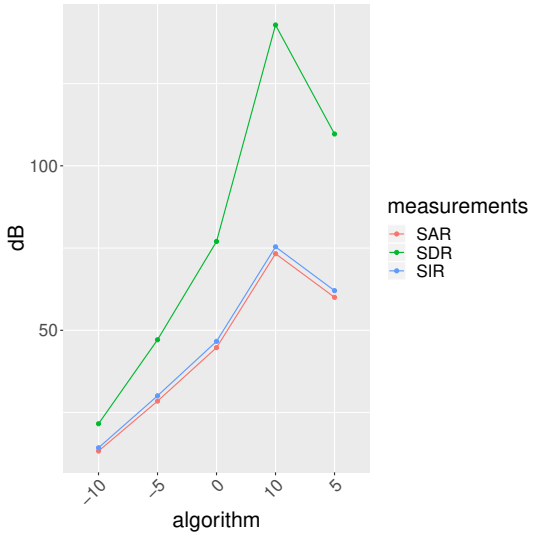


Figure 6.1: A comparison of IBF on the vocalization corpus.

Deep Learning

Looking at the deep learning techniques (see Figure 6.2), the normal LSTM (thus the recurrent network with two LSTM nodes) outperforms the other three techniques. Changing this particular network into a bidirectional network where it has information from the past and the future performs worse than the normal LSTM but similar the other techniques. The DNN has the lowest SDR results of the techniques meaning that it has issues with removing the removing distortion and introduces new distortion to the separated speech. Both CNN based techniques are able to remove the artefacts in the separation whereas the LSTM seems to introduce new artefacts.

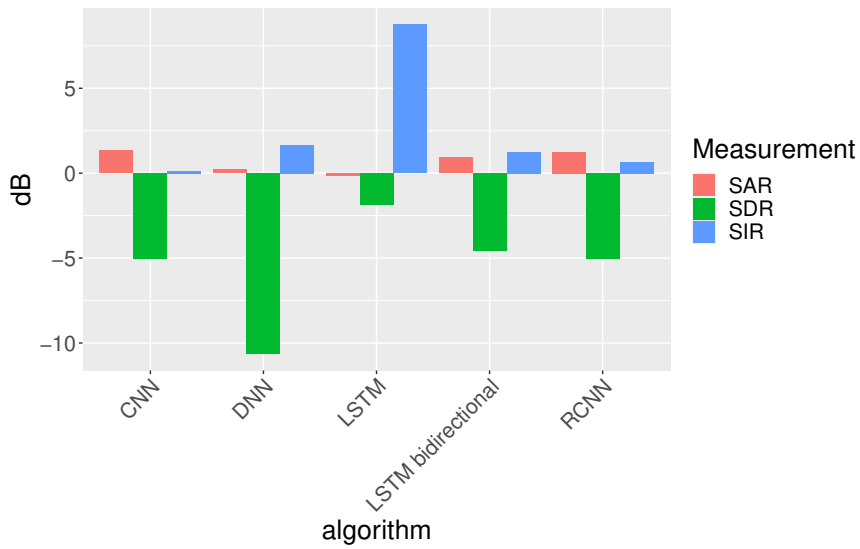


Figure 6.2: A comparison of deep learning algorithms on the vocalization corpus.

Non-negative matrix factorisation

The NMF results (see Figure 6.3) show that having a multiple microphone solution improves the ability to separate speakers. However this is in a completely clean environment where there is not interference except from a second speaker. The normal NMF techniques are able to remove the artefacts but struggle to reduce the distortion. On the vocalization corpus, the IS based techniques are all very similar in terms of removing artefacts and interference, there is a minor difference between vanilla IS and the convolution and sparse techniques in favour of the latter two. A similar pattern is seen between the convolutive and sparse algorithms for both the squared Euclidean and Kullback-Leibler cost functions. However the pattern is reversed in both cost functions. For the squared Euclidean cost function, the convolution algorithm outperforms the sparse whereas for the KL cost function, the sparse outperforms the convolution. This is most noticeable for the distortion and less so for the other two measurements.

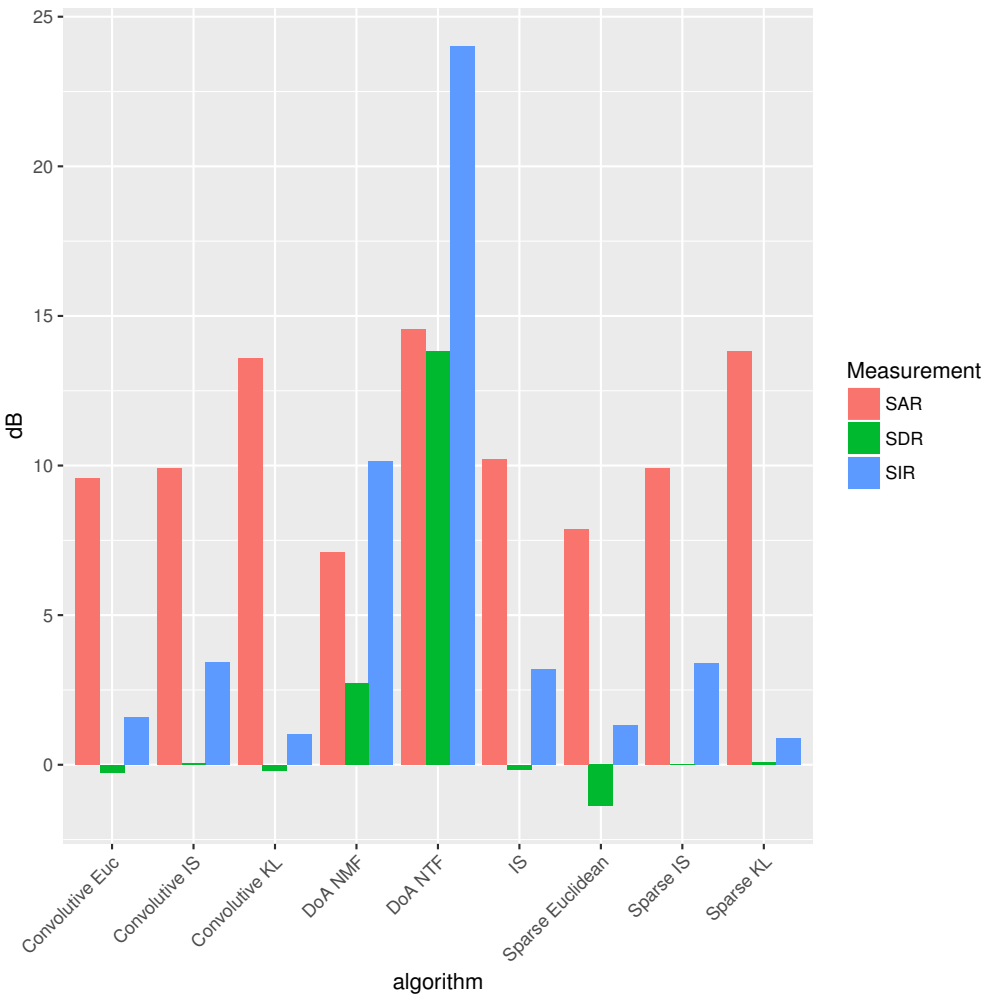


Figure 6.3: A comparison between different NTF and NMF techniques on the vocalization corpus.

6.5.2 MapTask corpus

Ideal Binary Filter

For the MapTask corpus when the second speaker is lower in volume (i.e. a positive number on the algorithm scale), the IBF performs better (see Figure 6.4). However, the difference between the SAR and the SIR measurements is very small and the improvement overall is lower than the SDR. This means that there is a relationship between the volume of the second speaker and the amount of distortion present in the separation. The result of the IBF on the MapTask corpus is very similar to that on the vocalization corpus. This is due to the similarity in data. Both corpora contain single near field speech with a secondary background speaker.

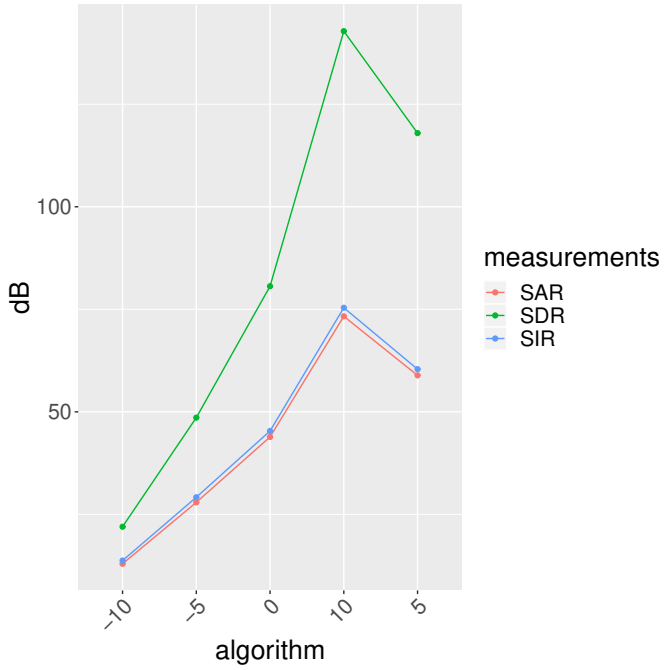


Figure 6.4: A comparison of IBF on the MapTask corpus.

Non-negative matrix factorisation

The NMF results of the MapTask corpus (see Figure 6.5) show that a multi microphone solution works best, especially when looking at the interference results. For the single channel results, the sparse Itakura-Saito performs similarly to the sparse squared Euclidean technique but is better at removing artefacts from the results. There is also a high similarity between the convolution Kullback-Leibler and the vanilla Itakura-Saito techniques, both have similar performances across the three measurements. This shows that the modifications do not always improve performance. Another similarity is between the distortion results of the convolutive squared Euclidean and the sparse Kullback-Leibler techniques. Both techniques are unable to remove the distortion from the separated speech. When comparing the multichannel DoA NMF technique to the single channel techniques the results show a similar performance on the removal of artefacts as the sparse Kullback-Leibler technique does. In terms of distortion, DoA NMF performs better than the vanilla IS technique but worse than the sparse modification of the same cost function.

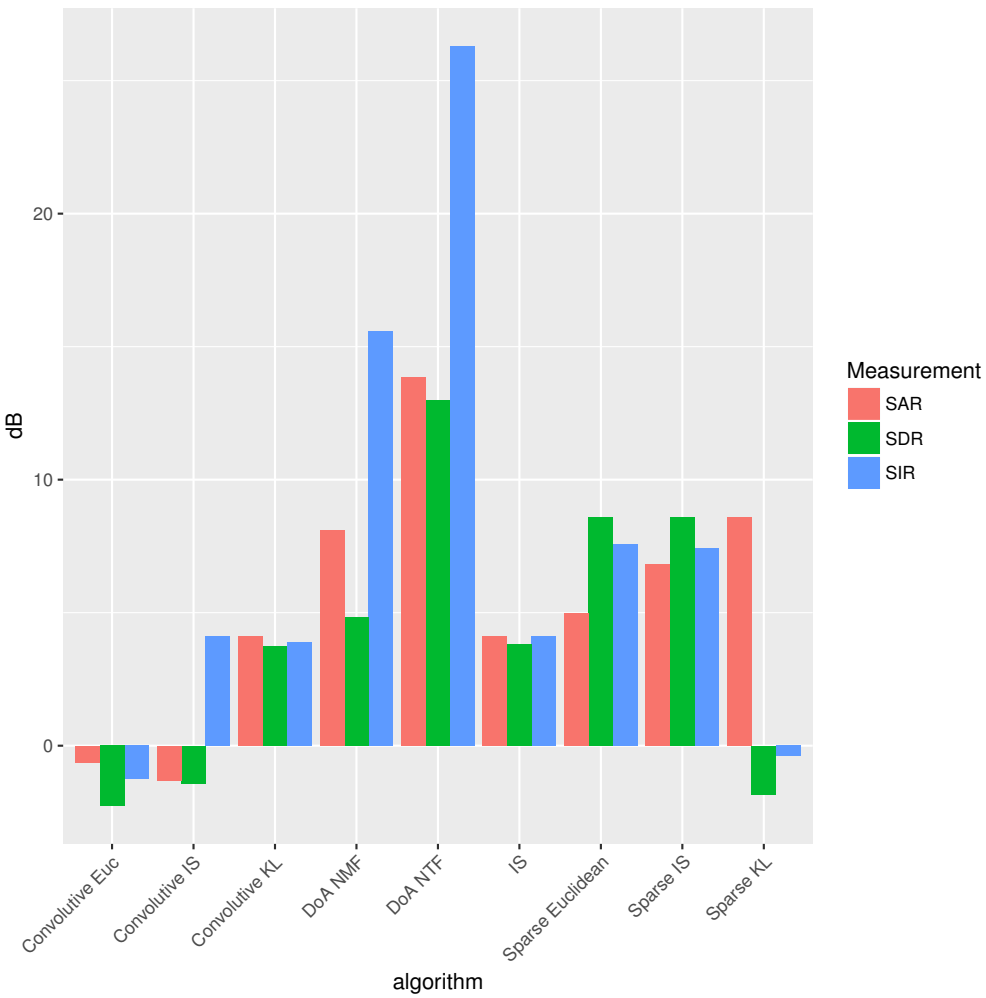


Figure 6.5: A comparison between different NTF and NMF techniques on the Map-Task corpus.

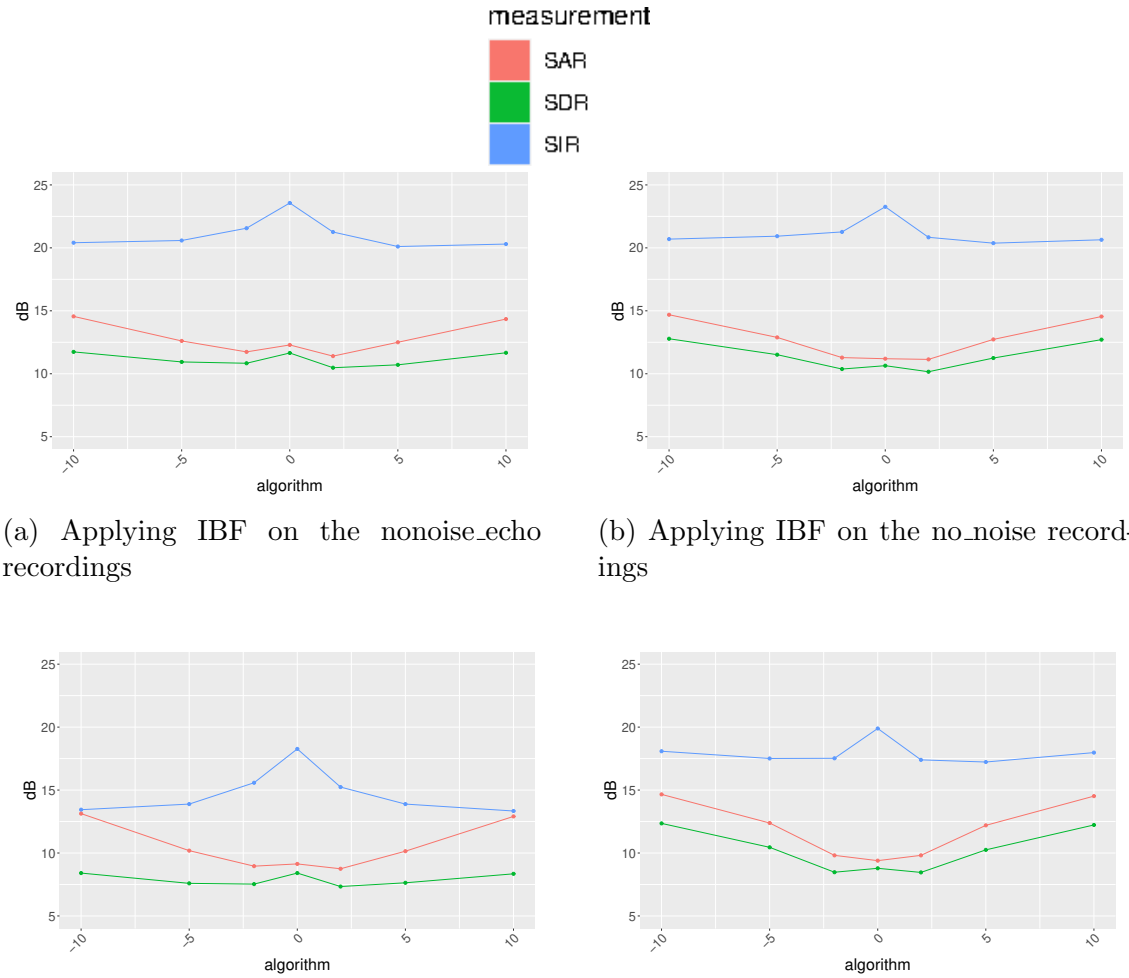
6.5.3 Acoustic Camera corpus

Ideal Binary Filter

Looking at the IBF results for the AC corpus, it shows that the filter has more difficulty separating the speakers when they have the same volume (0 dB on the algorithm scale) than when one of the speakers has a higher volume ($<$ 0db for the second speaker having a higher volume and $>$ 0 db for the first speaker). High distortion removal can be seen at both ends of the scale with the lowest point around 0 dB when both signals are equal (see Figure 6.6). An opposite pattern is seen for the interference.

For the no_echo dataset the peak interference is higher as well as the valley of the distortions being deeper (see Figure 6.6c). This shows that without the

reverberation, the IBF is better at removing the interference when both speakers have the same volume but has more difficulties controlling the distortions.



(a) Applying IBF on the nonoise_echo recordings

(b) Applying IBF on the no_noise recordings

(c) Applying IBF on the no_echo recordings

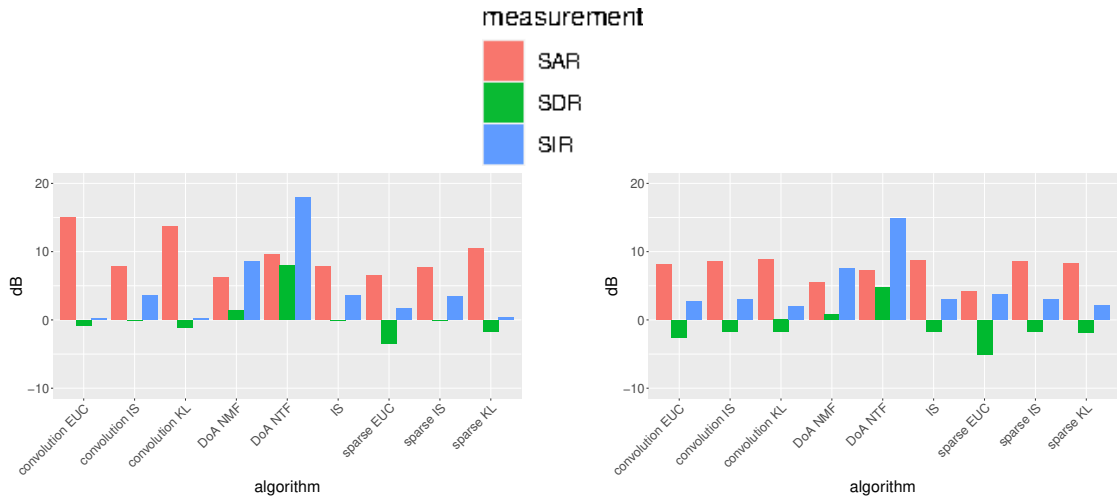
(d) Applying IBF on the original recordings

Figure 6.6: A comparison between of IBF on the nonoise_echo, no_echo, no_noise and original AC corpus recordings.

Non-negative matrix factorisation

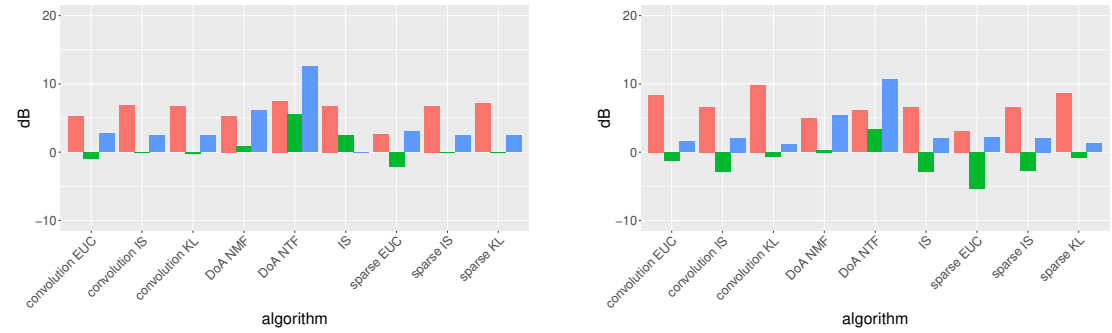
Looking at the performance of NMF on the AC corpus, the results show that DoA NTF outperforms the other techniques in all subsets of the corpus (see Figure 6.7). However, the technique had limited gain when it was run on the original and the no_noise recordings. This also applies to DoA NMF. On the no_noise dataset, the IS technique outperforms the other technique (see Figure 6.7b). The same can be seen on the no_echo dataset (see Figure 6.7c) but not on the nonoise_echo or on the original datasets (see Figures 6.7a and 6.7d). These latter two have mixed results where there is no one cost function better than another. On both the no_echo and

the no_noise datasets the convolution adaptation of NMF outperforms the vanilla and sparse adaptations. This shows that when one feature of the corpus is removed, convolution still best describes the mixing of the speakers. However, this is not the case when both are present or both are removed. In these latter two cases, both adaptations (sparse and convolution) have similar performance with the Kullback-Leibler cost function being the best performing for both adaptations.



(a) Applying NMF on the nonoise_echo recordings

(b) Applying NMF on the no_noise recordings



(c) Applying NMF on the no_echo recordings

(d) Applying NMF on the original recordings

Figure 6.7: A comparison between different NMF techniques on the nonoise_echo, no_echo, no_noise and original AC corpus recordings.

6.6 Conclusion

The NMF results show that all techniques have a good SAR on the vocalization corpus and the MapTask corpus, meaning not many artefacts are introduced (see Figures 6.3 and 6.5). The SDR and SIR values are poor for all techniques except

for NTF. This shows that NTF is able to remove both speakers from a single file more clearly than the other techniques. However, the combined speech files have no noise or reverb. Which in the real world is rarely the case, with the exception of telephone conversations.

The results of the NMF techniques applied to the vocalization and MapTask corpora are worse than the baseline provided by IBF. This is because the lack of noise makes it easier for the IBF algorithm to do a separation of the speakers. On the other hand, when looking at the results from the AC corpus, the baseline is lower. In the case of the no_echo dataset, the DoA NTF algorithm is removing a similar amount of artefacts as the IBF does (see Figures 6.6c and 6.7c).

The different NMF techniques were applied to compare the performance with and without noise and reverb on the acoustic camera corpus (see Figure 6.7). The IS cost function performs the best when noise is removed (see Figure 6.7b). The same applies when only the reverb (see Figure 6.7c) was removed and when the results were compared to the original (non post-processed) files (see Figure 6.7d). On removing both noise and reverb (see Figure 6.7a), the sound gets distorted to an extent that the KL and squared Euclidean versions of NMF outperform the IS and DoA versions. This happens because the technique used to remove the reverberation is a multi-bandpass filter which suppresses certain frequencies in the recordings, thereby introducing distortions.

Comparing the results of all techniques on the different versions of the acoustic camera corpus against to original dataset, when the reverb was removed, all techniques show an improvement in the SAR and SDR values but get lower SIR values. The squared Euclidean cost function has the greatest improvement compared to the rest of the techniques. However, over all three corpora, the squared Euclidean cost function has the lowest SAR of all the techniques and is therefore the worst performing technique on this corpora. Both the sparse and convolution techniques work better on the noisy version of the acoustic camera corpus. With this version, more positive values were seen. However, all the algorithms are outperformed by the IS cost function without the use of sparsity or convolution on the reverberant speech.

When specifically comparing the results of the AC corpus when both noise and

reverberation (see Figure 6.7a) have been removed with the results of the vocalization and MapTask corpora, the performance on the AC corpus is lower than of the other two corpora. However, there is a similarity between the AC corpus and the vocalization corpus, making it difficult to be conclusive about the influence of the distance between speaker and microphone.

The deep learning results are difficult to compare with the NMF results because they are different techniques. The deep learning algorithms are all used as supervised algorithms where there is a ground truth which the algorithm uses to adapt the weights to. On the other hand, NMF is an unsupervised technique using only the mixture (which is the input) to adapt towards. For deep learning, a LSTM shows the best performance on the reduction of distortion and interference. The LSTM network and sparse Euclidean NMF algorithms are comparable in terms of signal-to-distortion and signal-to-interference ratios. When the LSTM network is changed to bidirectional, then there is an improvement on the reduction of artefacts but has a negative influence on the other two measurements. However, because the vocalization corpus does not provide enough training data, the algorithm does not perform as well as the NMF algorithms. This lack of training data can be resolved by applying federated learning where the model is shared among different entities. These try to solve the same problem and have similar data (i.e. a single channel mixture containing two speakers without noise or reverberation). Similarly the three corpora can be combined without mixing the speakers between corpora. This gives the model more training data thus improving the overall result. Another usage for corpora of this size is to apply transfer learning where a model has been trained on a similar (sometimes broader) problem and is changed to learn the specifics. In this case a model used for single channel speaker separation can be retrained using the AC corpus for separating speakers in noisy and reverberant environments using far field speech.

Using multiple microphones and information as direction of arrival increases the performance of the NMF algorithm in a clean environment which can be seen by the results of the DoA NMF and NTF algorithms. However, algorithms which use multiple microphones are designed to work in more complex environments as described in Chapter 7.

Chapter 7

Multichannel channel speaker separation

7.1 Introduction

We, humans, are good at determining where sound is coming from and concentrate on particular sounds/voices that we are interested in. This is possible because we have two ears that act as a stereo microphone array. These allow us to remove noise from the things we are listening to and determine where the sound is coming from. When robots make use of microphone arrays they can determine, by calculating the direction of arrival or the time difference of arrival, where the sound is coming from. This information helps with noise and reverberation reduction but also with speaker separation.

For speaker separation, multi-channel NMF and non-negative tensor factorisation (NTF) are algorithms that use multiple microphones in an array. This can be a stereo array where two microphones are placed in the room with some space between them (in height, width or depth) or a multi microphone array where microphones are placed in a specific configuration (see Section 2.6.2). In the case of robotics, the microphones are typically placed on the robot in a forward facing configuration, i.e. listening where the robot is looking. This allows for a direct mapping between sound and vision but has problems with recognising when the sound is coming from behind. Having additional location information coming from multiple microphones allows the robot to distinguish between speakers in more complex environments.

Multiple microphones are able to reduce reverberation as shown in Chapter 5.

For source separation, having multiple microphone gives the algorithm the ability to distinguish between speakers in a 3D environment. One way to describe the location is by determining a spatial covariance matrix (see Sections 2.4.2 and 2.4.2), another is using the direction of arrival (see Section 2.4.2). This tells the algorithm where the greatest overlap is between microphones, this corresponds with the location of a speaker. This chapter concentrates on two different ways of determining the location of the different sources by using:

- a spatial covariance matrix (time-difference of arrival NTF and Covariance NTF).
- the direction of arrival (direction of arrival NMF and direction of arrival NTF).

This chapter concentrates on four different algorithms, two of which use direction of arrival (DoA NMF and DoA NTF), the other two use spatial covariance matrix (TDoA NTF and Covariance NTF). This matrix can be calculated using the covariance between the different microphone or using the time-difference of arrival (TDoA). Both the TDoA and DoA are beamforming techniques to calculate the location of the sound source (see Section 2.6.3). Covariance NTF uses the spatial covariance matrix without calculating the TDoA first. In this case the algorithm calculates the correlation between the different microphones to determine where the sound is coming from, which is similar to what dereverberation algorithms do (see Chapter 5). N.B. the DoA NMF and DoA NTF are the same algorithms as used for single channel speaker separation in Chapter 6.

Three corpora (vocalization corpus, map task corpus and acoustic-camera corpus) are used to evaluate four different algorithms (Covariance NTF, DoA NMF, DoA NTF and TDoA NTF) for multichannel speaker separation (the corpora are introduced in Chapters 3 and 4). Two of these corpora (vocalization and MapTask corpora) have reverberation added from a simulated environment with the same size as the room used for the AC corpus recordings. Whereas the recordings in the AC corpus contain noise and reverberation from the recording environment. This is to see how the algorithms perform when the recording contains only reverberation and how the reverberant and noise influences the performance of the algorithms.

7.2 Algorithms

The algorithms used in this chapter are described as non-negative tensor factorisation (NTF) algorithms, which is an extension of non-negative matrix factorisation. Instead of working with two matrices to try to approximate the multi-channel mixture, the NTF algorithms work with 3 or more matrices to approximate the mixture. N.B. the multi-channel mixture is considered to be a tensor because it has more than 2 dimensions.

7.2.1 DoA NTF

Direction of arrival (DoA) NMF (see Section 2.4.2) changes the updates rules for the W matrix. NMF multiplies the W matrix with the DoA matrix.

In comparison to the NMF version, DoA NTF factorises 3 matrices instead of 2. The third matrix is made up of the information from the DoA [7] containing the direction of the sound. This changes NMF into non-negative tensor factorisation (NTF). The additional information improves performance because it builds a mask taking into account the sound that is coming from each individual microphone. This has an advantage over NMF where the information is present in one of the two matrices (W or H) as an extra dimension, now it is seen as a separate multiplication over the whole mixture. Extra complexity and accuracy are therefore added to the algorithm.

An important change Stein [7] introduces is the change to probabilistic NMF where the mask is changed to a maximum-likelihood mask. As input the probabilities of the mixture is used instead of the power density spectrum. This keeps the update rules the same but changes the cost function to maximising the cross-entropy of the prediction and original mixture.

7.2.2 TDoA NTF

The time-difference of arrival NTF [39] (see Section 2.4.2) is one of the two SCM based algorithms used in this chapter. This algorithm uses look directions for each individual microphone instead of using the direction of arrival. For the look directions, the algorithm uses the phase difference per frequency for those look directions.

This information gives an approximate location of the speaker and can be used to approximate the mixture. Combined with a directional weight (Q) this can also be used to cluster the NMF components and separate the speakers. Using look directions allows the algorithm to describe the dominant frequencies per look direction and a field of view showing the location of the source.

Another difference is that this algorithm uses the Hermitian transpose of the input instead of the probabilities (as used in Section 7.2.1).

7.2.3 Covariance NTF

The covariance NTF [40] is another SCM based method and calculates the SCM by using the covariance between microphones (see Section 2.4.2). This method is similar to the WPE method discussed in Section 2.3.3. It accounts for the reverberation of the environment making the resulting signal free from reverberation. This should make it easier for the speech recogniser to determine when a speaker finishes and what the speaker has said.

The main difference between this algorithm, the DoA NTF and the TDoA NTF, is the usage of the spectral correlation matrix which is dependent on the spatial covariance matrix. The spectral correlation matrix shows the dominant (loudest) features in the spectrum which correspond with the different speakers. The information of the spectral correlation matrix is used in calculating the spatial covariance matrix together with the H_1 cross-correlation between the original mixture and the approximation. N. B. this H_1 cross-correlation takes in to account the noise that is present in the relative error and shows the strongly correlated components between the two which can be used to determine the spatial covariance matrix.

Another difference between the DoA NTF, the TDoA NTF and this algorithm is the separate calculation of the noise. This allows the algorithm to increase the similarity between the approximated mixture and the original mixture by assuming that the missing information can be modelled by the noise.

7.3 Corpora

Three corpora (vocalization, MapTask and acoustic-camera corpora) are used as input for the different algorithms which are the same corpora used for evaluating single channel speaker separation (these corpora are introduced in Chapters 3 and 4). Two of these corpora (vocalization and MapTask corpora) use a simulated environment because they are recorded with head-mounted microphones or with telephones. However, the AC corpus is recorded in realistic office and workshop environments using a microphone array. The simulated environment used for the vocalization and MapTask corpora is similar in size as the workshop environment used for the AC corpus.

7.4 Experimental setup

7.4.1 Corpora

The performance of the two DoA techniques (DoA NMF and DoA NTF) and the two SCM based techniques (Cov NTF and TDoA NTF) are compared by running them on three different corpora. These corpora are the same corpora as used for the single channel, namely the vocalization corpus, MapTask corpus and Acoustic Camera corpus (see Section 6.3).

7.4.2 Environment

The recordings of the first two corpora (the vocalization and MapTask corpora) are played in a simulated environment (called Room A, see Section 5.4.2) where there is only reverberation, and no noise, present. To simulate this room, the same library is used as for the dereverberation experiments (see Section 5.4). However, in this case only room A is being used because this room is similar in size as the environment in which the AC corpus is recorded. For the DoA NMF and DoA NTF the implementation of the author is used, the Covariance and TDoA NTF algorithms are based on implementations by Nikunen et al. [39] and Ozerov et al. [38, 40]. These algorithms implemented in Python, run on a OpenSuse Linux

computer with an Intel i3 processor with 4GB RAM and published on GitHub[1]. The results are measured using the mir_eval library for Python [184].

7.4.3 Parameters

The algorithms depend on several parameters (see Tables 7.1, 7.2 and 7.3), these have been empirically chosen by running them on 10 randomly chosen mixtures of the vocalization corpus, which are played in room A. The distance between the speakers and microphones in the simulated room A (> 5 metres) and in the room used for the AC corpus is the same. However, this only holds for the time that the speaker in the AC corpus is standing still. When the speaker is moving the distance is less than in room A. The shorter duration of the files within this corpus allows for multiple parameters to be tested in quick succession. All the permutations were run till convergence.

The settings for the short-term Fourier transform (STFT, see Section 2.2.3) are kept the same as for the single channel speaker separation case (see Section 6.4) and are used for all the multichannel speaker separation algorithms. A microphone array with two microphones are used for the algorithms. The recordings of the AC corpus are downsampled to the same sampling frequency as those of the vocalization and MapTask corpora.

Frequency bins	1024
window	256
overlap	128
microphones	2

Table 7.1: General settings for the algorithms

DoA NMF/NTF

The DoA algorithms do not take any additional parameters except for the number of components which is set equal to the number of speakers i.e. two.

Cov NTF

The Cov NTF algorithm takes two parameters: S which is the number of sources and

[1] `https://github.com/TeunKrikke/SourceSeparationNMF`

K_s the number of components per source (see Table 7.2). In this case K is the sum of K_s. The parameters for K_s were in the range of 3 to 300 per source omitting the range of 20 to 100. This was done to test if a larger number has a positive influence on the separation of the sources. The results for the different parameters show that when the K_s value increased, the performance on the SAR and SDR measurements dropped. However, there was not much difference in performance between a K_s value of 5 or 10. Therefore, the smaller number was chosen to increase the speed of the algorithm.

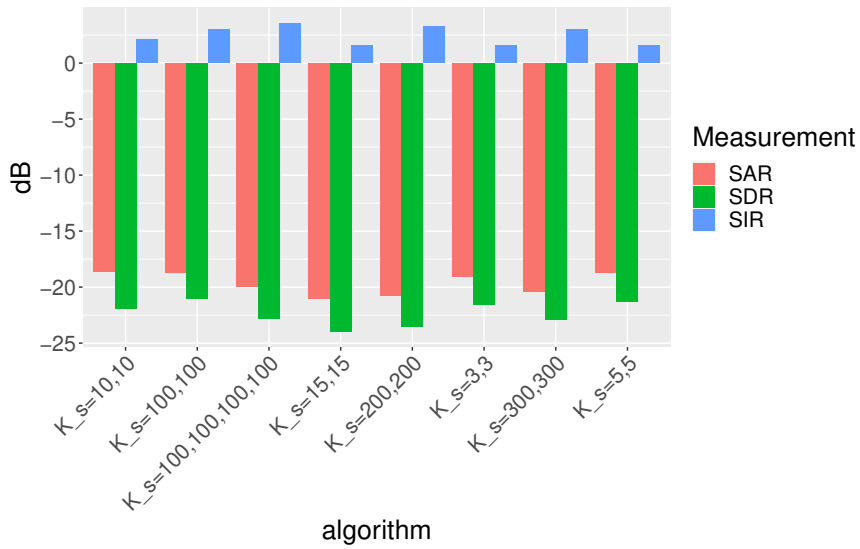


Figure 7.1: Cov NTF on the vocalization corpus with different parameters

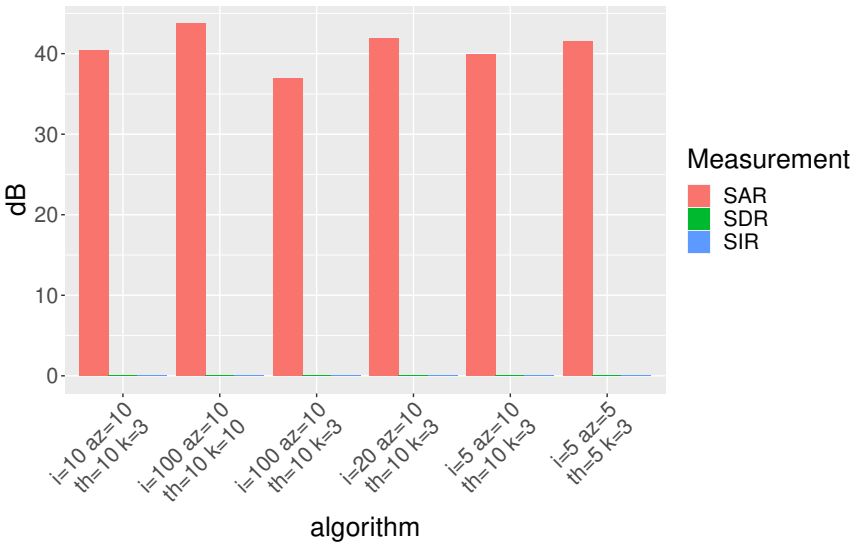
Parameter	value
S	2
K_s	5, 5
K	10

Table 7.2: General settings for the Cov NTF

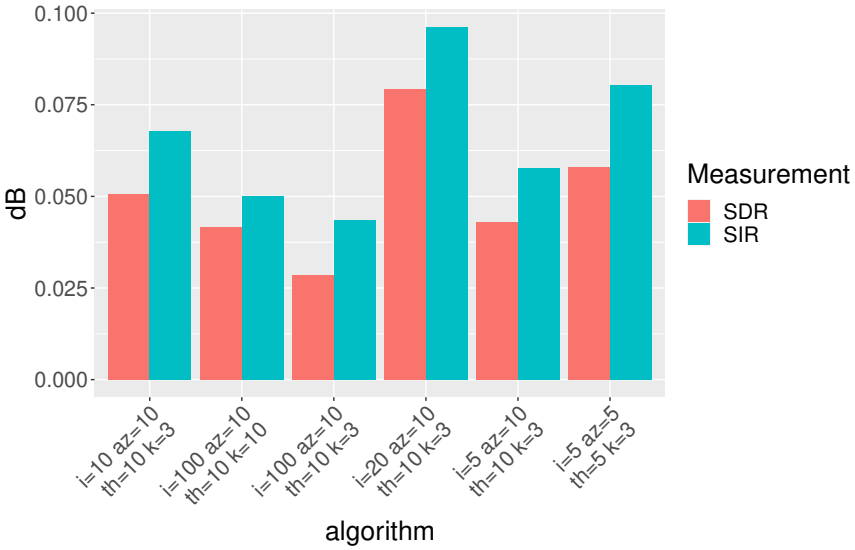
TDoA NTF

For the TDoA algorithm there are also three parameters, the azimuth, theta (combined define the number of look directions) and the number of components (see Table 7.3). The azimuth and theta are varied between 5 and 10, the results were checked after 5, 10, 20 and 100 iterations to see what the influence of the parameter was. After the TDoA algorithm ran on 10 files, the results showed that there was

little difference between it running for 5 iterations with 5 azimuth and 5 theta and it running for 100 iterations with 10 azimuth and 10 theta. Therefore, the lower number of iterations with lower numbers for azimuth and theta where chosen to increase the speed of the algorithm.



(a) The difference in SAR, SDR and SIR measurements using different settings for the TDoA NTF algorithm on the vocalization corpus



(b) This graph has the SAR measurement of figure 7.2a removed to show the difference between the parameters using the SDR and SIR measurments

Figure 7.2: TDoA NTF on the vocalization corpus with different parameters, where figure 7.2a shows the overview of the three measurements and figure 7.2b excludes the SAR measurement to concentrate on the SDR and SIR measurements.

Parameter	value
K	3
azimuth	5
theta	5

Table 7.3: General settings for the TDoA NTF

7.4.4 Performance measurements

To measure the performance of the algorithms, the same measurement algorithms as discussed in Sections 2.5 and 6.4, are used namely the signal-to-artifacts (SAR), signal-to-distortion (SDR) and signal-to-interference (SIR) ratios. Positive values indicate a better performance for all algorithms.

7.4.5 Experiments

The four algorithms (Cov NTF, DoA NMF, DoA NTF and TDoA NTF) were applied to the three corpora (vocalization, MapTask and AC corpora). This is to test the algorithms with different settings in different environments. The experiments on the vocalization and MapTaks corpora were run in a simulated environment without additional noise, whereas the AC corpus was recorded in a workshop environment with additional noise. This is to test how the algorithms deal with the additional noise, the difference in environment and the length of the files (10 s to 5 min).

As input to the algorithms, the STFT is used with a window size of 30 ms and 10 ms overlap between the windows. For Cov NTF, K_s is set to 5,5 (see Table 7.2). TDoA NTF uses a K of 3, azimuth of 5 and a theta of 5 (see Table 7.3).

7.5 Results

The two DoA (DoA NMF and DoA NTF) algorithms and two SCM (Cov NTF) algorithms are compared using the three different measurement: signal-to-artefact ratio (SAR), signal-to-distortion ratio (SDR) and signal-to-interference ratio (SIR). This comparison happens per corpus because of the different characteristics of each corpus. For the AC corpus, a comparison is made between the four different datasets

(original, no_echo, no_noise and nonoise_echo) that this corpus contains.

7.5.1 Vocalization corpus

The results on the vocalization corpus show that the covariance NTF method is least successful at removing artefacts and distortion compared to the three other methods (see Figure 7.3). Instead, it introduces new artefacts when the signals are separated. On the other hand, the TDoA algorithm, which also uses a spectral covariance matrix, is better at removing the artefacts. The TDoA allows for a more accurate calculation of the location of the source, making it easier for the algorithm to separate the sources. Both SCM algorithms fail to remove the distortion from the mixtures. Determining the time-difference of arrival allows the algorithm to reduce the number of artefacts in the separated speech. Whereas determining the cross-correlation between the microphones allows for a better interference detection. The two DoA algorithms are able to match the covariance algorithm on the interference measurement, but these need to run for a long time.

Overall, the two DoA algorithms perform well on the distortion and interference, meaning that they remove the distortion from the files and are able to separate the speakers. These two algorithms show a similar performance to when they are applied to the single channel problem (see Section 6.5). However, in this case the algorithms have to deal with reverberation as well, because the mixtures are coming from a simulated yet realistic environment. In comparison, they perform as well as the covariance algorithm on the interference but are better than both SCM algorithms (TDoA NTF and Covariance NTF) at dealing with the distortions. This would suggest that the SCM based techniques do not account for the distortions in the mixtures or introduce distortions when the mixtures are separated into different speakers. The latter happens when the two speakers are unmixed in the Wiener filter which means that the approximated masks are not correct.

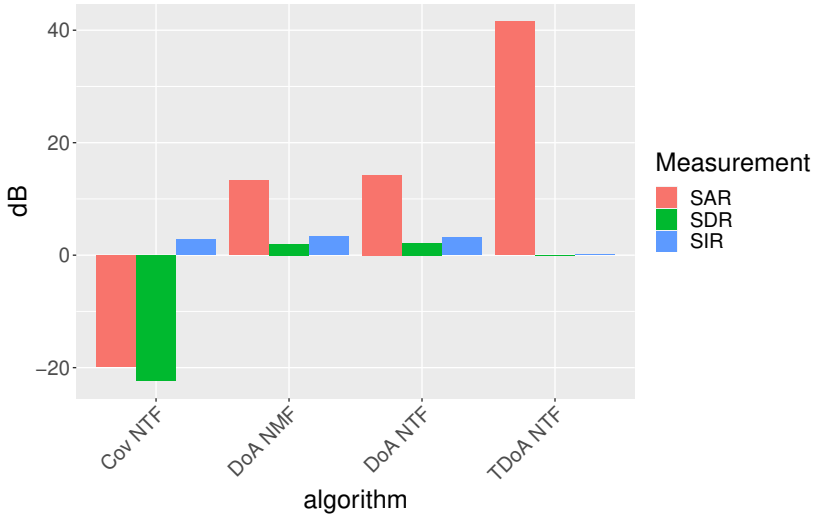
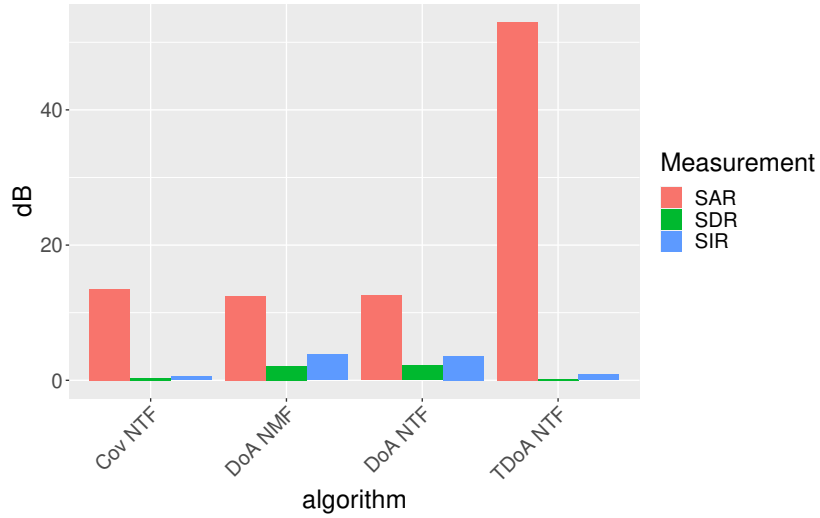


Figure 7.3: Multichannel algorithms on the vocalization corpus

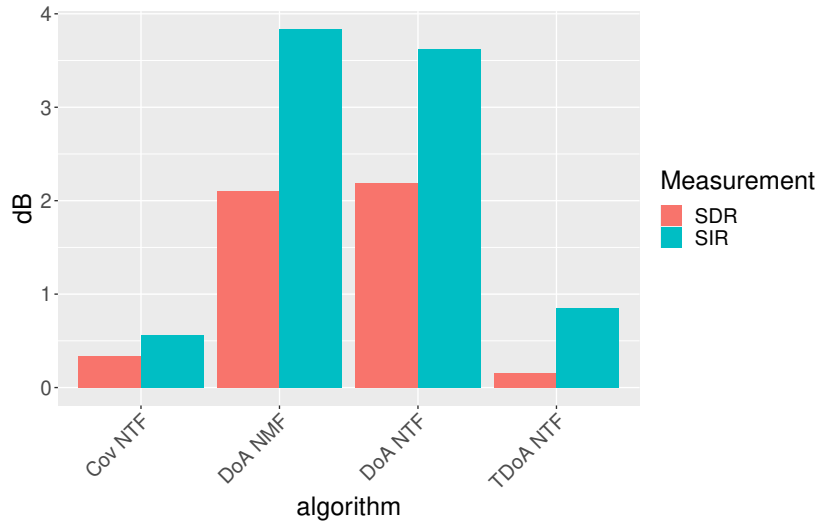
7.5.2 MapTask corpus

For the longer MapTask files, the results show that the TDoA NTF algorithm is better at removing the artefacts than the other two algorithms (see Figure 7.4a). However, looking at the results of the other two measurements, they show that the TDoA algorithm is worse at removing the distortions created by the separation process (see Figure 7.4b). The two DoA techniques (DoA NMF and DoA NTF) outperform the two SCM based techniques (Cov NTF and TDoA NTF). There is not much difference between the two DoA techniques. The results show that the DoA NMF technique performs similarly on removing the distortions but is better at removing the interference. This means that the NTF algorithm does not separate the sources as well as the NMF algorithm in this case.

Of the two SCM techniques, Cov NTF is better at removing the distortions than TDoA NTF is. This means that modelling the noise as a separate matrix allows for the removal of distortions that are otherwise introduced in the separated speakers. However, this process does not work for the interference which is higher in Cov NTF than it is in TDoA NTF. Therefore, TDoA NTF is better at locating the other speaker in the environment and subsequently removing that speaker.



(a) Multichannel algorithms on the MapTask corpus with all three measurements



(b) Multichannel algorithms on the MapTask corpus without the SAR measurement

Figure 7.4: Multichannel algorithms on the MapTask corpus with and without the signal-to-artifact ratio

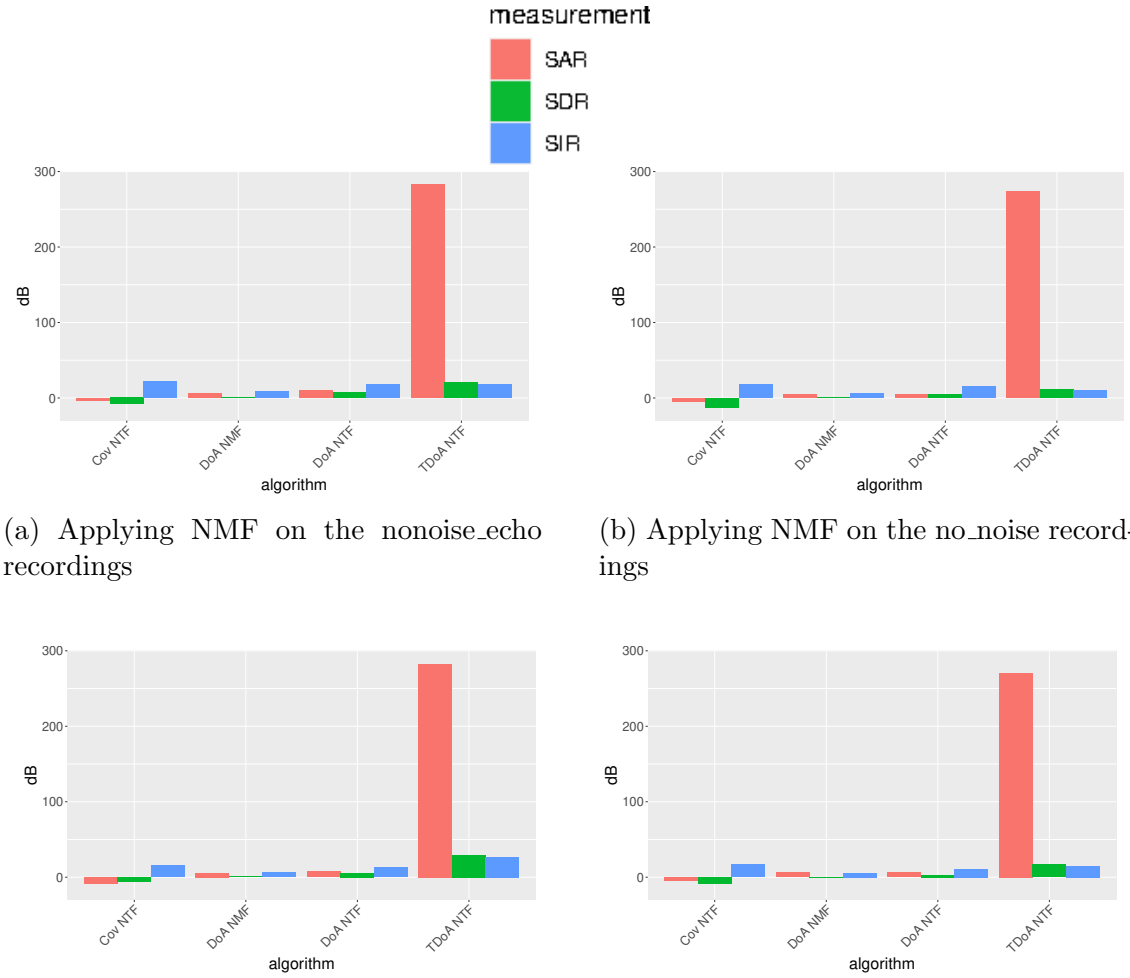
7.5.3 Acoustic Camera corpus

The AC corpus contains 4 different datasets each with their own properties. Looking at the overall results (see Figure 7.5), on the basis of the SAR the TDoA NTF algorithm outperforms the other algorithms. Therefore, this algorithm is best at removing artefacts from the separated recordings. However, when the SAR is ignored the results show a different picture (see Figure 7.6). Now on the basis of the SIR, the Cov NTF algorithm outperforms the rest except from when the algorithm is

applied to the no_echo dataset (see Figure 7.6c). With the latter recordings, the TDoA NTF algorithm outperforms the rest.

On the no_noise dataset, the DoA NTF algorithm as well as the Cov NTF algorithm outperform the TDoA NTF algorithm on the SIR measurement (see Figure 7.6b). Whereas on the nonoise_echo this measurement shows very little difference between the three NTF algorithms. The DoA NMF algorithm performs worse than the rest (see Figure 7.6a). Cov NTF introduces distortions into the separation, which is more noticeable in the no_noise and the original datasets than in the other two where the SDR value is close to -10 dB. This shows that the Cov NTF attributes certain frequencies to the wrong speaker. On the other hand, TDoA NTF has the worst performance on the no_noise dataset out of all four datasets, showing that the algorithms have trouble with separating the speakers in a reverberant environment. This is also seen in the original dataset.

The algorithms' best performance is on the no_echo dataset, where it has the highest values for both the SIR and SDR measurements. Overall, both DoA algorithms perform worse than the other two algorithms on all four datasets. Only on the no_noise dataset the result is comparable between the DoA NTF and the TDoA NTF techniques. Showing that both techniques are comparable in removing the second speaker but DoA NTF is worse at removing the distortions. The same goes for the Cov NTF algorithm where the SDR is the highest on the no_echo dataset. Showing that the Cov NTF technique has difficulties with removing the noise that is present in the recordings.



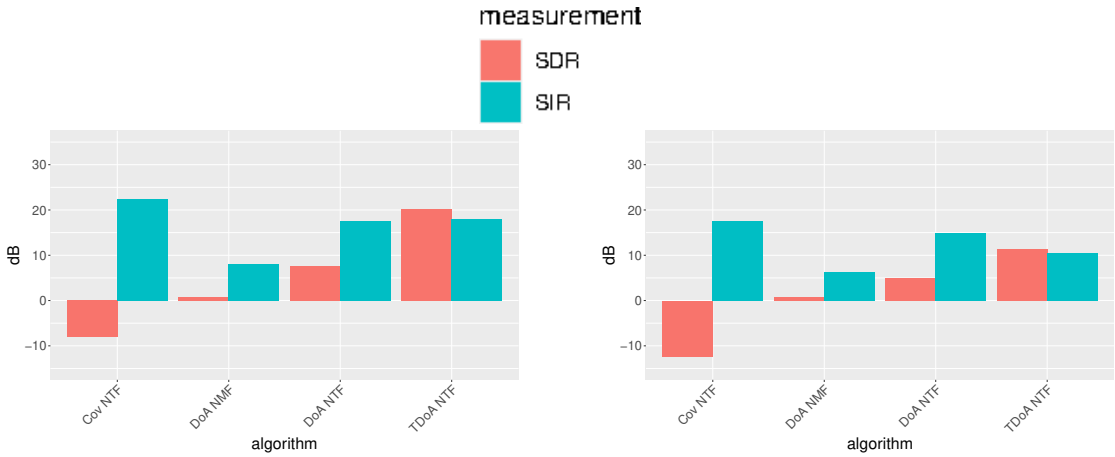
(a) Applying NMF on the nonoise_echo recordings

(b) Applying NMF on the no_noise recordings

(c) Applying NMF on the no_echo recordings

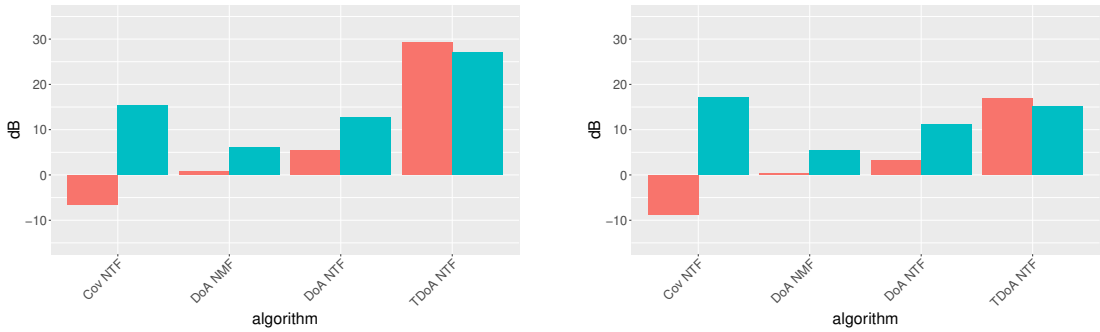
(d) Applying NMF on the original recordings

Figure 7.5: A comparison between different NMF techniques on the nonoise_echo, no_echo, no_noise and original AC corpus recordings.



(a) Applying NMF on the nonoise_echo recordings with SAR measurement removed

(b) Applying NMF on the no_noise recordings with SAR measurement removed



(c) Applying NMF on the no_echo recordings with SAR measurement removed

(d) Applying NMF on the original recordings with SAR measurement removed

Figure 7.6: A comparison between different NMF techniques on the nonoise_echo, no_echo, no_noise and original AC corpus recordings without the signal-to-artifacts ratio using the same data as Figure 7.5 however, now excluding the SAR measurement to show the difference in SDR and SIR measurements.

7.6 Conclusion

Multi-channel algorithms have the advantage of using the information coming from more than one microphone. This also increases the complexity of the algorithm and dependency on the right parameters. In simulated environments, the algorithms work independently of the speech or the length of speech, meaning that given an environment their performance is predictable and does not depend on whether the speaker speaks for 10 seconds or 5 minutes. For all algorithms, the change of environment changes the performance of the algorithm. A workshop environment where the speakers are recorded separately is easier for the TDoA algorithm to distinguish

between speakers than in a simulated environment. In the former setting, the reverberation of one speaker is mixed with that of another, meaning that they do not necessarily interact with, whereas in the simulated environment the reverberation of each speaker interacts with that of the other. Given the nature of this algorithm, it shows that it is important to take the reverberation of a speaker into account. A similar thing is seen with the Cov NTF algorithm that works better in the workshop environment than in the simulated one.

For all three corpora, the distance between the speakers and the microphones is kept the same, meaning that there is no performance degradation because the speakers were further away from the microphones. Instead, the AC corpus contains noise, whereas the corpora played in the simulated environment are clean from noise only containing reverberation.

When the TDoA algorithm is compared with the DoA algorithms, the results show that the TDoA algorithm is better at removing artefacts. However, the interference of a second source is still present when the speakers are separated. N.B. this is the second speaker used in the mixture not the second speaker already present in the files of the vocalization and MapTask corpora.

On the acoustic camera corpus, TDoA NTF also performs better on distortions, showing that when the audio is distorted TDoA NTF is able to separate the speakers and produces a cleaner result than the DoA algorithms. However, with shorter recordings the DoA algorithms outperform the SCM based techniques. When the recordings get longer then this pattern is less noticeable. Cov NTF is an algorithm that can be improved using different techniques to determine the correlation between the different microphones. The technique works well when there is noise and distortion in the recordings because this is modelled by a different matrix.

These results look very similar to the vocalization corpus. This shows that the algorithms work well independent of the data and their performance mainly depends on the environment.

Chapter 8

Conclusion

This chapter summarises the conclusions of previous chapters and compares them against the related work. Suggestions for future work are also described.

8.1 Discussion

This thesis concentrated on the introduction of novel algorithms for the dereverberation of speech signal and a new corpus which has been used for dereverberation and the separation of speakers. Both of these problems (dereverberation and speaker separation) are important for speech recognisers and robots in real world environments. When a speech recogniser is able to get a file that is free from noise, reverberation and overlapping speech, it is easier to create a correct transcription and let the robot execute the correct tasks. Correlation based dereverberation techniques were evaluated on different corpora, including the newly introduced corpus. Existing non-negative matrix techniques (NMF) and deep learning were applied to speaker separation and the influence of the cost function on the NMF result was compared. For deep learning, the performance of different algorithms on speaker separation was evaluated.

Corpus

The current corpora that are used for dereverberation and speaker separation often contain speech recorded in a lab environment with a close talk microphone. These corpora are clean, i.e. contain no reverberation or noise. To use a more complex

corpus for the evaluation of these problems a new corpus was presented in Chapter 4. This corpus contains realistic data with noise and reverberation. Instead of using a close-talk microphone, this corpus only contains far-field recordings. During the recording, the speakers are standing still as well as walking freely through the environment. This creates a more complex problem for speaker separation and dereverberation with the reverberations altering as the speaker moves. The corpus consists of four datasets, one original dataset, one without either noise or reverberation and one without noise and reverberation. These datasets can be used as a ground truth for the different problems making the corpus more versatile. Compared to the corpora described in the literature (see Section 3.1), this corpus is not limited to only dereverberation and speaker separation but can also be used for noise cancelling and speaker localization and tracking for example.

Dereverberation

The performance of the reverberation algorithms introduced in Chapter 5 was first measured on the TIMIT corpus in a simulated room to create a baseline for comparison against the algorithms discussed in Section 3.2. Correlation based algorithms can be easily modified from non-learning versions to create new learning versions. Combining the H_s correlation algorithm with non-negative tensor factorisation creates H_s NTF, a novel algorithm, which can adapt the approximation of the reverberant signal. The overall results of the non-learning algorithm are inconclusive, with none of the three algorithms outperforming the others. These algorithms are outperformed by the other algorithms within the literature, showing that only using the correlation between two microphones (or a clean signal and a reverberant signal) does not give a good approximation of the reverberant signal.

When these algorithms are used in a supervised learning setting and are combined with non-negative tensor factorisation, the results show an improvement to those of spectral subtraction and the non-negative matrix/tensor factorisation techniques in literature. This also shows that adapting the approximation of the reverberant signal improves the removal of the reverberation over using the correlation between two microphones. The downside of this technique is that it needs the ground truth signal as input next to the reverberant signal.

For purely unsupervised multichannel correlation-based learning techniques, weighted prediction error (WPE) and multiple input multiple output WPE (MIMO WPE) are compared with the novel Cauchy WPE and Cauchy MIMO WPE techniques. MIMO WPE is widely used for real-time dereverberation. Both WPE and MIMO WPE are based on the Gaussian distribution however, in Chapter 5 Cauchy WPE and Cauchy MIMO WPE are introduced. These latter two are based on the Cauchy distribution. For the WPE algorithm, this leads to an improvement in the performance. Therefore, changing the probability distribution affects the calculation of the reverberant signal. Both these techniques in their original form are comparable with the existing literature and Cauchy WPE and Cauchy MIMO WPE does not improve this.

Furthermore, both versions of WPE and MIMO WPE were tested in two rooms of different sizes. This was to investigate the influence of room size on the performance of the algorithm and to make a direct comparison with Parchami et al. [110]. However, there is no significant difference between the performance of the WPE and MIMO WPE algorithms on the rooms. This means that the algorithms are equally as effective in a larger room as in a smaller room. Another test was the difference in window functions used for the short time Fourier transform (STFT). This is because the original authors of both algorithms (Nakatani et al. [26] and Yoshioka et al. [33]) were using a different window function (Blackman) for the STFT, where normally a Hann window is used. In this case, the Hann window outperforms the Blackman window. This has as an advantage that the STFT is less likely to introduce artefacts into the frequency domain.

Single channel speaker separation

Two kinds of algorithms were compared in Chapter 6 for single channel speaker separation, supervised deep learning and unsupervised non-negative matrix factorisation (NMF). These algorithms were applied to three different corpora (vocalization, MapTask and AC). There are five different deep learning algorithms (convolution neural network, CNN, deep neural network, DNN, long short-term memory, LSTM, bidirectional long short-term memory and recurrent convolution neural network, RCNN). The deep learning algorithms are applied to the vocalization corpus and

non-negative matrix factorisation (NMF) is applied to all three corpora.

The differences between the corpora is the recording equipment, setting and task of the participants. For the vocalization corpus the recordings were made using a telephone, whereas the MapTask corpus used a close-talk microphone. The first two (vocalization and MapTask) corpora contain close-talk speech recorded in a lab environment, whereas the AC corpus contains far-field speech recorded in a workshop environment. In the first two corpora there is a second speaker present in the background of the recordings. This adds an additional complication for the algorithms to test their sensitivity to noise. Furthermore, for the AC corpus there are four datasets available to test the sensitivity of the different NMF algorithms to noise and reverberation.

For deep learning, it is important to use information from the past signal to create a separation between speakers. The three techniques that are able to use this outperform the other two. When this is expanded to using information from the future in a bidirectional LSTM, there is also a reduction in artefacts. When deep learning is compared to NMF, the results show that NMF out performs deep learning on the tested corpora. This is due to deep learning needing more data to be able to converge to a solution. However, the three corpora that were used for single channel speaker separation did not contain enough data for deep learning to converge to a solution.

Nonnegative matrix factorisation was tested with three different cost functions (squared Euclidean, Kullback-Leibler and Itakura-Saito). The cost functions have a different response to the speech signal. Where the Kullback-Leibler (KL) cost function relies on the larger data values, the Itakura-Saito (IS) is scale invariant, meaning that the smaller values in the recordings are of equal importance as the larger ones. Therefore, the latter should be better at separating the speakers. These cost functions are combined with sparsity, convolution and direction of arrival. In general, the performance on the AC corpus is lower than on the other two corpora showing that the AC corpus is a more difficult corpus for the different NMF algorithms. This is most clearly seen in the nonoise_echo dataset which shows that the distance between the speaker and microphone array has a negative influence on the performance. This also shows that different cost functions influence the performance

of the algorithms. The smaller values used by the IS cost function reduce the performance of the algorithms on the MapTask corpus and the orginal and nonoise_echo datasets of the AC corpus. This supports the idea that the larger values are more discriminative for the results.

Using directionality helps with the separation of speakers more in clean environments than in reverberant environments. The specific technique applied here uses the angle of arrival, which is more difficult to determine when there is reverberation present in the recordings. This is why directionality performs better on the vocalization and MapTask corpora.

Comparing the results of all techniques with the literature, it shows that the techniques applied to both the AC corpus and the vocalization corpus decrease in performance. For the AC corpus, this is because it contains far-field speech instead of close-talk speech and there is noise and reverberation present in the recordings. For the vocalization corpus, the main reason is the second speaker, which is interfering more than for the MapTask corpus. For example when comparing against the three techniques described by Magron et al. [143] only the sparse NMF with the Itakura-Saito cost function outperforms the ML-MUR technique on the signal-to-artifacts and signal-to-distortion ratios. This is also only the case for the MapTask corpus. This show that the other two corpora present the techniques with a bigger challenge to create a clean separation between the different speakers.

Looking at the results of deep learning, the networks presented in this thesis underperform those in the literature. However, this can be for two reasons. First, only a small amount of data is presented to the network making it difficult to converge to a solution. Second, looking at the result of the vocalization corpus on the NMF techniques it shows that this corpus presents a bigger challenge for an algorithm to create a clean separation between the different speakers. However looking ath the result of the LSTM network and comparing specifically the signal-to-interference ratio with those in the literature, it shows that his particular network shows competitive results towards those presented by Gang et al. [147] and Pandey et al. [161].

Multichannel speaker separation

Four different multi-channel techniques (Cov-NTF, DoA NMF, DoA NTF and TDoA NTF) have been tested on the same corpora as used for the single channel techniques. However, now a room simulator is used to provide reverberation to the recordings. The simulated room is the same size as the one used for creating the AC corpus (see Chapter 4). Multi-channel algorithms have the advantage of using the information coming from more than one microphone. This also increases the complexity of the algorithm, for it now needs to calculate where the sound is coming from and is more dependent on the right parameters. The environment has an influence on the performance of these algorithms.

Using the same simulated environment for two of the corpora (vocalization and MapTask), the performance of the algorithms stay the same. However, when comparing between a simulated with a real environment, a difference in performance is shown. For example, in the workshop environment it is easier for the TDoA NTF to distinguish between speakers than in the simulated environment. This shows that it is important to take the reverberation of a speaker into account, which in the workshop environment is more present than in the simulated environment. There is no degradation in performance when the distance from the speakers to the microphones is kept the same between the environments. The AC corpus contains noise whereas the simulated environment is clean from noise (only containing reverberation), showing that the TDoA NTF is better at dealing with noise than the other three algorithms. This is because it uses look directions to determine the speaker location and k-means clustering for separating the speakers.

Three of the four algorithms (DoA NMF, DoA NTF and TDoA NTF) perform very similarly to the existing literature (see Section 3.4). The TDoA NTF algorithm outperforms the version that ran in the simulated environment of Nikunen et al. [39]. Cov NTF, on the other hand, does not perform as well as the algorithms in the literature, but it shows improvement when it is applied to the real environment.

In this thesis a new corpus was presented, and different dereverberation and speaker separation algorithms were evaluated on it. This corpus focused on far-field speech in noise and reverberant environments. The new corpus shows the limits of the different techniques by presenting a challenging situation to perform

dereverberation and separation in. The techniques for dereverberation and speaker separation often matched the performance of the existing literature in simulated environments, but surpassed their performance in real environments.

8.2 Future work

The Cauchy WPE and Cauchy MIMO WPE algorithms, that change the estimation of the dereverberant signal, show small improvements in performance. Instead of using the Cauchy distribution for making modifications to the algorithms, the Gamma or Poisson distribution can be used for estimating the dereverberant signal. These distributions can be linked to different cost functions, where for example the Gamma distribution corresponds to a Itakura-Saito cost function. This allows for a comparison between these distributions with their respective cost functions and will show which performs better. In addition to this, it would allow for a comparison with the performance of the Itakura-Saito cost function on the speaker separation problem.

For both the multi- and single channel speaker separation techniques, it is important to choose the best performing cost function. Here the focus should be more on the multi-channel rather than on the single channel techniques because the existing multichannel non-negative matrix factorisation literature uses mainly the Kullback-Leibler cost function. In addition to this, the multi-channel algorithm is an extension of the single channel so can be easily adapted for this purpose. An example of this is the implementation of Cauchy NMF (Liutkus et al. [191]). This has not been implemented in a multichannel NTF algorithm but shows potential when used with WPE for dereverberation. The current version of the Cov NTF algorithm makes use of a correlation algorithm that assumes noise being present in the approximated signal (the so-called H_1 algorithm). This algorithm was tested on reverberant data for removing the reverberation. The outcome was compared to the H_s algorithm which assumes there is noise in the reverberant signal as well as in the approximation. Therefore, the H_1 algorithm could be replaced by the H_s algorithm to increase the accuracy of the Cov-NTF algorithm.

The corpus that has been presented is only a small corpus however, it highlights

the challenges on real data that may not be apparent in synthetic data. To make this usable for different algorithms (especially deep learning) this corpus needs to be expanded to include more participants. In addition to the number of recordings, the recording environments can be more varied including more noise from printers, television and machinery to create a more complex corpus. This complexity can also be increased by adding different environments, for example a sports hall, lecture theatre or a church where multiple speakers talk at the same time instead of having one speaker per recording.

The recordings of the corpus can be applied to speaker localisation for robots to learn where the speaker is located and subsequently tracking the speaker through the room.

The deep learning models are suited for applying to federated and transfer learning. For this purpose the models are available on GitHub. This is to train the models further and work towards convergence of the networks.

8.3 Conclusion

This thesis compared existing and novel dereverberation and speaker separation techniques on existing corpora and a new corpus. For dereverberation there is a dependency on whether a realistic or simulated environment is used for the testing of the algorithms. Where in a simulated environment the novel algorithms were not able to match the existing algorithm, in a realistic environment the opposite was the case. In addition to this, a learning based algorithm is better at removing the reverberation from the speech than a non-learning based algorithm. The learning phase of the algorithm allows it to adapt its prediction to fit the situation.

The introduced corpus has helped to show the limits of the non-negative matrix factorisation techniques as well as the importance of the cost function. In the multi-channel situation, the corpus allows comparison of a simulated environment with a realistic environment. In the latter, the algorithms show a decrease in performance when the reverberation of the environment is manually removed. This corpus is recorded with a device that is capable of making high quality recordings but unsuitable for positioning on a robot. However, when the recordings are used for training

a deep learning algorithm or a technique that is not concerned with the exact location of the microphones but concentrates on the recordings or viewing angles of the microphone (e.g. weighted prediction error or covariance NTF), then the recordings can be used for training the algorithm and the robot can be equipped with a smaller microphone array containing a similar configuration (stereo, triangular or square) of the microphones as used in training.

The corpus can used for far-field speech detection. However, the high sample rate allows it to be used for breath detection which can be used for detecting a possible interruption point for the robot or from the speaker. This creates a more natural dialogue between a robot and a speaker. Simialrly the high samplerate provides more features for emotion detection and the cancellation of unwanted interference.

The limits of the existing algorithms were shown by testing using realistic environments. In these environments, it was possible for the novel algorithms to match or improve the performance of the existing algorithms. Many of the existing algorithms have been evaluated on corpora containing near field speech. Instead, this thesis showed that on corpora containing far field speech, these algorithms are still able produce competitive results when the speaker is further away from the microphones.

Bibliography

[1] Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte, "Performance measurement in blind audio source separation," *IEEE transactions on audio, speech, and language processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[2] Antony W. Rix, John G. Beerends, Michael P. Hollier, and Andries P. Hekstra, "Perceptual evaluation of speech quality (pesq): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs," *Rec. ITU-T P. 862*, 2001.

[3] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett, "Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1," *NASA STI/Recon technical report n*, vol. 93, 1993.

[4] Lawrence E. Kinsler, Austin R. Frey, Alan B. Coppens, and James V. Sanders, *Fundamentals of acoustics*, Wiley, 4th edition, 1999.

[5] Tony F W Embleton, "Tutorial on sound propagation outdoors," *The Journal of the Acoustical Society of America*, vol. 100, no. 1, pp. 31–48, 1996.

[6] Daniel Jurafsky and James H Martin, "Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition," .

[7] Noah D Stein, "Nonnegative Tensor Factorization for Directional Blind Audio Source Separation," *stat*, vol. 1050, pp. 18, 2014.

[8] Sirko Molau, Michael Pitz, Ralf Schluter, and Hermann Ney, "Computing mel-frequency cepstral coefficients on the power spectrum," in *2001 IEEE in-*

ternational conference on acoustics, speech, and signal processing. Proceedings (cat. No. 01CH37221). IEEE, 2001, vol. 1, pp. 73–76.

[9] S. Alexander and Z. Rhee, "An analysis of finite precision effects for the autocorrelation method and burg's method of linear prediction," in *ICASSP '87. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1987, vol. 12, pp. 336–339.

[10] Hynek Hermansky, "Perceptual linear predictive (plp) analysis of speech," *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[11] Khiet P Truong and David A Van Leeuwen, "Automatic discrimination between laughter and speech," *Speech Communication*, vol. 49, no. 2, pp. 144–158, 2007.

[12] Guillermo Cámbara, Jordi Luque, and Mireia Farrús, "Convolutional speech recognition with pitch and voice quality features," *arXiv preprint arXiv:2009.01309*, 2020.

[13] Ethem Alpaydin, *Introduction to machine learning*, MIT press, 2020.

[14] Jont B Allen and Lawrence R Rabiner, "A unified approach to short-time fourier analysis and synthesis," *Proceedings of the IEEE*, vol. 65, no. 11, pp. 1558–1564, 1977.

[15] M Portnoff, "Time-frequency representation of digital signals and systems based on short-time fourier analysis," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 1, pp. 55–69, 1980.

[16] Fredric J Harris, "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978.

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016, `http://www.deeplearningbook.org`.

[18] Dragoslav S Mitrinovic, Josip Pecaric, and Arlington M Fink, *Classical and new inequalities in analysis*, vol. 61, Springer Science & Business Media, 2013.

[19] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 03 1951.

[20] F. Itakura and S. Saito, "Analysis synthesis telephony based upon the maximum likelihood method," 1968.

[21] Ayanendranath Basu, Ian R Harris, Nils L Hjort, and MC Jones, "Robust and efficient estimation by minimising a density power divergence," *Biometrika*, vol. 85, no. 3, pp. 549–559, 1998.

[22] MC Jones, Nils Lid Hjort, Ian R Harris, and Ayanendranath Basu, "A comparison of related density-based minimum divergence estimators," *Biometrika*, vol. 88, no. 3, pp. 865–873, 2001.

[23] Lev M Bregman, "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming," *USSR computational mathematics and mathematical physics*, vol. 7, no. 3, pp. 200–217, 1967.

[24] Andrzej Cichocki and Shun ichi Amari, "Families of alpha- beta- and gamma-divergences: Flexible and robust measures of Similarities," *Entropy*, vol. 12, no. 6, pp. 1532–1568, 2010.

[25] G. E. P. Box, "Non-Normality and Tests on Variances," *Biometrika*, vol. 40, no. 3/4, pp. 318, 1953.

[26] Tomohiro Nakatani, Takuya Yoshioka, Keisuke Kinoshita, Masato Miyoshi, and Biing-Hwang Juang, "Speech dereverberation based on variance-normalized delayed linear prediction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 7, pp. 1717–1731, 2010.

[27] Saeed Gazor and Wei Zhang, "Speech probability distribution," *IEEE Signal Processing Letters*, vol. 10, no. 7, pp. 204–207, 2003.

[28] Mohammed Usman, Mohammed Zubair, Mohammad Shiblee, Paul Rodrigues, and Syed Jaffar, "Probabilistic modeling of speech in spectral domain using maximum likelihood estimation," *Symmetry*, vol. 10, no. 12, pp. 750, 2018.

[29] Jont B Allen and David A Berkley, "Image method for efficiently simulating small-room acoustics," *The Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, 1979.

[30] Micah Taylor, Anish Chandak, Qi Mo, Christian Lauterbach, Carl Schissler, and Dinesh Manocha, "i-sound: Interactive gpu-based sound auralization in dynamic scenes," Tech. Rep., Tech. Rep. TR10-006, 2010.

[31] Alex Southern, Samuel Siltanen, Damian T Murphy, and Lauri Savioja, "Room impulse response synthesis and validation using a hybrid acoustic model," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 9, pp. 1940–1952, 2013.

[32] Quentin Leclere, NB Roozen, and Céline Sandier, "On the use of the hs estimator for the experimental assessment of transmissibility matrices," *Mechanical Systems and Signal Processing*, vol. 43, no. 1-2, pp. 237–245, 2014.

[33] Takuya Yoshioka and Tomohiro Nakatani, "Generalization of multi-channel linear prediction methods for blind mimo impulse response shortening," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 10, pp. 2707–2720, 2012.

[34] Cédric Févotte and Jérôme Idier, "Algorithms for nonnegative matrix factorization with the beta-divergence," *CoRR*, vol. abs/1010.1, 2010.

[35] Wikipedia, *NMF image*, 2013 (accessed April 15, 2020).

[36] Julian Eggert and Edgar Korner, "Sparse coding and NMF," in *IEEE International Joint Conference on Neural Networks, 2004*. IEEE, 2004, vol. 4, pp. 2529–2533.

[37] Mikkel N Schmidt, "Speech separation using non-negative features and sparse non-negative matrix factorization," *Elsevier*, 2007.

[38] Alexey Ozerov and Cédric Févotte, "Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 550–563, 2010.

[39] Joonas Nikunen and Tuomas Virtanen, "Direction of arrival based spatial covariance model for blind sound source separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 3, pp. 727–739, 2014.

[40] Alexey Ozerov, Cédric Févotte, and Emmanuel Vincent, "An introduction to multichannel nmf for audio source separation," in *Audio Source Separation*, pp. 73–94. Springer, 2018.

[41] Georg Thimm and Emile Fiesler, "High-order and multilayer perceptron initialization," *IEEE Transactions on Neural Networks*, vol. 8, no. 2, pp. 349–359, 1997.

[42] Javatpoint, "Single layer perceptron," `https://www.javatpoint.com/single-layer-perceptron-in-tensorflow`, 2008, [Online; accessed 09-02-2021].

[43] d2l, "Multilayer perceptron," `http://d2l.ai/chapter_multilayer-perceptrons/mlp.html`, 2008, [Online; accessed 09-02-2021].

[44] Anastasia Kyrykovych, "Deep neural networks," `https://www.kdnuggets.com/2020/02/deep-neural-networks.html`, 2008, [Online; accessed 09-02-2021].

[45] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[46] Michael I Jordan, "Serial order: A parallel distributed processing approach," in *Advances in psychology*, vol. 121, pp. 471–495. Elsevier, 1997.

[47] Jeffrey L Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[48] Christopher Olah, "Understanding LSTM networks," `http://colah.github.io/posts/2015-08-Understanding-LSTMs`, 2008, [Online; accessed 09-02-2021].

[49] Tim Jones, "Elman and Jordan RNN," `https://developer.ibm.com/articles/cc-cognitive-recurrent-neural-networks/`, 2017, [Online; accessed 23-02-2021].

[50] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio, "On the difficulty of training recurrent neural networks," *30th International Conference on Machine Learning, ICML 2013*, , no. PART 3, pp. 2347–2355, 2013.

[51] Yoshua Bengio, Patrice Simard, and Paolo Frasconi, "Learning Long-Term Dependencies with Gradient Descent is Difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.

[52] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutnik, Bas R. Steunebrink, and Jurgen Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[53] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[54] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[55] Dana H Ballard, "Modular learning in neural networks.," in *AAAI*, 1987, pp. 279–284.

[56] Jürgen Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[57] Chervinskii, "AutoEncoder," `https://commons.wikimedia.org/w/index.php?curid=45555552`, 2008, [Online; accessed 09-02-2021].

[58] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, pp. 2672–2680, 2014.

[59] Google, "Generative adversarial networks," `https://developers.google.com/machine-learning/gan/gan_structure`, 2008, [Online; accessed 09-02-2021].

[60] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R Hershey, "Sdr–half-baked or well done?," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 626–630.

[61] Antony W Rix, John G Beerends, Michael P Hollier, and Andries P Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*. IEEE, 2001, vol. 2, pp. 749–752.

[62] Nobuhiko Kitawaki, Hiromi Nagabuchi, and Kenzo Itoh, "Objective quality evaluation for low-bit-rate speech coding systems," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 2, pp. 242–248, 1988.

[63] Colin H Hansen, "Fundamentals of acoustics," *Occupational Exposure to Noise: Evaluation, Prevention and Control. World Health Organization*, pp. 23–52, 2001.

[64] D Siano, M Viscardi, and MA Panza, "Experimental acoustic measurements in far field and near field conditions: characterization of a beauty engine cover," *Recent Advances in Fluid Mechanics and Thermal Engineering*, pp. 50–57, 2014.

[65] Simon Doclo and Marc Moonen, "Design of far-field and near-field broadband beamformers using eigenfilters," *Signal Processing*, vol. 83, no. 12, pp. 2641–2673, 2003.

[66] Yan Zhao, DeLiang Wang, Buye Xu, and Tao Zhang, "Monaural speech dereverberation using temporal convolutional networks with self attention," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2020.

[67] David Gelbart and Nelson Morgan, "Double the trouble: handling noise and reverberation in far-field automatic speech recognition," in *Seventh International Conference on Spoken Language Processing*, 2002.

[68] Yuki Tamai, Yoko Sasaki, Satoshi Kagami, and Hiroshi Mizoguchi, "Three Ring Microphone Array for 3D Sound Localization and Separation for Mobile Robot Audition," .

[69] Guinness world Records, "largest-microphone-array," .

[70] Sorama, "World's largest microphone array," .

[71] Heinrich W Löllmann, Alastair H Moore, Patrick A Naylor, Boaz Rafaely, Radu Horaud, Alexandre Mazel, and Walter Kellermann, "Microphone array signal processing for robot audition," in *Hands-free Speech Communications and Microphone Arrays - HSCMA*, 2017.

[72] Aldebaran, *Sound localization.*

[73] Softbank Robotics, *Pepper.*

[74] Jon Barker, Ricard Marxer, Emmanuel Vincent, and Shinji Watanabe, "The third 'CHiME' speech separation and recognition challenge: Analysis and outcomes," *Computer Speech and Language*, vol. 46, pp. 605–626, 2017.

[75] Yan Hui Tu, Jun Du, Qing Wang, Xiao Bao, Li Rong Dai, and Chin Hui Lee, "An information fusion framework with multi-channel feature concatenation and multi-perspective system combination for the deep-learning-based robust recognition of microphone array speech," *Computer Speech and Language*, vol. 46, pp. 517–534, 2017.

[76] Charles Knapp and Glifford Carter, "The generalized correlation method for estimation of time delay," *IEEE transactions on acoustics, speech, and signal processing*, vol. 24, no. 4, pp. 320–327, 1976.

[77] Byoungho Kwon, Youngjin Park, and Youn-sik Park, "Analysis of the gcc-phat technique for multiple sources," in *ICCAS 2010*. IEEE, 2010, pp. 2070–2073.

[78] Benedikt Loesch and Bin Yang, "Adaptive segmentation and separation of determined convolutive mixtures under dynamic conditions," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2010, pp. 41–48.

[79] Jacob Benesty, Jingdong Chen, and Yiteng Huang, "A generalized mvdr spectrum," *IEEE Signal Processing Letters*, vol. 12, no. 12, pp. 827–830, 2005.

[80] Charles Blandin, Alexey Ozerov, and Emmanuel Vincent, "Multi-source tdoa estimation in reverberant audio using angular spectra and clustering," *Signal Processing*, vol. 92, no. 8, pp. 1950–1960, 2012.

[81] Keisuke Kinoshita, Marc Delcroix, Takuya Yoshioka, Tomohiro Nakatani, Emanuël Habets, Reinhold Haeb-Umbach, Volker Leutnant, Armin Sehr, Walter Kellermann, Roland Maas, et al., "The reverb challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 2013, pp. 1–4.

[82] Jon Barker, Shinji Watanabe, Emmanuel Vincent, and Jan Trmal, "The fifth 'chime' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. Interspeech 2018*, 2018, pp. 1561–1565.

[83] Nattanun Chanchaochai, Christopher Cieri, Japhet Debrah, Hongwei Ding, Yue Jiang, Sishi Liao, Mark Liberman, Jonathan Wright, Jiahong Yuan, Juhong Zhan, and Yuqing Zhan, "Globaltimit: Acoustic-phonetic datasets for the world's languages," in *Proc. Interspeech 2018*, 2018, pp. 192–196.

[84] Douglas B Paul and Janet M Baker, "The design for the wall street journal-based csr corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[85] Peter Kabal, "Tsp speech database," 2002.

[86] Mike Lincoln, Iain Mccowan, Jithendra Vepa, and Hari Krishna Maganti, "THE MULTI-CHANNEL WALL STREET JOURNAL AUDIO VISUAL CORPUS (MC-WSJ-AV): SPECIFICATION AND INITIAL EXPERIMENTS," *ASRU*, pp. 357–362, 2005.

[87] Hugues Salamin, Anna Polychroniou, and Alessandro Vinciarelli, "Automatic detection of laughter and fillers in spontaneous mobile phone conversations," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2013.* IEEE, 2013, pp. 4282–4287.

[88] Anne H Anderson, Miles Bader, Ellen Gurman Bard, Elizabeth Boyle, Gwyneth Doherty, Simon Garrod, Stephen Isard, Jacqueline Kowtko, Jan McAllister, Jim Miller, Catherine Sotillo, Henry S Thompson, and Regina Weinert, "The Hcrc Map Task Corpus," *Language and Speech*, vol. 34, no. 4, pp. 351–366, 1991.

[89] Djamel Mostefa, Nicolas Moreau, Khalid Choukri, Ae Gerasimos, Potamianos Ae, Stephen M Chu, Ae Ambrish, Tyagi Ae, Josep R Casas, Jordi Turmo, Ae Luca, Cristoforetti Ae, Francesco Tobia, Ae Aristodemos, Pnevmatikakis Ae, Vassilis Mylonakis, Ae Fotios, Talantzis Ae, Susanne Burger, Rainer Stiefelhagen, Ae Keni, Bernardin Ae, Cedrick Rochet, D Mostefa, Á N Moreau, Á K Choukri, N Moreau, K Choukri, G Potamianos, Á S M Chu, Á A Tyagi, S M Chu, A Tyagi, J R Casas, Á J Turmo, J Turmo, L Cristoforetti, Á F Tobia, F Tobia, A Pnevmatikakis, Á V Mylonakis, Á F Talantzis, V Mylonakis, F Talantzis, S Burger, R Stiefelhagen, Á K Bernardin, Á C Rochet, K Bernardin, and C Rochet, "The CHIL audiovisual corpus for lecture and meeting analysis inside smart rooms," *Lang Resources & Evaluation*, vol. 41, pp. 389–407, 2007.

[90] I McCowan, J Carletta, W Kraaij, S Ashby, S Bourban, M Flynn, M Guillemot, T Hain, J Kadlec, V Karaiskos, et al., "The ami meeting corpus," in *Proceedings of Measuring Behavior 2005, 5th International Conference on Methods and Techniques in Behavioral Research.* Noldus Information Technology, 2005, pp. 137–140.

[91] Jon Barker, Ricard Marxer, Emmanuel Vincent, and Shinji Watanabe MERL, "THE THIRD 'CHIME' SPEECH SEPARATION AND RECOGNITION CHALLENGE: DATASET, TASK AND BASELINES," .

[92] Alex Stupakov, Evan Hanusa, Jeff Bilmes, and Dieter Fox, "COSINE -A CORPUS OF MULTI-PARTY CONVERSATIONAL SPEECH IN NOISY ENVIRONMENTS," .

[93] Colleen Richey, Maria A. Barrios, Zeb Armstrong, Chris Bartels, Horacio Franco, Martin Graciarena, Aaron Lawson, Mahesh Kumar Nandwana, Allen Stauffer, Julien van Hout, Paul Gamble, Jeffrey Hetherly, Cory Stephenson, and Karl Ni, "Voices obscured in complex environmental settings (voices) corpus," in *Proc. Interspeech 2018*, 2018, pp. 1566–1570.

[94] Emmanuel Vincent, Shoko Araki, and Pau Bofill, "The 2008 signal separation evaluation campaign: A community-based approach to large-scale evaluation," in *International Conference on Independent Component Analysis and Signal Separation*. Springer, 2009, pp. 734–741.

[95] Shoko Araki, Fabian Theis, Guido Nolte, Dominik Lutter, Alexey Ozerov, Vikrham Gowreesunker, Hiroshi Sawada, and Ngoc QK Duong, "The 2010 signal separation evaluation campaign (sisec2010): biomedical source separation," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2010, pp. 123–130.

[96] Zhong-Qiu Wang, Xueliang Zhang, and DeLiang Wang, "Robust tdoa estimation based on time-frequency masking and deep neural networks," in *Proc. Interspeech 2018*, 2018, pp. 322–326.

[97] Wolfgang Mack, Soumitro Chakrabarty, Fabian-Robert Stöter, Sebastian Braun, Bernd Edler, and Emanuël Habets, "Single-channel dereverberation using direct mmse optimization and bidirectional lstm networks," in *Proc. Interspeech 2018*, 2018, pp. 1314–1318.

[98] Yi Luo and Nima Mesgarani, "Real-time single-channel dereverberation and separation with time-domain audio separation network," in *Proc. Interspeech 2018*, 2018, pp. 342–346.

[99] Hao Zhang and DeLiang Wang, "Deep learning for acoustic echo cancellation in noisy and double-talk scenarios," in *Proc. Interspeech 2018*, 2018, pp. 3239–3243.

[100] Randy Gomez, Keisuke Nakamura, and Kazuhiro Nakadai, "Dereverberation robust to speaker's azimuthal orientation in multi-channel human-robot communication," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 3439–3444.

[101] Randy Gomez, Levko Ivanchuk, Keisuke Nakamura, Takeshi Mizumoto, and Kazuhiro Nakadai, "Dereverberation for active human-robot communication robust to speaker's face orientation," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[102] Ryu Takeda, Kazuhiro Nakadai, Toru Takahashi, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G Okuno, "Ica-based efficient blind dereverberation and echo cancellation method for barge-in-able robot audition," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009, pp. 3677–3680.

[103] Chenxing Li, Tieqiang Wang, Shuang Xu, and Bo Xu, "Single-channel speech dereverberation via generative adversarial training," in *Proc. Interspeech 2018*, 2018, pp. 1309–1313.

[104] Ori Ernst, Shlomo E Chazan, Sharon Gannot, and Jacob Goldberger, "Speech dereverberation using fully convolutional networks," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 390–394.

[105] Peter Guzewich, Stephen Zahorian, Xiao Chen, and Hao Zhang, "Cross-corpora convolutional deep neural network dereverberation preprocessing for speaker verification and speech enhancement," in *Proc. Interspeech 2018*, 2018, pp. 1329–1333.

[106] Ina Kodrasi and Hervé Bourlard, "Single-channel late reverberation power spectral density estimation using denoising autoencoders," in *Proc. Interspeech 2018*, 2018, pp. 1319–1323.

[107] Disong Wang and Yuexian Zou, "Joint noise and reverberation adaptive learning for robust speaker doa estimation with an acoustic vector sensor," in *Proc. Interspeech 2018*, 2018, pp. 821–825.

[108] Lukas Drude, Christoph Boeddeker, Jahn Heymann, Reinhold Haeb-Umbach, Keisuke Kinoshita, Marc Delcroix, and Tomohiro Nakatani, "Integrating neural network based beamforming and weighted prediction error dereverberation," in *Proc. Interspeech 2018*, 2018, pp. 3043–3047.

[109] Saeed Mosayyebpour and Francesco Nesta, "Neural-network supervised maximum likelihood-based on-line dereverberation," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1552–1556.

[110] Mahdi Parchami, Wei-Ping Zhu, and Benoit Champagne, "Speech dereverberation using linear prediction with estimation of early speech spectral variance," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 504–508.

[111] Ante Jukić, Toon van Waterschoot, Timo Gerkmann, and Simon Doclo, "Multi-channel linear prediction-based speech dereverberation with sparse priors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 9, pp. 1509–1520, 2015.

[112] Christoph Boeddeker, Tomohiro Nakatani, Keisuke Kinoshita, and Reinhold Haeb-Umbach, "Jointly optimal dereverberation and beamforming," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 216–220.

[113] Sahar Hashemgeloogerdi and Sebastian Braun, "Joint beamforming and reverberation cancellation using a constrained kalman filter with multichannel linear prediction," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 481–485.

[114] Alejandro Cohen, Anna Barnov, Shmulik Markovich-Golan, and Peter Kroon, "Joint beamforming and echo cancellation combining qrd based multichannel aec and mvdr for reducing noise and non-linear echo," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 6–10.

[115] Sharon Gannot and Marc Moonen, "Subspace methods for multimicrophone speech dereverberation," *EURASIP Journal on Advances in Signal Processing*, vol. 2003, no. 11, pp. 769285, 2003.

[116] Nikhil M, Rajbabu Velmurugan, and Preeti Rao, "A non-convolutive nmf model for speech dereverberation," in *Proc. Interspeech 2018*, 2018, pp. 1324–1328.

[117] Sanna Wager and Minje Kim, "Collaborative speech dereverberation: Regularized tensor factorization for crowdsourced multi-channel recordings," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1532–1536.

[118] Katia Lebart, Jean-Marc Boucher, and Philip N Denbigh, "A new method based on spectral subtraction for speech dereverberation," *Acta Acustica united with Acustica*, vol. 87, no. 3, pp. 359–366, 2001.

[119] Emanuel AP Habets, "Single-channel speech dereverberation based on spectral subtraction," in *Proceedings of the 15th Annual Workshop on Circuits, Systems and Signal Processing (ProRISC)*. Citeseer, 2004.

[120] Hagai Attias, John C Platt, Alex Acero, and Li Deng, "Speech denoising and dereverberation using probabilistic models," in *Advances in neural information processing systems*, 2001, pp. 758–764.

[121] Alberto Carini, Stefania Cecchi, Alessandro Terenzi, and Simone Orcioni, "On room impulse response measurement using perfect sequences for wiener nonlinear filters," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 982–986.

[122] Suresh Subramaniam, Athina P Petropulu, and Christopher Wendt, "Cepstrum-based deconvolution for speech dereverberation," *IEEE transactions on speech and audio processing*, vol. 4, no. 5, pp. 392–396, 1996.

[123] Scott Griebel and Michael Brandstein, "Wavelet transform extrema clustering for multi-channel speech dereverberation," in *IEEE Workshop on Acoustic Echo and Noise Control*. Citeseer, 1999, pp. 27–30.

[124] Carlos Avendano and Hynek Hermansky, "Study on the dereverberation of speech based on temporal envelope filtering," in *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*. IEEE, 1996, vol. 2, pp. 889–892.

[125] Bradford W Gillespie, Henrique S Malvar, and Dinei AF Florêncio, "Speech dereverberation via maximum-kurtosis subband adaptive filtering," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*. IEEE, 2001, vol. 6, pp. 3701–3704.

[126] Ricky Der, "Blind signal separation," *Materials for Laboratory of Telecommunications & Signal Processing of the McGill University, Montreal*, 2001.

[127] Marcel Joho and Heinz Mathis, "Joint diagonalization of correlation matrices by using gradient methods with application to blind signal separation," in *Sensor Array and Multichannel Signal Processing Workshop Proceedings, 2002*. IEEE, 2002, pp. 273–277.

[128] Stefan Winter, Hiroshi Sawada, and Shoji Makino, "Geometrical understanding of the pca subspace method for overdetermined blind source separation," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03)*. IEEE, 2003, vol. 2, pp. II–769.

[129] Nikolaos Mitianoudis and Michael E. Davies, "Audio source separation of convolutive mixtures," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 5, pp. 489–497, 2003.

[130] Nikolaos Mitianoudis, "A generalized directional Laplacian distribution: Estimation, mixture models and audio source separation," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 9, pp. 2397–2408, nov 2012.

[131] Nikolaos Mitianoudis and Tania Stathaki, "Batch and online underdetermined source separation using Laplacian mixture models," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 6, pp. 1818–1832, 2007.

[132] Nikolaos Mitianoudis and Tania Stathaki, "Underdetermined source separation using mixtures of warped laplacians," *Proceedings of the 7th international conference on Independent component analysis and signal separation*, , no. 1, pp. 236–243, 2007.

[133] Emmanuel Vincent and Xavier Rodet, "Underdetermined Source Separation with Structured Source Priors," *Independent Component Analysis and Blind Signal Separation*, pp. 327–334, 2004.

[134] Nikolaos Mitianoudis, King College, and Mike Davies, "A fixed point solution for convolved audio source separation," *Signal Processing*, , no. October, pp. 21–24, 2001.

[135] Paris Smaragdis and Michael Casey, "Audio/visual independent components," in *Proc. ICA*, 2003, pp. 709–714.

[136] Mike E Davies and Christopher J James, "Source separation using single channel ICA," *Signal Processing*, vol. 87, no. 8, pp. 1819–1832, 2007.

[137] Aapo Hyvärinen, "The Fixed-Point Algorithm and Maximum Likelihood Estimation for Independent Component Analysis," *Neural Processing Letters*, vol. 10, no. 1, pp. 1–5, 1999.

[138] Nikoloas Mitianoudis and Mike Davies, "New fixed-point ica algorithms for convolved mixtures," *ICA'01*, pp. 633–638, 2001.

[139] Tian Shi, Kyeongpil Kang, Jaegul Choo, and Chandan K Reddy, "Short-text topic modeling via non-negative matrix factorization enriched with local word-context correlations," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 1105–1114.

[140] David Guillamet and Jordi Vitria, "Non-negative matrix factorization for face recognition," in *Catalonian Conference on Artificial Intelligence*. Springer, 2002, pp. 336–344.

[141] Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu, "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis," *Neural computation*, vol. 21, no. 3, pp. 793–830, 2009.

[142] Cédric Févotte and A. Taylan Cemgil, "Nonnegative matrix factorizations as probabilistic inference in composite models," *European Signal Processing Conference*, , no. Eusipco, pp. 1913–1917, 2009.

[143] Paul Magron and Tuomas Virtanen, "Expectation-maximization algorithms for itakura-saito nonnegative matrix factorization," in *Proc. Interspeech 2018*, 2018, pp. 856–860.

[144] Rintaro Ikeshita, "Independent positive semidefinite tensor analysis in blind source separation," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1652–1656.

[145] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio, "Batch-normalized joint training for DNN-based distant speech recognition," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 28–34.

[146] Andrew J R Simpson, "Probabilistic binary-mask cocktail-party source separation in a convolutional deep neural network," *arXiv preprint arXiv:1503.06962*, 2015.

[147] Arpita Gang, Pravesh Biyani, and Akshay Soni, "Towards automated single channel source separation using neural networks," in *Proc. Interspeech 2018*, 2018, pp. 3494–3498.

[148] Yang Sun, Wenwu Wang, Jonathon Chambers, and Syed Naqvi, "Enhanced time-frequency masking by using neural networks for monaural source separation in reverberant room environments," 09 2018, pp. 1647–1651.

[149] Andrew Varga and Herman JM Steeneken, "Assessment for automatic speech recognition: Ii. noisex-92: A database and an experiment to study the effect of additive noise on speech recognition systems," *Speech communication*, vol. 12, no. 3, pp. 247–251, 1993.

[150] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu, "Visual tracking with fully convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3119–3127.

[151] Mohammad Havaei, Nicolas Guizard, Nicolas Chapados, and Yoshua Bengio, "HeMIS: Hetero-Modal Image Segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2016, pp. 469–477.

[152] Klaus Greff, Antti Rasmus, Mathias Berglund, Tele Hao, Harri Valpola, and Juergen Schmidhuber, "Tagger: Deep unsupervised perceptual grouping," in *Advances in Neural Information Processing Systems*, 2016, pp. 4484–4492.

[153] Christian J Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf, "Learning to deblur," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 7, pp. 1439–1451, 2016.

[154] Keunwoo Choi, György Fazekas, and Mark Sandler, "Explaining deep convolutional neural networks on music classification," .

[155] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR abs/1609.03499*, 2016.

[156] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu, "Conditional Image Generation with PixelCNN Decoders," *arXiv preprint arXiv:1606.05328*, 2016.

[157] Weipeng He, Petr Motlicek, and Jean-Marc Odobez, "Joint localization and classification of multiple sound sources using a multi-task neural network," in *Proc. Interspeech 2018*, 2018, pp. 312–316.

[158] Felix Weninger, John R Hershey, Jonathan Le Roux, and Björn Schuller, "Discriminatively trained recurrent neural networks for single-channel speech separation," in *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*. IEEE, 2014, pp. 577–581.

[159] Chenglin Xu, Wei Rao, Eng Siong Chng, and Haizhou Li, "A shifted delta coefficient objective for monaural speech separation using multi-task learning," in *Proc. Interspeech 2018*, 2018, pp. 3479–3483.

[160] Ryo Aihara, Gordon Wichern, and Jonathan Le Roux, "Deep clustering-based single-channel speech separation and recent advances," *Acoustical Science and Technology*, vol. 41, no. 2, pp. 465–471, 2020.

[161] Laxmi Pandey, Anurendra Kumar, and Vinay Namboodiri, "Monoaural audio source separation using variational autoencoders," in *Proc. Interspeech 2018*, 2018, pp. 3489–3493.

[162] Lianwu Chen, Meng Yu, Yanmin Qian, Dan Su, and Dong Yu, "Permutation invariant training of generative adversarial network for monaural speech separation," in *Proc. Interspeech 2018*, 2018, pp. 302–306.

[163] Naoya Takahashi, Purvi Agrawal, Nabarun Goswami, and Yuki Mitsufuji, "Phasenet: Discretized phase modeling with deep neural networks for audio source separation," in *Proc. Interspeech 2018*, 2018, pp. 2713–2717.

[164] Liwen Zhang, Ziqiang Shi, Jiqing Han, Anyan Shi, and Ding Ma, "Furcanext: End-to-end monaural speech separation with dynamic gated dilated temporal convolutional networks," in *International Conference on Multimedia Modeling*. Springer, 2020, pp. 653–665.

[165] Enea Ceolini, Jithendar Anumula, Adrian Huber, Ilya Kiselev, and Shih-Chii Liu, "Speaker activity detection and minimum variance beamforming for source separation," in *Proc. Interspeech 2018*, 2018, pp. 836–840.

[166] Nobutaka Ito, Christopher Schymura, Shoko Araki, and Tomohiro Nakatani, "Noisy cgmm: Complex gaussian mixture model with non-sparse noise model for joint source separation and denoising," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1662–1666.

[167] Kazuhiro Nakadai, Ken-ichi Hidai, Hiroshi G Okuno, and Hiroaki Kitano, "Real-time speaker localization and speech separation by audio-visual integration," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*. IEEE, 2002, vol. 1, pp. 1043–1049.

[168] Fakheredine Keyrouz, Werner Maier, and Klaus Diepold, "Robotic binaural localization and separation of more than two concurrent sound sources," in

2007 9th International Symposium on Signal Processing and Its Applications. IEEE, 2007, pp. 1–4.

[169] Kazuyoshi Yoshii, Koichi Kitamura, Yoshiaki Bando, Eita Nakamura, and Tatsuya Kawahara, "Independent low-rank tensor analysis for audio source separation," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1657–1661.

[170] Shinichi Mogami, Hayato Sumino, Daichi Kitamura, Norihiro Takamune, Shinnosuke Takamichi, Hiroshi Saruwatari, and Nobutaka Ono, "Independent deeply learned matrix analysis for multichannel audio source separation," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1557–1561.

[171] Aditya Arie Nugraha, Antoine Liutkus, and Emmanuel Vincent, "Multichannel audio source separation with deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 9, pp. 1652–1664, 2016.

[172] Lu Yin, Ziteng Wang, Risheng Xia, Junfeng Li, and Yonghong Yan, "Multi-talker speech separation based on permutation invariant training and beamforming," in *Proc. Interspeech 2018*, 2018, pp. 851–855.

[173] Zhong-Qiu Wang and DeLiang Wang, "Integrating spectral and spatial features for multi-channel speaker separation," in *Proc. Interspeech 2018*, 2018, pp. 2718–2722.

[174] Ke Tan and DeLiang Wang, "A two-stage approach to noisy cochannel speech separation with gated residual networks," in *Proc. Interspeech 2018*, 2018, pp. 3484–3488.

[175] Emad M Grais, Dominic Ward, and Mark D Plumbley, "Raw multi-channel audio source separation using multi-resolution convolutional auto-encoders," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1577–1581.

[176] Eugene Weinstein, Kenneth Steele, Anant Agarwal, and James Glass, "LOUD : A 1020-Node Microphone Array and Acoustic Beamformer," *International Congress on Sound and Vibration*, 2007.

[177] Boaz Rafaely, "Phase-mode versus delay-and-sum spherical microphone array processing," *IEEE signal processing Letters*, vol. 12, no. 10, pp. 713–716, 2005.

[178] Ronald Mucci, "A comparison of efficient beamforming algorithms," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 3, pp. 548–558, 1984.

[179] Henry Cox, Robertm Zeskind, and Markm Owen, "Robust adaptive beamforming," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 10, pp. 1365–1376, 1987.

[180] Pieter Sijtsma, "Clean based on spatial source coherence," *International journal of aeroacoustics*, vol. 6, no. 4, pp. 357–374, 2007.

[181] Matthew Aldeman and Ganesh Raman, "Effects of array scaling and advanced beamforming algorithms on the angular resolution of microphone array systems," *Applied Acoustics*, vol. 132, pp. 58–81, 2018.

[182] ," Audacity® software is copyright ©1999-2018 Audacity Team. The name Audacity® is a registered trademark of Dominic Mazzoni.

[183] Robin Scheibler, Eric Bezzam, and Ivan Dokmanić, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 351–355.

[184] Colin Raffel, Brian McFee, Eric J Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel PW Ellis, "mir_eval: A transparent implementation of common mir metrics," in *In Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR*. Citeseer, 2014.

[185] Lukas Drude, Jahn Heymann, Christoph Boeddeker, and Reinhold Haeb-Umbach, "NARA-WPE: A python package for weighted prediction error dere-

verberation in Numpy and Tensorflow for online and offline processing," in *13. ITG Fachtagung Sprachkommunikation (ITG 2018)*, Oct 2018.

[186] Cédric Févotte, Emmanuel Vincent, and Alexey Ozerov, "Single-channel audio source separation with nmf: divergences, constraints and algorithms," in *Audio Source Separation*, pp. 1–24. Springer, 2018.

[187] Guan-Xiang Wang, Chung-Chien Hsu, and Jen-Tzung Chien, "Discriminative deep recurrent neural networks for monaural speech separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2016.* IEEE, 2016, pp. 2544–2548.

[188] Yang Yu, Wenwu Wang, Jian Luo, and Pengming Feng, "Localization based stereo speech separation using deep networks," in *IEEE International Conference on Digital Signal Processing (DSP), 2015.* IEEE, 2015, pp. 153–157.

[189] Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis, "Deep learning for monaural speech separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014.* IEEE, 2014, pp. 1562–1566.

[190] Cédric Févotte, Rémi Gribonval, and Emmanuel Vincent, "Bss_eval toolbox user guide–revision 2.0," 2005.

[191] Antoine Liutkus, Derry Fitzgerald, and Roland Badeau, "Cauchy nonnegative matrix factorization," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA).* IEEE, 2015, pp. 1–5.