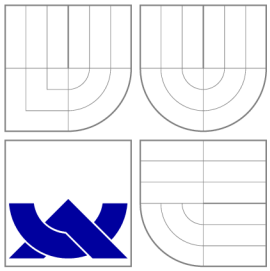
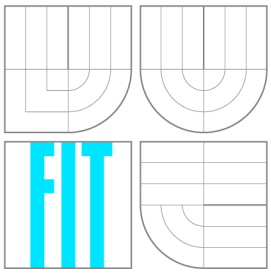


BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ



FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FAR-FIELD SPEECH RECOGNITION

FAR-FIELD SPEECH RECOGNITION

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR
AUTOR PRÁCE

KATEŘINA ŽMOLÍKOVÁ

SUPERVISOR
VEDOUCÍ PRÁCE

Doc. Dr. Ing. JAN ČERNOCKÝ

BRNO 2016

Abstract

The accuracy of speech recognition systems today is very high. However, when speech is captured by a far-field microphone, it can be severly distorted by noise and reverberation and the performance of speech recognition degrades significantly. One way to alleviate this problem is to use microphone arrays. This thesis addresses the methods of combining signals from multiple microphones to improve the quality of the signal and final speech recognition accuracy. It summarizes the theory of speech recognition and the most popular techniques for array processing. Afterwards, it demonstrates and analyzes the results obtained by two different methods for beamforming and a method for dereverberation of multichannel signals. Finally, it examines an alternative way of performing beamforming using neural networks.

Abstrakt

Systémy rozpoznávání řeči v dnešní době dosahují poměrně vysoké úspěšnosti. V případě řeči, která je snímána vzdáleným mikrofonem a je tak narušena množstvím šumu a dozvukem (reverberací), je ale přesnost rozpoznávání značně zhoršena. Tento problém je možné zmírnit využitím mikrofonních polí. Tato práce se zabývá technikami, které umožňují kombinovat signály z více mikrofonů tak, aby byla zlepšena kvalita výsledného signálu a tedy i přesnost rozpoznávání. Práce nejprve shrnuje teorii rozpoznávání řeči a uvádí nejpoužívanější algoritmy pro zpracování mikrofonních polí. Následně jsou demonstrovány a analyzovány výsledky použití dvou metod pro beamforming a metody dereverberace vícekanálových signálů. Na závěr je vyzkoušen alternativní způsob beamformingu za použití neuronových sítí.

Keywords

speech recognition, microphone arrays, beamforming, dereverberation

Klíčová slova

rozpoznávání řeči, mikrofonní pole, beamforming, dereverberace

Reference

ŽMOLÍKOVÁ, Kateřina. *Far-field speech recognition.* Brno, 2016. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Černocký Jan.

Far-field speech recognition

Declaration

I declare that this thesis is my own work supervised by Doc. Dr. Ing. Jan Černocký. All sources have been acknowledged by references.

.....
Kateřina Žmolíková
May 25, 2016

Acknowledgements

I would like to thank my supervisor Honza Černocký for his guidance, positive attitude and lots of valuable suggestions throughout the work on this thesis. I would also like to thank all members of Speech@FIT group for creating great work environment and sharing their knowledge.

Contents

Chapter 1

Introduction

Automatic speech recognition is a research field aiming at automatically translating spoken language into text. Through last fifty years, it has evolved from recognizing small vocabulary of carefully pronounced words to recognizing whole languages and spontaneous speech. Nowadays, the accuracy of the state-of-the-art systems is already sufficient for being used in many applications such as intelligent personal assistants or in-car systems.

However, the translation of speech technologies into real world applications gives rise to many problems which were not present in small artificial tasks. In real tasks it is often more convenient to use far-field rather than close talking microphone. This introduces lots of noise and reverberation which cause significant performance degradation. In such far-field setting, possible solution is to use microphone array rather than single microphone to alleviate the problem.

The usage of multiple microphones enables to use spatial information during the speech pre-processing which can significantly help to separate the speech signal from surrounding noise. The most commonly used class of methods for combining signals from multiple microphones is beamforming whose goal is to artificially steer the microphone array to particular direction.

This thesis will address the problem of far-field speech recognition using microphone arrays. The goal is to implement and experiment with common signal processing techniques to deal with microphone arrays, denoising and dereverberation and possibly to suggest ways for improving them.

In Chapter 2, we will provide basic overview of how speech recognition works. Chapters 3 and 4 will sum up existing beamforming and dereverberation methods and their principles. In Chapter 5, we will describe the datasets which we are using for experiments. Chapter 6 follows with the baseline results on these datasets. Chapters 7 and 8 will show and analyze the results of experiments with two beamforming methods and a dereverberation method. Finally, Chapter 9 will examine alternative way of performing beamforming using neural networks.

Chapter 2

Automatic speech recognition

The objective of speech recognition system is to convert the input speech signal into corresponding sequence of phonemes or words. The main obstacle of this task is a huge variability in speech which can be caused by speaker characteristics, speech style or environmental noise. A successful speech recognition system must be able to deal with all this variability. This is achieved by training flexible statistical models reflecting different aspects of speech.

Typical architecture of speech recognition system is shown in Figure 2.1. It is composed of four basic blocks — feature extraction, acoustic model, language model and hypothesis search. This chapter will provide a quick overview of each of these components.

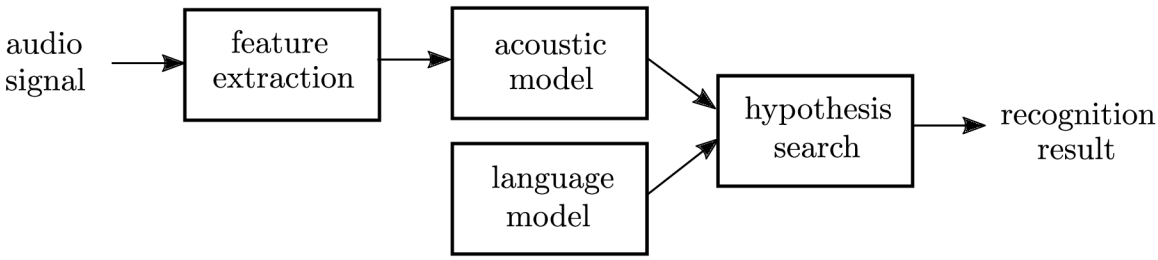


Figure 2.1: Speech recognition system.

2.1 Feature extraction

Feature extraction is the first processing unit of speech recognition system. It performs a transformation of the input speech signal into representation which is more suitable for the rest of the system. The main reasons for doing feature extraction are reduction of dimensionality and removal of information which is irrelevant for the recognition task.

The process of feature extraction is inspired by findings in the field of speech perception. The most popular feature extraction techniques today are Mel-frequency cepstral coefficients (MFCCs) [6] and Perceptual Linear Prediction (PLP) [13]. They both follow similar processing steps, which include segmentation of the input signal, computation of the Fourier spectrum, auditory-like modifications, decorrelation and taking derivatives of final features.

Features are also often transformed by some kind of linear transformation to further increase the dimensionality reduction and robustness. The transformations used in systems in this thesis include Linear Discrimination Analysis (LDA), Maximum Likelihood Linear

Transform (MLLT) [9] or Constrained Maximum Likelihood Linear Regression (CMLLR) [8].

Although mentioned techniques are the standard appearing in most of the modern systems, alternative ways to perform feature extraction have been proposed. Recent trend heads towards replacing feature extraction by neural networks which are trained with classification objective [11] [26].

2.2 Acoustic modeling

The acoustic model incorporates knowledge about acoustics and phonetics. It represents a mapping between sequences of feature vectors and sequences of phones. To deal with variable length of feature vector sequences, it is usually based on Hidden Markov Models (HMM) [10]. In HMM-based large vocabulary systems, each phone is represented by an HMM, which models the process of generating the sequence of feature vectors corresponding to this phone. This section will briefly explain basic ideas of Hidden Markov Models as well as neural networks which are usually used in combination with HMMs.

2.2.1 Hidden Markov Models

Hidden Markov Model is a generative model which we can see as a finite-state automaton generating vectors from state-dependent probability distributions. It can be characterized by its topology, transition probabilities and output probability distributions associated with each state. The topology (number of states and possible transitions) is usually expertly chosen — for speech recognition, each phone is represented by HMM with three states and left-to-right topology as depicted in Figure 2.2. Transition probabilities and parameters of output probability distributions are estimated from training data.

When we want to evaluate the likelihood that a sequence of feature vectors $\mathbf{O}$ of length T was generated by given model M, we need to sum over all possible state sequences $\mathbf{S}$ of length T.

$$p(\mathbf{O}|M) = \sum_{\mathbf{S}} p(\mathbf{O}, \mathbf{S}|M) = \sum_{\mathbf{S}} p(\mathbf{O}|\mathbf{S}, M) P(\mathbf{S}|M). \tag{2.1}$$

The likelihood of observation sequence given a state sequence and a model $p(\mathbf{O}|\mathbf{S}, M)$ can be computed as

$$p(\mathbf{O}|\mathbf{S}, M) = \prod_{t=1}^{T} b_{s(t)}(\mathbf{o}(t)), \tag{2.2}$$

where b_j denotes output probability distribution associated with state j and $\mathbf{o}(t)$ is the t-th vector of feature vectors $\mathbf{O}$. The probability of state sequence given a model $P(\mathbf{S}|M)$ is

$$P(\mathbf{S}|M) = \prod_{t=2}^{T} a_{s(t-1)s(t)}, \tag{2.3}$$

where $a_{s_i s_j}$ denote transition probability between states s_i and s_j and $s(t)$ is the t-th element of the state sequence.

Since summing over all possible state sequences is an expensive operation, the evaluation is often approximated by the likelihood of the best state sequence

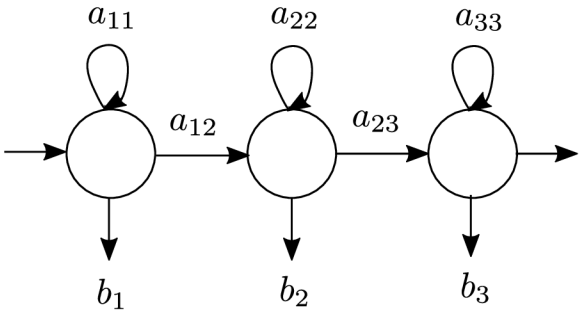


Figure 2.2: Typical structure of Hidden Markov Model representing one phoneme — left-to-right topology with three states.

$$\hat{p}(\mathbf{O}|M) = \max_{\mathbf{S}}\{p(\mathbf{O}|\mathbf{S}, M)P(\mathbf{S}|M)\}. \tag{2.4}$$

The output probability distributions of HMM in acoustic models can be modeled by Gaussian mixture models (GMM), which was typically done in the past. However today, systems using neural networks have been shown to outperform GMMs [14]. GMMs are therefore usually used only in first stages of building an ASR system and the final stage incorporates a neural network to estimate the output probabilities.

2.2.2 Neural Networks

Feed forward neural network (also known as multilayer perceptron) is a mathematical model, which represents a transformation of its input to its output. It is basically a parametric function which is composed from multiple layers of computations. Each layer is connected to the previous one through weight matrix and computes its output as

$$y_i^L = f(\sum_j w_{ij}^{L,L-1} y_j^{L-1}), \tag{2.5}$$

where y_i^L is the output of unit i in layer L, $w_{ij}^{L,L-1}$ is an element of the weight matrix between layers $L-1$ and L and f is an activation function which is typically a logistic sigmoid. Figure 2.3 shows the computational structure of a small neural network.

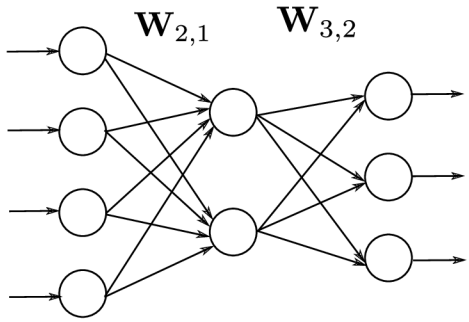


Figure 2.3: Neural network with two layers characterized by weight matrices $\mathbf{W}_{2,1}$ and $\mathbf{W}_{3,2}$.

The weight matrices at each layer form the parameters of the network which can be learnt. This way the network can be trained to perform specific task. The training procedure aims to minimize an error function computed from the output of the network with respect to the training data. The most popular training algorithm is back-propagation which is derived from chain rule used for gradient computation. More information about neural networks and their training can be found in [3].

In speech recognition systems, neural networks are used to estimate posterior probabilities of states of Hidden Markov Models given the acoustic observations. This means it performs a multi-class classification task with sequences of feature vectors as input and HMM states corresponding to these feature vectors as target classes.

2.3 Language modeling

The language model incorporates knowledge about language or potentially about the domain of the task. Its goal is to estimate the probability of word sequences which is done by learning correlations between words in training corpora. Most popular language models are n-gram models, although they are being replaced by recurrent neural networks. Since this thesis does not focus on language modeling, more detailed description of language model will be omitted.

2.4 Hypothesis search

The previous sections summarize three components of speech recognition systems

feature extraction used to obtain sequence of feature vectors $\mathbf{O}$ from input speech signal,

acoustic model providing the likelihood of feature sequence given a sequence of words $p(\mathbf{O}|\mathbf{W})$,

language model providing the prior probability of sequence of words $P(\mathbf{W})$.

Using these, the overall task of speech recognition can be viewed as finding the most likely sequence of words given observed sequence of feature vectors:

$$\mathbf{W} = \arg\max_{\mathbf{W}'} P(\mathbf{W}'|\mathbf{O}) = \arg\max_{\mathbf{W}'} p(\mathbf{O}|\mathbf{W}')P(\mathbf{W}'). \tag{2.6}$$

The hypothesis search component therefore just combines the acoustic and language model scores and outputs the word sequence with the highest score as the recognition result. The search of the most likely word sequence is typically done on the level of weighted finite state transducers which can represent both acoustic and language model jointly [21].

Chapter 3

Beamforming

As discussed in the introduction, the speech signal captured by far-field microphone is often disrupted by noise and reverberation which causes significant decrease in speech recognition accuracy. Using microphone arrays instead of single microphone is a possible way to facilitate the task of restoring the clean speech signal. The most popular class of methods for processing microphone arrays is beamforming. The name 'beamforming' refers to purpose of the methods of artificially directing the array towards the desired sound source — forming a beam in that direction. The directing of the array is achieved by suitable combination of the signals received at individual microphones.

The main concept of beamforming methods makes use of the fact that the signals coming from different directions arrive at each of the microphones with different delay and attenuation. Having multiple recordings from different spatial positions therefore gives us information about the direction that each of the sources comes from. The beamformer is then able to spatially select only the signal of interest while suppressing the others.

In this chapter, we will first define a multichannel signal model and general framework of beamforming methods. Then two most widely used methods will be presented — delay-and-sum and minimum variance distortionless response (MVDR) beamforming. Since both of these methods are estimating their parameters based on the delays with which speech signal arrives at microphones, methods for time delay estimation will be also overviewed.

3.1 Multichannel signal model

This section will introduce assumed model of signals arriving at microphones and used notation. We will follow notation and assumptions introduced by Souden [30].

The source speech signal $s(t)$ is received by an array of N microphones in the following form

$$y_n(t) = g_n(t) * s(t) + v_n(t) = x_n(t) + v_n(t), \qquad n = 1, 2, ..., N \tag{3.1}$$

where $*$ is convolution operator, g_n is the channel impulse response affecting the signal on n-th microphone, $x_n(t)$ is reverberant speech component and $v_n(t)$ is noise component on n-th microphone. We assume that all noise components are uncorrelated with $s(t)$ and that both noise components and speech signal are zero-mean processes. Figure 3.1 shows the schema of the assumed model.

As beamforming is typically defined in frequency domain, we also introduce frequency-domain counterpart of above model

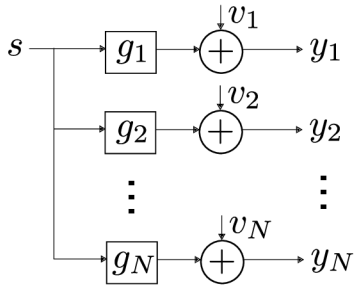


Figure 3.1: The model of the generation of the signals received by each microphone. N denotes the number of microphones, g_n is the impulse response affecting signal on n-th microphone, v_n is the noise component and y_n is the received signal by nth microphone.

$$Y_n(j\omega) = G_n(j\omega)S(j\omega) + V_n(j\omega) = X_n(j\omega) + V_n(j\omega), \qquad n = 1, 2, ..., N \tag{3.2}$$

where $Y_n(j\omega), G_n(j\omega), S(j\omega), V_n(j\omega)$ are discrete-time Fourier transforms of $y_n(t), g_n(t), s(t)$ and $v_n(t)$.

The goal of the beamforming procedure is to recover speech component at one of the microphones $X_{n_0}(j\omega)$ ($n_0 \in \{1, ..., N\}$ will be referred to as reference microphone). This will be done by applying linear filter $\mathbf{h}_{n_0}(j\omega)$ to the received signals. The output of beamforming will then for each normalized angular frequency ω be

$$Z(j\omega) = \mathbf{h}_{n_0}^H(j\omega)\mathbf{y}(j\omega), \tag{3.3}$$

where ^{H} denotes transpose-conjugate operator and $\mathbf{y}(j\omega)$ is vector of frequency-domain observations at each microphone

$$\mathbf{y}(j\omega) = [Y_1(j\omega) \;\; Y_2(j\omega) \;\; ... \;\; Y_N(j\omega)]^T. \tag{3.4}$$

3.2 Delay-and-sum beamformer

Delay-and-sum (DS) is a simple and straightforward method to perform beamforming. It uses the fact that microphones situated at different spatial positions receive the same source signal with different delays. Moreover, the delays vary with the direction from which the source signal is coming. If we know the delays corresponding to the desired direction, we can use them to shift the signals. The shift operation aligns the desired signal in all channels, while the signals coming from different directions remain unaligned. We can then simply average all such shifted signals which will cause attenuation of the signals from unwanted directions. Figure 3.2 illustrates the idea — the delays τ_n are computed to align the desired speech signal, which causes the attenuation of noise coming from different direction.

To fit this process into the general beamforming formula, we define a steering vector

$$\mathbf{d}(\omega) = [e^{-i\omega\tau_1} \; e^{-i\omega\tau_2} \; ... \; e^{-i\omega\tau_N}], \tag{3.5}$$

where τ_i is the delay of the speech signal at microphone i with respect to the reference microphone. Using this vector, the delay-and-sum filter can be defined as

$$\mathbf{h}_{n_0}^{DS}(j\omega) = \frac{1}{N}\mathbf{d}(\omega). \tag{3.6}$$

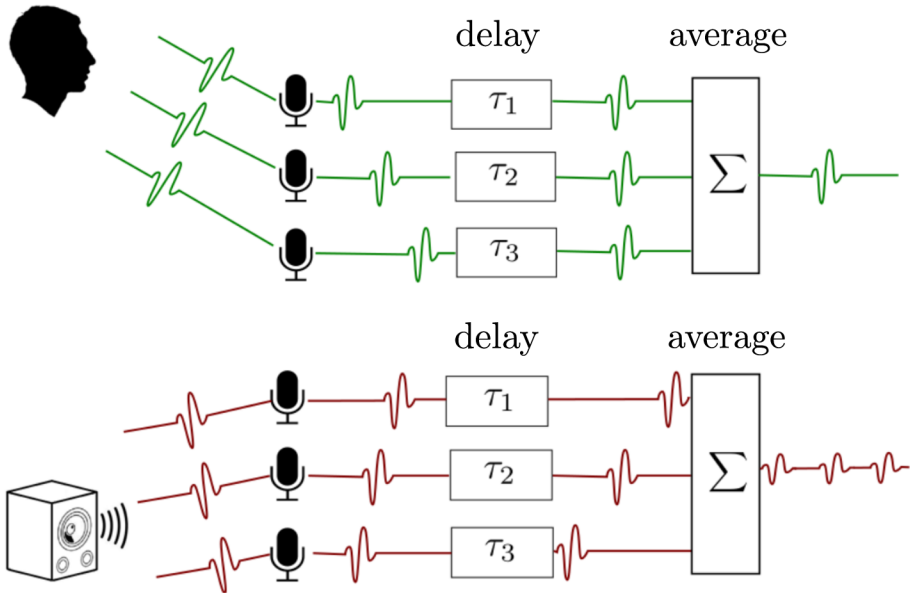


Figure 3.2: Illustration of the idea of the Delay-and-Sum beamforming. Delays τ_n are computed to align the desired speech signal, which causes the attenuation of noise coming from different direction.

If the source signal was not disrupted by any interference, the output of delay-and-sum beamformer would be equivalent to the original signal. Thus we say that the filter satisfies distortionless constraint — the signal coming from look direction is neither amplified nor attenuated.

3.3 Time delay estimation

To use the delay-and-sum beamformer, we need to know the delays of the source signal as received by each microphone. This can be either known apriori (from the microphone array architecture and position of the source) or, in more common case, we need to estimate the delays from the received signals.

The most popular techniques for time delay estimation are based on cross correlation of the signals [5]. For two signals x_0, x_1, the cross correlation is defined as

$$\Psi^{CC}_{x_0,x_1}[m] = E\{x_0[l]x_1[l+m]\}. \tag{3.7}$$

The peak of the cross correlation should occur at the time of the delay. Thus, we can then estimate the delay as

$$\overset{\wedge}{\tau}{}^{CC}_{x_0,x_1} = \arg\max_m \Psi^{CC}_{x_0,x_1}[m]. \tag{3.8}$$

Assuming ideal conditions where the two signals are just delayed versions of each other ($x_1[n] = x_0[n-D]$), the cross correlation function reduces to

$$\Psi^{CC}_{x_0,x_1} = \Psi^{CC}_{x_0,x_0} \circledast \delta[n-D]. \tag{3.9}$$

As the maximum of auto-correlation function $\Psi^{CC}_{x_0,x_0}[m]$ is at $m = 0$, the peak of $\Psi^{CC}_{x_0,x_1}[m]$ occurs indeed at the time of the delay $m = D$. However, the peak is smeared by the

auto-correlation function of the signal x_0. When factors as noise and reverberation take place, this may lead to inaccuracies in the time delay estimate. To deal with this problem, Knapp and Carter [19] proposed improved version of cross correlation called Generalized cross-correlation (GCC) implementing a frequency domain weighting. This can make the peak more sharp and the estimate more robust. The general formula for computing GCC is

$$\Psi_{x_0,x_1}^{GCC}[m] = \sum_{k'=0}^{K'-1} \Phi[k'] S_{x0x1}[k'] e^{j2\pi mk'/K'}, \tag{3.10}$$

where $S_{x0x1}[k'] = E\{X_0[k']X_1^*[k']\}$ is the cross-spectrum, $X_0[k']$ is the discrete Fourier transform of $x_n[k]$, K' is the length of the DFT and $\Phi[k']$ is a weighting function. The time delay estimate then can be again obtained as

$$\overset{\wedge}{\tau}_{x_0,x_1}^{GCC} = \arg\max_m \Psi_{x_0,x_1}^{GCC}[m]. \tag{3.11}$$

Lots of choices of the weighting function $\Phi[k']$ have been proposed. In the case of constant weighting, the GCC becomes equivalent to simple cross-correlation. Table 3.1 summarizes other alternatives.

CC	1
PHAT	$\frac{1}{\lvert S_{x_0x_1}[k'] \rvert}$, where $S_{x_0x_1} \approx X_0X_1^*$
Roth	$\frac{1}{S_{x_0x_0}[k']}$, where $S_{x_0x_1} \approx X_0X_0^*$

Table 3.1: Weighting functions for generalized cross correlation.

The Roth weighting was proposed in [25] and according to Knapp [19] it should suppress the frequency regions where the noise in signal x_0 is large. The PHAT (PHase Transform) weighting is an ad-hoc technique which in effect normalizes the magnitudes of the spectra of both signals x_0 and x_1 and uses just the phase which should contain the information about the delay of the signals. Since it uses the phase information from all the frequency bins equally, it should perform well when the speech occupies most of the frequency bins. Figure 3.3 shows an example of a short segment of speech, its delayed version, and the cross-correlation function between the two, using constant, Roth and PHAT weighting. It is visible, that in this simple case, the Roth and PHAT weighting lead to much sharper peak than simple cross-correlation.

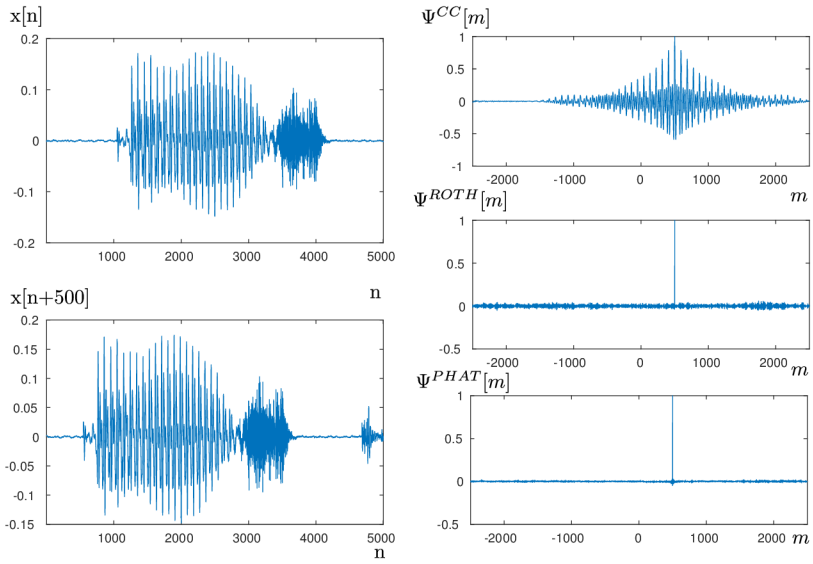


Figure 3.3: Example of a delayed signal and its cross-correlation using CC (constant), ROTH and PHAT weighting functions.

3.4 Minimum variance distortionless response beamformer

The Delay-and-sum beamformer is, for its simplicity, a very common choice for beamforming and often performs sufficiently well. However, the main drawback of the DS method is, that it estimates its parameters considering only the position of the desired source and not the positions of the interfering sound sources. This gives rise to Minimum variance distortionless response beamformer (MVDR) [20] [30] which explicitly aims to minimize the effect of the noise.

To achieve the maximum noise reduction, it uses an estimate of noise covariance matrix which represents how correlated the noise signals are between the microphones. The knowledge of these correlations provides the information about the directions of the noise sources and enables the beamformer to suppress the signals coming from these directions.

In order to derive a filter with these optimal properties, we define two performance measures:

noise reduction factor

$$\xi_{nr}[\mathbf{h}_{n_0}(j\omega)] = \frac{E\{|V_{n_0}(j\omega)|^2\}}{E\{|\mathbf{h}_{n_0}^H(j\omega)\mathbf{v}(j\omega)|^2\}}, \tag{3.12}$$

where $\mathbf{h}_{n_0}(j\omega)$ is the filter, n_0 denotes the reference microphone, $V_{n_0}(j\omega)$ is the frequency-domain noise signal as received by the reference microphone, $\mathbf{v}(j\omega)$ is the vector of frequency-domain noise signals received by each microphone $\mathbf{v}(j\omega) = [V_1(j\omega)\ \ V_2(j\omega)\ \ \ldots\ \ V_N(j\omega)]^T$ and superscript ^{H} denotes conjugate-transpose. The noise reduction factor represents the amount of noise which was suppressed by the beamforming procedure. Its optimal value would be ∞.

speech distortion factor

$$v_{sd}[\mathbf{h}_{n_0}(j\omega)] = \frac{E\{|X_{n_0}(j\omega) - \mathbf{h}_{n_0}^H(j\omega)\mathbf{x}(j\omega)|^2\}}{E\{|X_{n_0}(j\omega)|^2\}}, \tag{3.13}$$

where $X_{n_0}(j\omega)$ is the frequency-domain clean speech signal as received by the reference microphone, $\mathbf{x}(j\omega)$ is the vector of frequency-domain clean speech signals received by each microphone $\mathbf{x}(j\omega) = [X_1(j\omega)\ \ X_2(j\omega)\ \ ...\ \ X_N(j\omega)]^T$ and rest of the symbols have the same meaning as above. The speech distortion factor represents the amount of speech suppressed by the beamforming procedure. Its optimal value would be 0.

Using noise reduction and speech distortion, we can define an optimization problem resulting in optimal filter — filter which removes as much noise as possible while keeping the speech signal undistorted.

$$\mathbf{h}_{n_0}^{MVDR}(j\omega) = \arg\max_{\mathbf{h}_{n_0}(j\omega)} \xi_{nr}[\mathbf{h}_{n_0}(j\omega)] \tag{3.14}$$

$$\text{subject to} \quad v_{sd}[\mathbf{h}_{n_0}(j\omega)] = 0 \tag{3.15}$$

The weights obtained by this optimization correspond to MVDR filter which has the solution

$$\mathbf{h}_{n_0}^{MVDR}(j\omega) = \frac{\mathbf{d}^H(\omega)\mathbf{\Sigma}_N^{-1}(\omega)}{\mathbf{d}^H(\omega)\mathbf{\Sigma}_N^{-1}(\omega)\mathbf{d}(\omega)}, \tag{3.16}$$

where $\mathbf{d}(\omega)$ is the steering vector defined earlier and $\mathbf{\Sigma}_N(\omega) = E[\mathbf{v}(j\omega)\mathbf{v}^H(j\omega)]$ is the noise covariance matrix, which is usually estimated from silence parts of speech recordings. The derivation of 3.16 is described in detail in Appendix A.

Chapter 4

Dereverberation

The speech signal captured by distant microphones contains not only the additive noise, but also reflections of the signal from walls and other objects — effect known as reverberation. Reverberation causes the microphone to receive multiple copies of the original signal with different delays and attenuation. Due to this effect, the resulting speech signal is less intelligible and the accuracy of speech recognition systems drops substantially.

This degradation is difficult to solve due to the distinctive properties of reverberation. In [12], Habets points out that traditional beamforming algorithms become ineffective in the presence of reverberation. As a consequence, it is necessary to use special methods to dereverberate the input signal. There are multiple classes of approaches which are designed to defeat the reverberation problem. In this work, we focused on Weighted prediction error method (WPE) as it has been shown successful in recent evaluations [16][35], it can effectively make use of multichannel signals and it can be easily coupled with beamforming techniques.

In this chapter, we first describe the properties of reverberation, review existing methods for dereverberation and finally focus on the method applied in this work — Weighted prediction error.

4.1 Properties of reverberation

Reverberation is the effect of propagation of the speech signal in an enclosed space. On the way from the speaker to the microphones it is propagated not only directly, but it also reflects on walls and other objects. The microphone then receives a mixture of the reflected signals which differ by their delay (due to different lengths of propagation paths) and attenuation (due to the walls absorbing different amounts of the signal). This has the effect of a smearing in time which causes masking of subsequent phonemes. Figure 4.1 shows a comparison of clean and reverberant speech. We can observe how the reverberation corrupts the transitions between the phonemes and makes them less distinct. Perceptually, these effects lead to speech sounding “distant” and “echoic”.

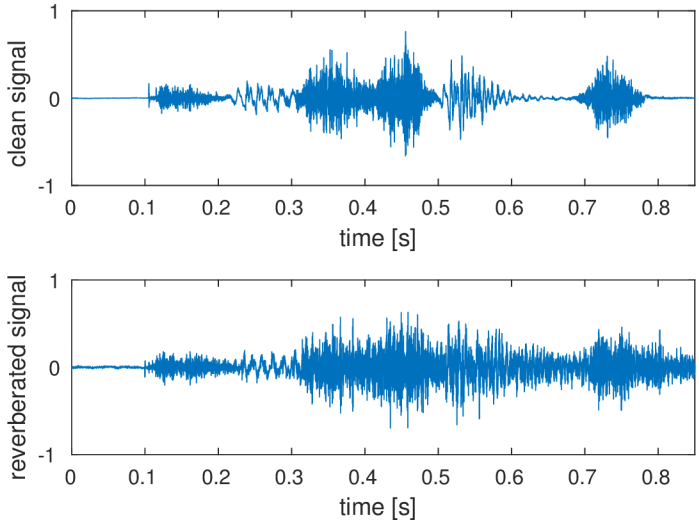


Figure 4.1: Comparison of clean and reverberated speech.

The sum of repeated delayed reflections of the original signal, which the microphone receives, can be mathematically described as a convolution of the original signal with room impulse response (RIR)

$$y(t) = s(t) \star h(t) + d(t), \tag{4.1}$$

where $s(t)$ is the original signal, $h(t)$ is the room impulse response, $d(t)$ is additive noise and $y(t)$ is the distorted signal received by the microphone. The room impulse response $h(t)$ describes the response of the room to an impulsive sound. According to [38], it can be divided into three parts corresponding to three components of the reverberation

- **direct sound** This part of reverberant speech is the signal received through the shortest path from source to the microphone.
- **early reflections** Early reflections describe the first few copies of the signal received by the microphone within the first 50 milliseconds after the direct sound. This part of reverberation has been shown to improve the intelligibility of the speech. Also in speech recognition systems, the early reflections are not causing much degradation as their effect is short-term and thus can be suppressed by conventional methods as cepstral mean normalization.
- **late reverberation** The late reverberation describes the mixture of high amount of small reflections received by the microphone after the first 50 milliseconds after the direct signal. This part of reverberation is causing the degradation in both intelligibility and accuracy of the speech recognition. Most of the dereverberation methods are focused on compensating of this component.

The main problem with the reverberated speech which causes conventional noise reduction techniques to be inefficient is its high non-stationarity. Since the original speech signal is non-stationary and the reverberation is composed of copies of this speech signal, its characteristics also change very rapidly. The widely used methods of noise robust speech

processing are usually based on the assumption of stationary noise and thus cannot reduce the reverberation effectively.

Another property of reverberation with great impact on the speech recognition are the long-term dependencies which are present in the reverberated signal. The effect of reverberation is usually longer than the length of a frame used for feature extraction, which causes dependency between subsequent feature vectors. This contradicts the assumption of Hidden Markov Models in the acoustic model, that the feature vector depends only on the current state and not on the previous feature vectors. As a result, the traditional acoustic model is not very effective for modeling of the reverberant speech. Most of the dereverberation methods make use of these long-term dependencies to compensate for the reverberation.

4.2 Methods for dereverberation

The methods which aim to make speech recognizer robust to reverberation can be divided into several categories depending on the stage of the processing where they take place. We will briefly overview these categories and further focus on a specific method used in this work.

According to [38], the approaches can be roughly classified into the following categories

- **linear filtering** Linear filtering methods work in the time or short-time Fourier transform (STFT) domain. As the only one of presented classes of methods, they use the phases of the input signal. They are suitable for combining with multi-microphone approaches.
- **spectrum enhancement** Spectrum enhancement methods dereverberate the corrupted power spectra. They can be easily combined with other noise reduction approaches which often act in the power spectrum domain.
- **feature enhancement** Feature enhancement methods dereverberate the features extracted from the input signal. They can be combined with methods for uncertainty decoding.
- **back-end approaches** Back-end approaches aim to adapt the acoustic model to be robust to reverberant features. This may include well-known HMM adaptation techniques as Maximum A Posteriori (MAP) or Maximum Likelihood Linear Regression (MLLR) adaptation or more advanced methods as REMOS (REverberation MOdeling for Speech recognition).

For more thorough description of the mentioned methods we refer the reader to [38]. In this work, we focused on a method based on linear filtering as it fits the best the presented beamforming framework.

4.3 Weighted prediction error method

Weighted prediction error method (WPE) is based on multichannel linear prediction. It processes the signals in the time-frequency domain and transforms the reverberant speech signals from all the channels to the same number of dereverberated signals. The processing preserves the delay differences which makes the method suitable for preprocessing the signals prior to beamforming. The method was introduced by Yoshioka in [34]. In this section, we

briefly summarize the crucial mathematical foundation of the method — we define the problem, introduce the assumed models and finally summarize the WPE dereverberation procedure.

The dereverberation problem is defined as estimating the clean speech signal s from the observed reverberated signals y_n where n denotes the index of the microphone. The reverberated signal can be described in time-frequency domain as

$$y_{n,t,k} = \sum_{m=0}^{J_k} h_{n,m,k} s_{t-m,k}, \tag{4.2}$$

where h is the narrow-band room impulse response for kth frequency bin, t is the time index and J_k is the order of the room impulse response (RIR). To facilitate the dereverberation problem, we define a model of the clean speech and a model of the reverberation process and then use these models to obtain the estimate of the clean speech.

The speech is modeled with time-varying Gaussian model. This model assumes that each STFT coefficient is sampled independently from complex normal distribution with mean 0 and variance $\lambda_{t,k}$

$$p(s;\Phi) = \prod_{k=0}^{K-1} \prod_{t=0}^{T-1} \frac{1}{\pi \lambda_{t,k}} \exp\left(-\frac{|s_{t,k}|^2}{\lambda_{t,k}}\right), \tag{4.3}$$

where Φ are the parameters of the speech model. The time-varying variance $\lambda_{t,k}$ is represented by an all-pole model which assumes

$$\lambda_{t,k} = \frac{\sigma_t^2}{|A(\exp\left(j\frac{2\pi k}{K}\right))|^2}, \tag{4.4}$$

where A is the frequency response of the linear prediction filter and σ_t^2 is the prediction residual power (PRP). The coefficients of the filter together with the PRP form the speech model parameters Φ.

The reverberation process is modeled by multiple-input single-output auto-regressive model (MISO AR) as

$$y_{n,t,k} = \sum_{m=\delta_k}^{\delta_k+M_k-1} g_{n,m,k}^H y_{t-m,k} + s_{t,k}, \tag{4.5}$$

where $g_{n,m,k}$ is the 1-by-N vector of mth coefficients of the reverberation prediction filter for microphone n and frequency bin k, M_k is the order of the filter, $y_{t,k}$ is the vector of observed signals at all microphones for time t and kth frequency bin $[y_{1,t,k} \ \dots \ y_{N,t,k}]^T$ and δ_k is the prediction delay. Using the prediction delay $\delta_k > 1$ was suggested by Kinoshita in [17] and has been shown to be preventing excessive whitening problem which occurs when classic multichannel linear prediction is used for dereverberation. We denote the set of the reverberation model parameters as Ψ.

The used MISO AR model assumes the presence of only one speaker and no background noise. In [36, 37] Yoshioka presented generalizations of the method which takes into account the situation of multiple sound sources. However, the basic WPE algorithm has been reported to be successful even in the situation, where multiple sound sources are present, despite the fact that theoretically it expects only one sound source. Considering the much lower computational complexity than the extensions, we used the simple WPE in this work.

Combining the speech and reverberation model, we can define the probability density function of the reverberated signal at one of the microphones as

$$p(y;\Phi,\Psi)=\prod_{k=0}^{K-1}\prod_{t=0}^{T-1}\frac{1}{\pi\lambda_{t,k}}\exp\left(-\frac{|y_{1,t,k}-\sum_{m=\delta_k}^{\delta_k+M_k-1}g_{m,k}^{H}y_{t-m,k}|^2}{\lambda_{t,k}}\right). \tag{4.6}$$

To optimize the parameters of both models, we maximize a log-likelihood function which is obtained from 4.6 as

$$\mathcal{L}(\Phi,\Psi)=-\sum_{k=0}^{K-1}\sum_{t=0}^{T-1}\left(\log\lambda_{t,k}+\frac{|y_{1,t,k}-\sum_{m=\delta_k}^{\delta_k+M_k-1}g_{m,k}^{H}y_{t-m,k}|^2}{\lambda_{t,k}}\right). \tag{4.7}$$

Since the log-likelihood function cannot be maximized analytically for both Φ and Ψ, we need to alternate between optimizing Φ using fixed Ψ (optimizing the parameters of the speech model) and optimizing Ψ while fixing Φ (optimizing the reverberation prediction filter).

To optimize the parameters of the speech model, we use the current estimate of the reverberation prediction filter to obtain the estimate of the clean speech as

$$\hat{s}_{t,k}=y_{1,t,k}-\sum_{m=\delta_k}^{\delta_k+M_k-1}g_{1,m,k}^{H}y_{t-m,k} \tag{4.8}$$

and use this estimate to apply linear prediction and update the all-pole model parameters using 4.4.

Optimization of the parameters of the reverberation model is analogous to optimization of classical multichannel linear prediction filter, with the modification of considering the time-varying variance. The modified formula is given by

$$\hat{g}_k=\left(\frac{\sum_{t=0}^{T-1}y_{t-\delta_k,k}y_{t-\delta_k,k}^{H}}{\hat{\lambda}_{t,k}}\right)^{-1}\left(\frac{\sum_{t=0}^{T-1}y_{t-\delta_k,k}y_{t,k}^{*}}{\hat{\lambda}_{t,k}}\right). \tag{4.9}$$

The overview of all the steps of the algorithm is given in Figure 4.2. The method iterates between estimation of the speech properties and the reverberation prediction filter until it reaches convergence.

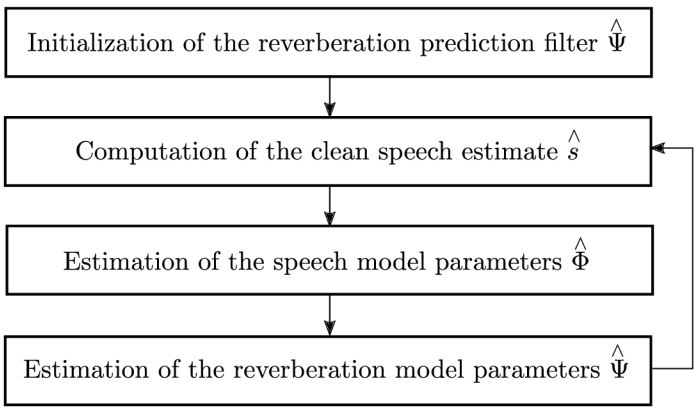


Figure 4.2: The overview of the steps of the Weighted prediction error method.

Chapter 5

Datasets

This chapter will describe the datasets used to evaluate the implemented methods. We used three datasets containing recordings from multiple distant microphones — AMI, CHiME3 and REVERB. Each of the datasets has slightly different properties. The AMI dataset includes recordings of meetings, so it has relatively small amount of background noise, but in contrast with the other sets, it contains conversational speech. On the contrary, CHiME3 dataset was recorded in real-world conditions, so the recordings are very noisy. The last one is REVERB dataset which contains lots of reverberation.

5.1 AMI

The AMI Meeting Corpus [4] is a collection of meetings recorded in three standardized meeting rooms. The recordings were obtained using 12 microphones — a headset microphone per participant and an 8-element circular microphone array. All meetings are in English, though mostly spoken by non-native speakers. The total amount of recorded data is about 100 hours, which is partitioned into trainining, development and evaluation sets following [32]. This makes about 78 hours of speech for training, 9 hours for dev set and 9 hours for eval set.

Figure 5.1: The recording of AMI meeting corpus[1].

[1]Photo from http://www.amiproject.org/ami-scientific-portal.

5.2 CHiME3

The CHiME3 dataset was recorded for the 3rd CHiME Challenge for Speech Separation and Recognition [2]. It includes recordings of people speaking in real environments including cafe, street junction, public transport and pedestrian area. Recordings were obtained using tablets with six channel microphone array. Apart from from the real recordings, part of the data were also obtained by artificially mixing clean speech data with noisy backgrounds.

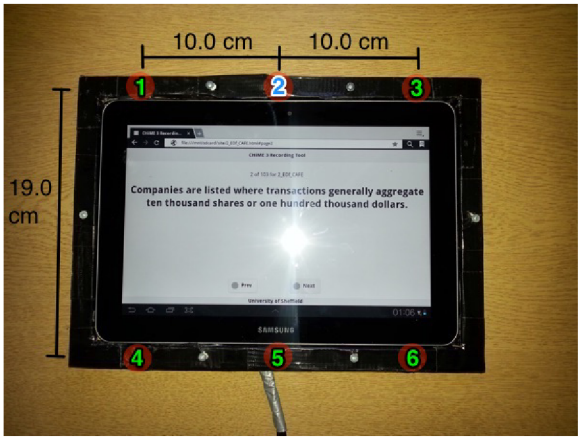


Figure 5.2: The tablet used for recording the CHiME3 corpus[2].

The training set comprises of 1600 utterances from 4 speakers speaking in real environments and 7138 simulated utterances from 83 speakers. In the development set, there are 1640 real and 1640 simulated utterances from 4 speakers (which do not overlap with speakers from training data). The evaluation set contains 1320 real and 1320 simulated utterances from 4 other speakers.

5.3 REVERB

The REVERB dataset was recorded for REVERB challenge [18] aimed at evaluation of speech enhancement and recognition in reverberant environments. The recordings were obtained with distant microphone array in reverberant rooms with a limited amount of stationary noise. For all data, 1 microphone, 2 microphone and 8 microphone versions were available — here we focused on 8 microphone recordings.

The training data were obtained by mixing clean data from WSJCAM0 dataset [24] with room impulse responses and noise signals measured in real rooms. The development set consists of data from 4 rooms. For three of them, the data were simulated, for the fourth room the recordings are real. The evaluation set consists of the same environments as dev set, but with different speakers and different positions in the rooms.

[2]Photo from http://spandh.dcs.shef.ac.uk/chime_challenge.

Chapter 6

Baselines

In this chapter, we describe built speech recognition systems which we used for testing of the implemented methods. We reference the tools we used and present the results achieved on the previously described datasets.

6.1 Tools

BeamformIt

To compare the results of our beamforming implementation with the state-of-the-art, we used BeamformIt toolkit[1] [1] as a baseline. It is a publicly available tool written by Xavier Anguera, initially for the purpose of multichannel speaker diarization of meetings. It implements delay-and-sum beamforming including blind reference channel selection, two-step time delay of arrival Viterbi postprocessing and a dynamic output signal weighting algorithm.

Kaldi

For building the speech recognition systems, we used Kaldi speech recognition toolkit[2] [23]. It is a freely available tool intended for use by researchers. Its core is written in C++, using OpenFst library for working with finite-state transducers and BLAS/LAPACK for linear algebra. One of the main strenghts of Kaldi is number of complete recipes for building speech recognition systems on widely used databases.

6.2 AMI

The baseline system for AMI was built using standard Kaldi recipe. The input features were 13-dimensional MFCCs transformed by LDA and MLLT transforms. The GMM/HMM models were trained in speaker adaptive fashion using CMLLR transform. Final GMM/HMM systems also used discriminative training with boosted Maximum mutual information (bMMI) criterion. The trained models consist of about 4000 tied-states with roughly 20 Gaussians per state. This was used for providing an alignment for building DNN system.

The DNN system was trained on the LDA+MLLT+CMLLR transformed features with +-5 frames context and global mean and variance normalization. The DNN, consisting of 6

[1] BeamformIt toolkit http://www.xavieranguera.com/beamformit

[2] Kaldi speech recognition toolkit http://kaldi-asr.org/

layers of 2048 neurons, is first initialized with generative pretraining using stacked restricted Boltzmann machines (RBM). After this, the network is trained by stochasic gradient descent optimizing cross-entropy and finally trained in sequence discriminative fashion to optimize state Minimum Bayes Risk (sMBR) criterion.

system	1-best channel		BeamformIt	
	dev	eval	dev	eval
GMM	60.5	66.1	57.4	61.9
DNN (CE)	54.7	59.0	48.7	53.1
DNN (sMBR)	50.55	55.12	45.0	49.6

Table 6.1: Baseline results of AMI system. Results show word error rates [%] using data from only single channel (the best one) and using BeamformIt on all 8 channels.

6.3 CHiME3

The CHiME3 baseline system was built using Kaldi recipe which we modified to use CMLLR transformed features on the input of neural networks for better results. The input features and the training procedure of GMM/HMM system is analogous to the AMI system. The trained models consist of about 2000 tied-states with roughly 10 Gaussians per state. The DNN system was initialized using generative pretraining and then finetuned with discriminative training optimizing cross entropy criterion.

system	1-best channel	BeamformIt
GMM	18.20 / 18.75 / 21.49 / 33.07	13.69 / 13.83 / 18.76 / 24.61
DNN	13.51 / 13.89 / 15.68 / 28.45	9.98 / 9.87 / 13.79 / 19.11

Table 6.2: Baseline results of CHiME3 system. Results show word error rates [%] using data from only single channel (the best one) and using BeamformIt on all 6 channels. The four results correspond to subsets — dev-simu, dev-real, eval-simu, eval-real.

6.4 REVERB

The features for the REVERB system were 13-dimensional MFCCs with LDA+MLLT transforms. The GMM/HMM system was discriminatively trained using boosted MMI criterion. The final model consisted of about 2000 tied-states with roughly 10 Gaussians per state. The DNN system was built using generative pretraining and discriminative training optimizing cross entropy. The network consisted of 6 layers of 2048 neurons.

system	1-best channel	BeamformIt
GMM	11.91 / 29.75 / 12.12 / 30.39	7.97 / 17.02 / 7.12 / 13.48
DNN (CE)	10.35 / 26.12 / 10.56 / 28.95	7.21 / 15.19 / 6.67 / 12.75

Table 6.3: Baseline results of REVERB system. Results show word error rates [%] using data from only single channel (the best one) and using BeamformIt on all 6 channels. The four results correspond to subsets — dev-simu, dev-real, eval-simu, eval-real.

Chapter 7

Experiments with beamforming

This chapter follows the theory outlined in Chapter 3. It describes the usage of the Delay-and-Sum and MVDR method and the general steps of the beamforming procedure. It also introduces additional modifications of the basic algorithms which improved the performance of the methods. Finally, it shows the overall achieved results and compares the two used methods.

For building the speech recognition systems in this work, we used Kaldi speech recognition toolkit[1] [23]. The tested methods were implemented in Matlab. The accuracy of the speech recognition systems is measured using Word error rate (WER), which refers to the percentage of words that were recognized incorrectly.

7.1 Description of the beamforming procedure

The general implementation of the beamforming procedure follows these steps:

1. **Segmentation** Since the characteristics of the speech signal develop over time and also the direction from the microphone to the speaker can be changing, the signal needs to be processed in short segments. Here, we used windows of 500 milliseconds length with 250 millisecond overlap.

2. **Transform into frequency domain** Fourier transform is applied to each window because most of the computations are more efficient in the frequency domain.

3. **Computation of the delays** Time delay estimation methods introduced in Section 3.3 are used to get the delays corresponding to each microphone in the array. More details about this procedure will follow.

4. **Estimation of the beamformer weights** For delay-and-sum beamformer, the weights are determined directly from the estimated delays. For MVDR, the Equation 3.16 is used together with the noise covariance matrix estimation for computing the weights. More thorough description will follow.

5. **Application of the beamformer** The beamforming filter determined in previous step is used to filter the multichannel signal.

6. **Transform to time-domain and sum of segmented signals** Inverse steps to 1. and 2. For this, we used overlap-and-add procedure.

[1] http://kaldi-asr.org

7.2 Delay-and-sum and its adjustments

Apart from these basic steps, we experimented with several ways to improve the beamforming performance, which will be briefly explained as follows.

Choice of GCC weighting function

To estimate the time delay, we used the generalized cross correlation (GCC) introduced in Section 3.3. Section 3.3 discussed theoretical advantages of using different GCC weighting functions and showed, that they may lead to more accurate estimation of the time delay. In our experiments, we performed beamforming using time delays estimated from GCC-PHAT, GCC-ROTH and GCC-CC (simple cross-correlation) methods and compared the obtained speech recognition accuracy. Table 7.2 shows the word error rates on subsets of CHiME3 dataset.

	dev simu	dev real	eval simu	eval real
GCC CC	10.53	11.53	15.01	21.39
GCC PHAT	9.96	10.23	13.92	19.94
GCC ROTH	10.46	10.49	14.34	20.67

Table 7.1: Comparison of different GCC weighting function on CHiME3 dataset.

From the results, the PHAT weighting shows to be the best choice which is consistent with results published on different tasks. For the following experiments we use GCC-PHAT as implicit weighting.

Weighting of channels

In both Delay-and-sum and MVDR beamforming methods, the contributions of the individual channels to the final signal are all equal. However, the quality of the signal captured by the microphones often varies and some channels may contain more corrupted signal than others. For these reasons, it is reasonable to weigh the signals from the channels in the sum operation. To measure how much the signal from a channel can contribute to the beamforming, we computed average cross correlation to signals at all the other channels and used this as the weighting.

The weight for each channel was computed in the following way

$$w_c \propto \sum_{\substack{c' \geq 1 \wedge c' \leq N \\ \wedge\, c' \neq c}} \Psi^{GCC}_{x_c, x_{c'}}[D_{c,c'}], \qquad \sum_{c=1}^{N} w_c = 1, \tag{7.1}$$

where N is number of the channels, c is the channel to compute weight for, c' iterates over all the other channels and $D_{c,c'}$ is the chosen delay between the two channels.

Table 7.2 shows the obtained results using the weighting of the channels. For all subsets, it led to an improvement, especially on the eval subset.

Detecting reliable parts of utterances

Some parts of the recordings which we use as the input for beamforming may not be very reliable for the estimation of the time delays. This applies mainly for very noisy parts of

	dev simu	dev real	eval simu	eval real
GCC PHAT	9.96	10.23	13.92	19.94
GCC PHAT + weighting	9.91	10.06	12.79	19.09

Table 7.2: The improvement brought by weighting the channels on CHiME3 dataset.

utterances or segments not containing speech. We experimented with two ways to deal with this problem. First, we used Voice activity detector (VAD) based on neural networks [22] for detecting non-speech parts of utterances and eliminated them from the delay estimation process. Second, we detected the frames for which the cross correlation values between the channels were too small (we refer to this as xcorr filter). For such unreliable frames, we simply copied the delays from previous frames.

The results from both of these modifications are shown in Table 7.2.

	dev simu	dev real	eval simu	eval real
GCC PHAT	9.96	10.23	13.92	19.94
GCC PHAT + VAD	9.97	10.15	13.53	19.52
GCC PHAT + xcorr filter	10.20	10.03	13.61	19.32
GCC PHAT + VAD + xcorr filter	9.91	10.01	13.42	19.26

Table 7.3: The improvements brought by detecting the unreliable frames for time delay estimation.

Detecting unreliable frames by both VAD and cross correlation values led to an improvement. However, combining both methods together did not lead to much additional gain. This is caused by the fact that both methods largely agree on the parts which are unreliable. The advantage of using VAD is that it more accurate in detecting the silence parts of the utterances. On the other hand, using the cross correlation values may also identify segments which contain speech though they are too noisy or corrupted in other way, therefore it is beneficial to skip them too. Moreover, the used VAD requires to preprocess all the recordings and thus is not suitable for real-time processing.

Figure 7.1 shows an example of an utterance with the unreliable parts detected by cross correlation based and VAD based detection. The beginning and the end of the utterance which contain silence were considered unreliable by both VAD and cross correlation though VAD was more accurate in detecting the last speech segment. The cross correlation also eliminated some frames in the middle of an utterance which could have lead to an inaccurate estimate of the time delay.

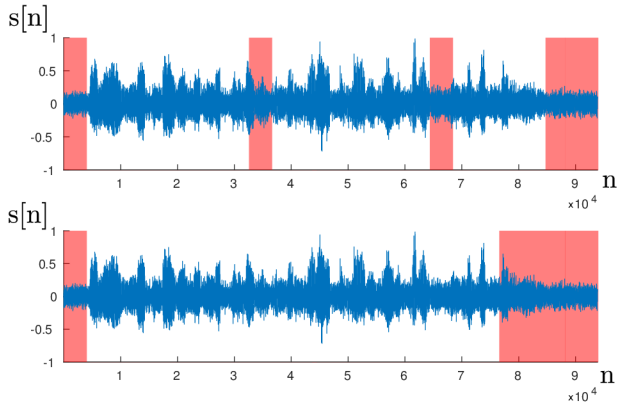


Figure 7.1: Example of detected unreliable parts of an utterance by both cross-correlation (top) and VAD (bottom).

Smoothing of the delay estimates

The estimation of the delays is done per frame although the position of the speaker is not expected to change much during the utterance. It is therefore beneficial to implement some kind of smoothing of the time delays within the utterance. We explored two ways to do this — first, we simply averaged the cross-correlation values in subsequent frames. Second, we aimed to find the best path between the estimated delays through the utterance in a similar way as [1].

The smoothing of the cross-correlation values was done using

$$\Psi_f^{smoothed} = (1-k)\Psi_{f-1}^{smoothed} + k\Psi_f, \tag{7.2}$$

where f is the index of frame, Ψ_f is the generalized cross-correlation function for frame f, k is the smoothing coefficient (which we set to 0.9). The smoothing was done for each channel, the index of the channel is omitted from the formula for simplicity. By doing such smoothing we hoped to encourage the algorithm to select delays closer to the chosen delays in preceding frames and prevent sudden changes in the beamforming direction.

The second method followed the same goal but instead of simply smoothing the values of the cross-correlation function, it computed a score for each delay considering the scores of the delays in the previous frame and also the distances of the subsequent delays. The score was computed as follows

$$\text{score}_{d,f} = \max_{d'} \left(\text{score}_{d',f-1} \times \text{inv\_del\_dist}(d,d') \times \Psi_f[d]\right) \tag{7.3}$$

$$\text{inv\_del\_dist}(d,d') = \frac{\text{max\_del\_dist} - |d-d'|}{\text{max\_del\_dist}}, \tag{7.4}$$

where f is the index of the current frame, d is the evaluated delay in the current frame, d' iterates over all considered delays in the previous frame, $\Psi_f[d]$ is the value of the cross-correlation function for delay d in current frame f. The expression $\text{inv\_del\_dist}(d,d')$ computes the inverse distance between subsequent delays, where max_del_dist is the maximum distance between all the considered delays in the two subsequent frames $f-1$ and f. The scores are computed for each channel independently, the index of the channel is omitted for simplicity. Because this computation would be quite computationally inefficient to evaluate for each possible delay, in each frame, only a limited number of possible delays with

the best cross-correlation values are selected as candidates and the scores are computed only for them. In our experiments six candidate delays has shown to be a good choice. Figure 7.2 demonstrates the principle on a simplified case of three frames and considering only two best delays for each frame.

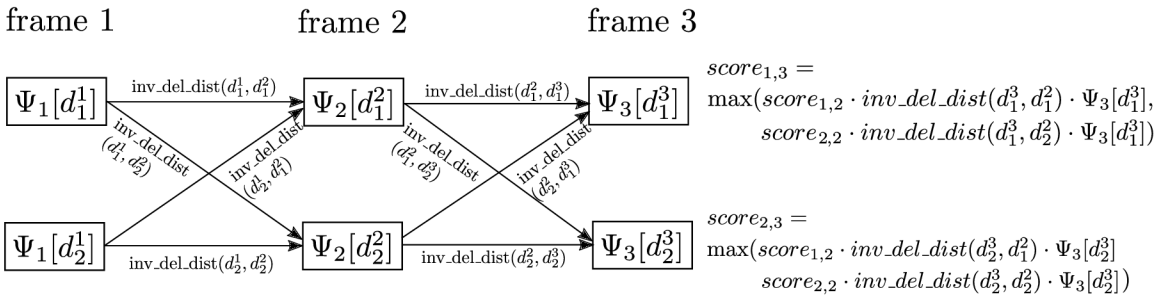


Figure 7.2: The demonstration of selecting the best delay on the case of three frames and considering only 2-best delays for each frame. d_i^f denotes the i-best delay in f-th frame by the cross-correlation value. $\Psi_f[d_i^f]$ is the value of the cross-correlation function for delay d_i^f.

Figure 7.3 shows the example of the delays estimated from an utterance, the smoothed delays using weighted average and smoothed delays using best-path method. The best-path method seems to be superior as it can take into account longer relations than the averaged estimate. Table 7.4 shows the obtained results on CHiME3 dataset.

	dev simu	dev real	eval simu	eval real
GCC PHAT	9.96	10.23	13.92	19.94
GCC PHAT + average smoothing	9.92	10.1	14.1	19.85
GCC PHAT + best-path smoothing	9.85	9.99	14.00	19.49

Table 7.4: Improvements brought by smoothing the time delay estimates across the frames.

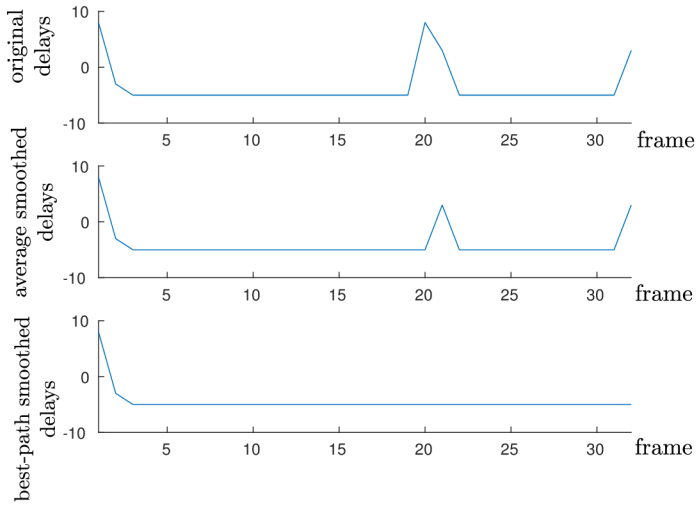


Figure 7.3: Example of delays estimates for an utterance, and both smoothed versions.

Fractional delays

The data we are using as input to beamforming are sampled with 16 kHz frequency. However, the optimal value of the delay used to shift the signal may be at the fraction of the sampling period. In such case, using the sampled cross-correlation function may lead to incorrect estimate of the delay due to insufficient resolution. Figure 7.4 illustrates real example of this problem. In this case, the maximum of the cross-correlation function happened in between the samples, which caused detecting fake maximum three samples apart from the optimal one.

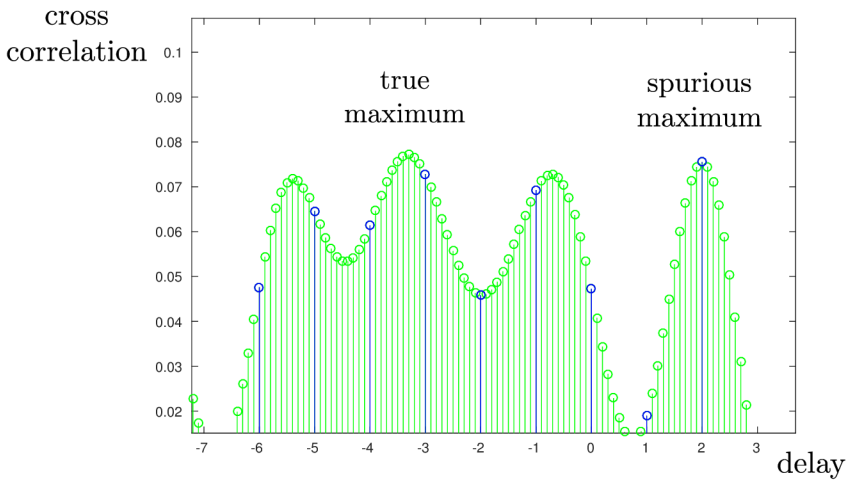


Figure 7.4: Example of detecting spurious maximum of cross-correlation function due to insufficient sampling.

This problem may be solved by interpolating the cross-correlation values between the samples. After detecting such fractional maximum, it is also possible to shift the original signals by fractions of a sample by interpolating the signals. In our experiments we used $\frac{1}{10}$ fractions of the sampling period.

The interpolation was done by zero-padding the spectrum of the signal. The principle is illustrated by Figure 7.5. Extending the spectrum to n times the original length by adding zeros to the middle leads to upsampling the time-domain signal by factor n.

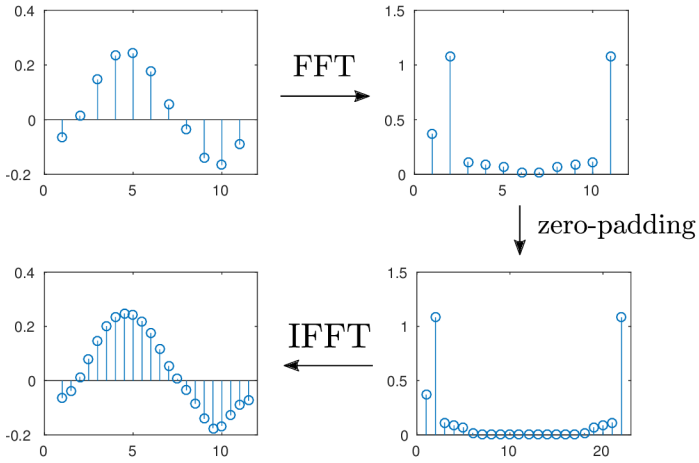


Figure 7.5: The upsampling of the time domain signal using the zero-padding of the spectrum. Figure shows the time-domain signal (top left), frequency-domain signal (top right), frequency-domain signal after zero-padding (bottom right) and upsampled time-domain signal (bottom left).

Table 7.2 shows the results obtained by using $\frac{1}{10}$ fractions for delaying the signal.

	dev simu	dev real	eval simu	eval real
GCC PHAT	9.96	10.23	13.92	19.94
GCC PHAT + fractional delays	9.79	10.11	13.53	19.59

Table 7.5: Improvements brought by using fractional delays.

Determining the reference channel

In the beamforming process, one of the microphones needs to be chosen as a reference. Signal from this microphone is then not being delayed. It is convenient to choose the microphone with the cleanest signal as the reference one. To find the best channel to be used as reference, we computed cross correlation values of each channel to all the other channels and chose the one with largest values

$$\text{reference\_channel} = \arg\max_{c} \sum_{\substack{c' \geq 1 \wedge c' \leq N \\ \wedge\, c' \neq c}} \Psi^{GCC}_{x_c, x_{c'}}[D_{c,c'}], \tag{7.5}$$

where c and c' are the indices of channels, $\Psi^{GCC}_{x_c, x_{c'}}$ is the cross-correlation function between signals from channels c and c' and $D_{c,c'}$ is the chosen delay between signals from these two channels. This selection was done once for the entire utterance.

Figure 7.6 shows the frequency of chosen channels on the CHiME3 dataset. We can see that mostly channel 5 was chosen which corresponds to the used microphone array architecture.

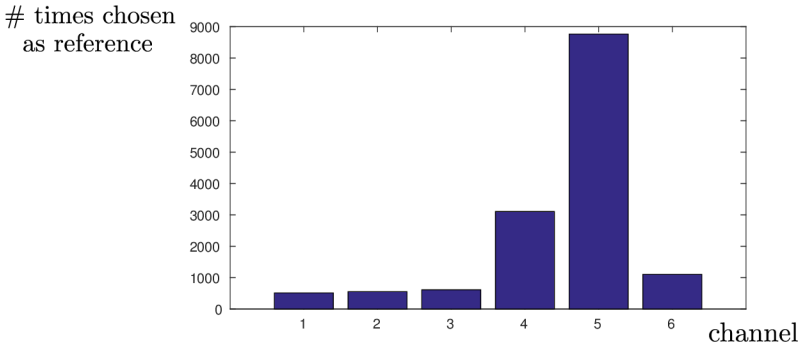


Figure 7.6: The frequency of selecting each of the channels as the reference one on CHiME3 dataset.

Table 7.2 shows the improvement obtained by computing reference channel rather than using a fixed one.

	dev simu	dev real	eval simu	eval real
GCC PHAT	9.96	10.23	13.92	19.94
GCC PHAT + ref channel	10.15	9.96	14.08	19.93

Table 7.6: Improvement brought by automatic selection of a reference channel.

7.3 Minimum variance distortionless beamforming and its adjustments

The implementation of the MVDR beamformer followed the same steps and adjustments as the DS method. The difference of this method lies in the step of estimation of the beamformer weights, which uses Equation 3.16. For this, the knowledge of noise covariance matrix is required. Here, we describe the issues specific to MVDR which are mainly related to the noise covariance matrix estimation and present the results achieved by this method.

Estimation of the noise covariance matrix

The exact computation of the noise covariance matrix $\mathbf{\Sigma}_N(\omega) = E[\mathbf{v}(j\omega)\mathbf{v}^H(j\omega)]$ requires averaging over all the realizations of the noise signal. Practically, the result is approximated by averaging over multiple segments in time. To properly capture the characteristics of the noise, it is necessary to perform the estimation on the parts of the signal without speech. To identify such parts of the signal, we used Voice activity detector based on neural networks [22].

The accuracy of the VAD is a crucial factor in the performance of the MVDR beamformer. If the VAD labels part of signal containing speech as silence, the beamformer evaluates the direction of the speaker as unwanted and suppresses the signals coming from this direction. On the contrary, when the VAD is too greedy and detects speech even in the silence segments, the amount of signal available for noise covariance estimation is decreased and the accuracy of the estimation goes down. To demonstrate this effect, we tried to adjust the threshold of the VAD detection to extreme values and observed the changes in the speech recognition accuracy after the MVDR beamforming. Table 7.8 shows that when

VAD fails to detect parts of speech, the performance of the beamformer drops dramatically. One the other hand, when VAD detects speech even in the silence parts, the accuracy goes closer to the accuracy of the Delay-and-Sum.

	average WER on CHiME3
Delay-and-Sum	12.57
MVDR + VAD threshold -5	12.55
MVDR + VAD threshold -0.5 (optimal)	12.40
MVDR + VAD threshold +5	23.07

Table 7.7: The influence of VAD threshold on the MVDR method. Higher threshold means that less speech is detected.

Diagonal loading

Another issue related to the noise covariance matrix estimation is the technique of diagonal loading which acts as regularizer and thus may be beneficial in cases when the data for the estimation are limited. The modification of the method lies in adding small factor to the diagonal of the noise covariance matrix.

$$\mathbf{\Sigma}_N^{(diag)} = \mathbf{\Sigma}_N + \mathrm{diag}(\epsilon). \tag{7.6}$$

This is equivalent to adding white noise to the signal observed at the microphones. In our experiments diagonal loading has shown to be important for the MVDR beamformer to properly function.

	dev simu	dev real	eval simu	eval real
MVDR	9.62	9.68	13.12	18.5
MVDR + diagonal loading	9.23	9.52	12.95	17.92

Table 7.8: The influence of diagonal loading on the performance of MVDR on CHiME3 dataset.

7.4 Overall results

Finally, we present the overall results of the Delay-and-Sum and MVDR beamforming methods on all three datasets and compare them to results achieved by using toolkit BeamformIt [1].

Table 7.9 shows word error rate of systems with implemented delay-and-sum and MVDR beamforming in comparison with the baseline systems. For AMI dataset, two results correspond to WERs on dev and eval set. For CHiME3 the results are divided to simulated part of dev set, real part of dev set, simulated part of eval set and real part of eval set. REVERB results follow the same division.

	AMI	CHiME3	REVERB
delay-and-sum	47.5 / 52.2	9.53 / 9.64 / 13.02 / 18.12	**5.32** / 21.47 / **5.91** / 15.29
MVDR	**47.1** / **51.6**	**9.23** / **9.52** / **12.95** / **17.92**	-
BeamformIt	48.7 / 53.1	9.98 / 9.87 / 13.79 / 19.11	9.51 / **20.68** / 9.28 / **12.81**

Table 7.9: Overall results of implementation of delay-and-sum and MVDR beamformer.

The obtained results with delay-and-sum method are comparable to the results obtained by BeamformIt toolkit, which also uses delay-and-sum. The usage of the MVDR method leads to additional improvements on both CHiME3 and AMI datasets. On REVERB dataset, we did not test MVDR beamforming as the data does not contain high amount of additive noise.

7.5 Comparison of the methods

As the results suggest, the Minimum Variance Distortionless Response beamforming is more efficient than the simple Delay-and-Sum. Despite its better accuracy the MVDR method also brings some disadvantages which are summarized as follows:

- **the need of accurate VAD** As discussed above, the performance of the MVDR method highly depends on high quality Voice activity detector. The used detection of the speech parts also requires preprocessing of the signals, thus is not very suitable for real-time processing.
- **stationarity of the noise** The properties of the noise are estimated from a short segment preceding the speech. As a consequence, if the noise character changes during the utterance, the MVDR method may perform worse than if no estimate of the noise was used.
- **availability of noise example** If the data does not contain enough amount of silence, the MVDR degrades to Delay-and-Sum method.

In the datasets used in this work the benefits of the MVDR beamforming prevailed and using the method led to better results. However, the method may not be optimal choice in every case and the character of the data should be take into consideration when selecting the method.

Chapter 8

Experiments with dereverberation

In this chapter, we present the results of experiments performed with Weighted prediction error method introduced in Chapter 4. First, we describe the main steps of the implemented method. Then, we study the influence of the parameters, specifically the order of the filter and number of iterations. Then, we show the effect of the dereverberation of the signal and discuss the results.

As with the beamforming methods, we used Kaldi speech recognition toolkit [23] for building the speech recognition systems and the WPE was implemented in Matlab. The accuracy is measured using Word error rate (WER). Most of the experiments in this chapter are performed on the REVERB dataset and the final results are generated on all three datasets - REVERB, AMI and CHiME3.

8.1 Description of the dereverberation procedure

The basic steps of the implementation (with the theory presented in Chapter 4) were the following:

1. **Segmentation of the input signals and transform to frequency domain** In WPE, speech and reverberation are modeled in the time-frequency domain. Therefore we divided the signal into overlapping windows which were then transformed using FFT. We used 32 ms windows with 8 ms frame-shift.

2. Iterating between the estimate of the speech model parameters and reverberation prediction filter.

   - **Estimation of the speech model parameters** We used Levinson-Durbin algorithm for computing the linear prediction filter coefficients, which we then used to estimate the power spectral density of the clean speech. In the initial iteration, the estimate is computed from the corrupted speech (which is equivalent to initializing the reverberation prediction filter coefficients to zeros). In subsequent iterations, the computations are performed on the current estimate of the clean speech.
   - **Estimation of the reverberation prediction filter** This follows Equation 4.9 using the speech variances computed in the previous step.

8.2 Influence of the parameters

Here, we explore the influence of the reverberation prediction filter order and the number of performed iterations on the accuracy of the method.

Filter order

The Weighted prediction error method aims to find a filter which can predict the reverberant component of the corrupted signal from its past samples. The filter acts in time-frequency domain with 8 ms frame-shift (equivalent to sampling frequency 125 Hz). The size of the filter is fixed during the procedure and should be set so that it can properly capture the properties of the reverberation. Figure 8.1 shows the influence of the filter order on the accuracy of the speech recognition on four subsets of REVERB. Note that the comparison was made using a GMM system, thus it does not compare to the overall results.

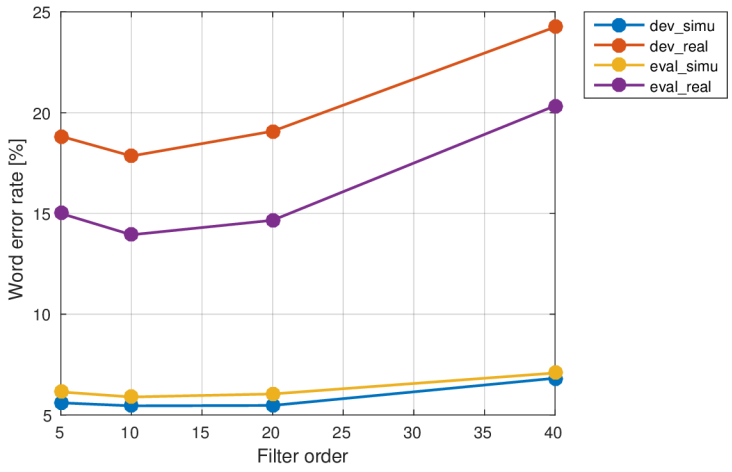


Figure 8.1: The influence of the filter order (5, 10, 20 and 40) on the accuracy of speech recognition on four subsets of REVERB (development and evaluation subset with real or simulated condition).

It can be seen that choosing the correct length of the filter has quite significant influence on the performance. Especially on the real condition, where the error rates are higher, setting the filter order too high leads to significant inaccuracies. The used windows are 32 milliseconds long with 8 millisecond frame-shift, thus the filter order of 40 taps corresponds to looking about 350 millisecond in the past (considering the prediction delay set to 3). Considering such long past segment may not be important for the filter and the increased number of parameters likely causes the the inability of learning the right prediction.

Number of iterations

Since WPE is an iterative method, we studied how many iterations are needed for desired performance. Figure 8.2 shows how the variance of the estimated clean speech evolves through the iterations. The variance is summed across all time points and channels and averaged over all utterances in REVERB. It can be seen that the first two iterations have the most significant effect and after this, the algorithm converges.

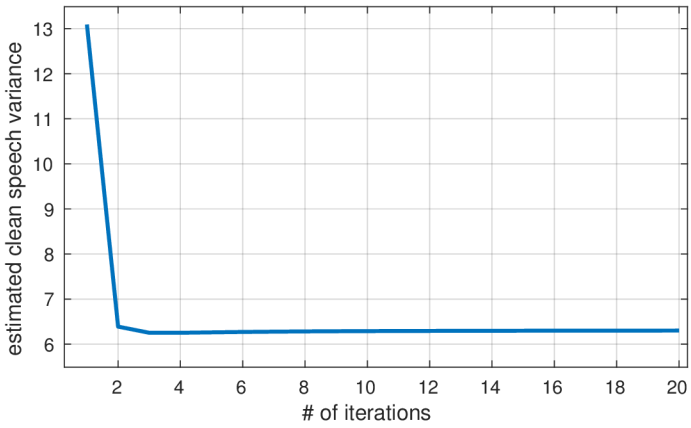


Figure 8.2: Evolution of the variance of estimated clean speech through the iterations. The variance is summed across all time points and channels and averaged over all utterances in REVERB.

As depicted in Figure 8.3, the accuracy of speech recognition confirms these observations. After the first three iterations, the improvement stops or in some cases, the accuracy even degrades.

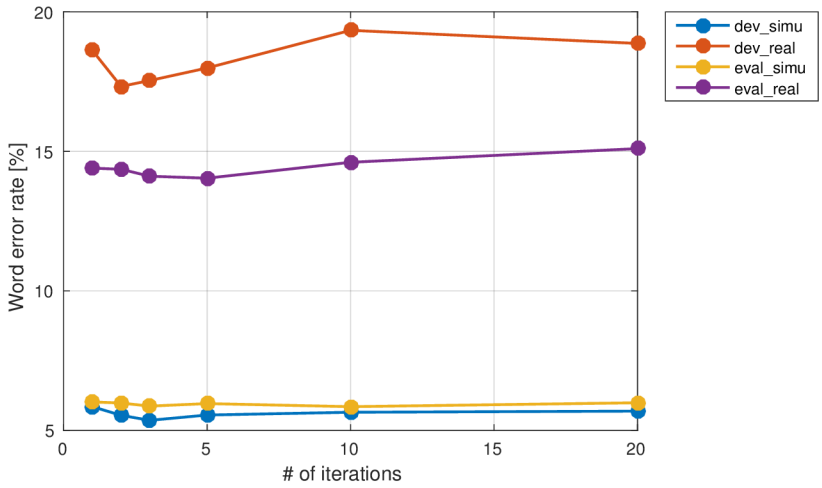


Figure 8.3: The effect of number of iterations of the Weighted prediction error method on the accuracy of speech recognition of four subsets of REVERB.

8.3 Effect of the dereverberation

To show that the method does have the intended effect of removing the reverberant part of speech, we show an example of the dereverberated utterance. Figure 8.4 shows a comparison of spectrograms of a reverberant and a dereverberated utterance. We can see that the WPE method removed the temporal smearing of phonemes caused by reverberation.

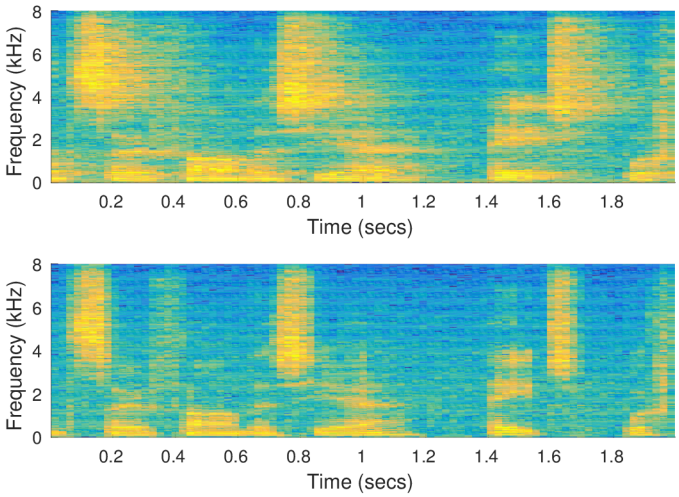


Figure 8.4: Example of the spectrograms of reverberated and dereverberated signal.

8.4 Overall results

To conclude, we show the results obtained by applying the WPE method on all three datasets — CHiME3, AMI and REVERB. Table 8.1 compares the accuracies on the testsets using both MVDR and Delay-and-Sum beamforming together with the WPE.

	AMI	CHiME3	REVERB
DS	47.5 / 52.2	9.53 / 9.64 / 13.02 / 18.12	5.32 / 21.47 / 5.91 / 15.29
DS + WPE	47.61 / 52.14	9.42 / 9.51 / 12.53 / 17.98	**4.89** / **15.01** / **5.56** / **12.65**
MVDR	47.1 / 51.6	9.23 / 9.52 / 12.95 / 17.92	-
MVDR + WPE	**47.07** / **51.53**	**9.15** / **9.42** / **12.43** / **17.53**	-

Table 8.1: Overall results (Word error rates) of Weighted prediction error method on three datasets.

The dereverberation brought substantial improvement on the REVERB dataset, while on CHiME3, the difference was rather moderate and on the AMI dataset, the result did not change much. This is likely caused by the amount of reverberation present in the data, which is strongest in REVERB.

Chapter 9

Neural network based beamforming

Methods for beamforming presented in the previous text are well-established and often used in speech community. Although they work well and significantly outperform the single-channel case, there is still space for improvement. This chapter explores a novel approach for beamforming exploiting neural network framework.

The main idea of the methods in this chapter is to connect the beamforming and the speech recognition into one joint system. This is in contrast with the conventional methods where the preprocessing stage and speech recognition stage are two separate parts which do not share any information. In such setting, the beamforming is following a speech enhancement objective such as noise reduction or higher signal-to-noise ratio. Although these measures are clearly correlated with accuracy of the speech recognizer, the relationship may not be that strong and optimizing the front-end with respect to these measures may not be optimal for the final speech recognition goal.

This motivates the need for joining the back-end classification and front-end preprocessing and learn them together to optimize the final objective of accurate speech recognition. This may enable the beamforming to focus more on the enhancement which is beneficial for the following system. Since the state-of-the-art systems today employ deep neural networks for the acoustic modeling, the direct approach for connecting the acoustic model with the beamforming would be to extend the network to the preprocessing stage. By this, the training of the network will backpropagate the errors back to the beamforming. The following text presents particular approaches for doing this together with the experiments we performed. Note than the work presented in this Chapter is still quite open and will be subject of future research.

9.1 Published approaches

Recently, the topic of neural network based beamforming gained attention and several works in this area were published. Apart from the recently published papers, we will also present two past works related to the same idea.

- **Likelihood maximizing beamforming** Seltzer, in his dissertation [29], followed the idea of using the information from the acoustic modeling in the beamforming procedure. To do this, he optimized the parameters of the beamformer to maximize the likelihood of the best path in the Hidden Markov Model. By this, he achieved substantial improvement in speech recognition accuracy. This approach was developed on GMM-HMM architecture and does not combine well with the Deep neural network

approach which is used today. However, this work shows that joint optimization of beamformer and acoustic model has the potential of improving the performance of the system.

- **Antenna array beamforming** The beamforming methods are not unique to microphone array processing. The same concept can be used for any array of sensors such as antennas. In the field of antenna array signal processing, the beamforming using neural networks has been successfully explored before. In [7], neural methods are used to determine the weights of the beamformer. In this case, the network performs solely the task of beamforming and is trained using known input-output pairs. This differs from our case where we aim to train the beamforming to follow a higher level objective. Nevertheless, the work done in antenna field shows the suitability of using neural networks for the task of array signal processing.

- **Learning the speech front-end using CLDNNs** In series of papers [26, 28, 27] Sainath et al. explored using deep neural networks for both beamforming and feature extraction. The methods build upon convolutional long short-term memory deep neural network (CLDNN) architecture which combines time-convolutional, frequency-convolutional and long short-term memory layers. In [26], Sainath showed the ability of this architecture to learn to extract features similar to auditory-like filterbanks and match the performance of log-Mel features. In [28], the network was further extended to perform multichannel preprocessing. For this, the waveform from each channel is processed by a group of learnable filters represented by the first layer of the network. The outputs of the filters are then summed across the channels and passed to the CLDNN. This was further extended by experiments in [27] where multitask training and factorization of the input layer were incorporated to improve the results.

  The results presented by Sainath show improvement over standard beamforming techniques and the first layer of the network is also shown to learn the spatial filtering. One drawback of this approach is that the learnt filters look into fixed directions and do not adapt to the input data as conventional beamformers. Moreover, the network was trained on relatively large dataset and having more limited amount of training data could influence the ability of learning the desired processing. For these reasons, we decided not to follow this architecture.

- **Deep beamforming** A different approach of neural network based beamforming was published by Xiao in [33]. In contrast with [28], the network architecture in this case is much more constrained. It involves a neural network which learns to predict the beamforming weights from the values of cross-correlation functions between the signals from different channels. The learned weights are then used to process the multichannel signal, followed by fixed feature extraction and a standard DNN classifier.

  Due to the usage of the cross-correlation functions as the input of the network, this architecture should be able to adapt the beamforming to the input data and not focus on fixed directions. As we decided to use this architecture in our experiments, we will describe it in more detail in the following text.

9.2 Used architecture

The architecture of the network we used to jointly learn beamforming and acoustic modeling is inspired by [33]. As depicted in Figure 9.1, it consists of four main blocks — weights estimating DNN, beamforming, feature extraction and DNN classifier. Weights estimating DNN is used to predict the coefficients of the beamforming filter from the values of cross-correlation functions between the channels. This is the part of the network whose purpose is to learn the beamforming. The beamforming block uses the predicted weights to filter the spectra of all channels and to produce the beamformed spectrum. This is passed to feature extraction block which is fixed to perform conventional feature extraction. Extracted features are then used by DNN classifier to predict the HMM states posteriors. In this section, we describe each of these blocks in detail, discuss its inputs, outputs and the way to propagate and backpropagate through the block.

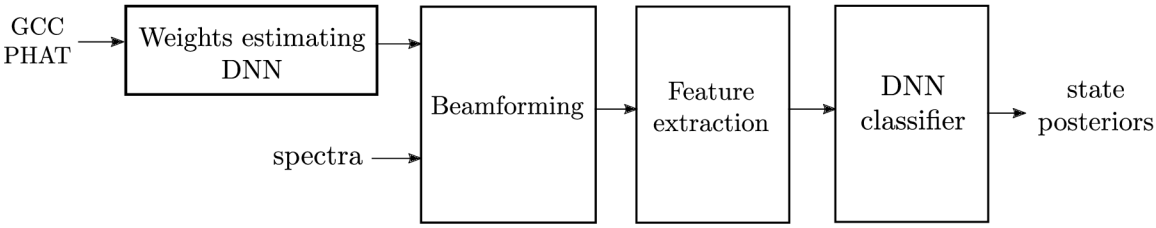


Figure 9.1: The architecture of the deep neural network performing beamforming and classification to HMM states.

Weights estimating DNN

This block aims to predict the optimal weights of the beamformer from the values of cross-correlation functions between the signals from all channels. Its input is vector

$$[\Psi^{PHAT}_{x_{c1},x_{c2}}[-M:M]] \quad \forall\ c_1 = 1..C,\ \ c_2 = 1..C,\ \ c_1 \neq c_2,$$

where $\Psi^{PHAT}_{x,y}[-M:M] = [\Psi^{PHAT}_{x,y}[-M]\ \ldots\ \Psi^{PHAT}_{x,y}[0]\ \ldots\ \Psi^{PHAT}_{x,y}[M]]$, C is the number of channels and M is the maximum considered delay, which we set to 10. The length of the input vector is $(2M+1)\cdot\binom{C}{2}$.

The output of the block is a vector of the predicted beamformer weights $w_c[k]$, where c is the index of channel and k is the index of frequency bin. These weights are complex, thus we represent their real and imaginary component separately as $w^R_c[k], w^I_c[k]$.

The computation of the weights from the cross-correlation coefficients is done by a feed-forward neural network with logistic sigmoid as activation function.

Beamforming block

The inputs to the beamforming block consists of the weights predicted by the weight estimating DNN $w^R_{c,l}, w^I_{c,l}$ and the short-time spectra of all channels computed from 20 ms windows with 10 ms overlap $X_{c,k}$, where c is the index of a channel and k is a frequency bin. The spectra are again represented by their real and imaginary parts $X^R_{c,k}, X^I_{c,k}$. The output of the component is the spectrum of the beamformed signal Y^R_l, Y^I_l.

The computation of the beamforming block is fixed and follows standard equations for complex number arithmetics

$$Y_l = \sum_c w_{c,l} \cdot X_{c,l} = \sum_c (w^R_{c,l} + w^I_{c,l} i) \cdot (X^R_{c,l} + X^I_{c,l} i) \tag{9.1}$$

$$Y^R_l = \sum_c w^R_{c,l} X^R_{c,l} - w^I_{c,l} X^I_{c,l} \tag{9.2}$$

$$Y^I_l = \sum_c w^R_{c,l} X^I_{c,l} + w^I_{c,l} X^R_{c,l}. \tag{9.3}$$

To be able to backpropagate the error from the output of the component towards the Weights estimating DNN, we need to compute the derivatives of each of the outputs with respect to the beamforming weights. This is done by:

$$\frac{\partial Y^R_l}{\partial w^R_{c,l}} = X^R_{c,l} \qquad \frac{\partial Y^R_l}{\partial w^I_{c,l}} = -X^I_{c,l}, \tag{9.4}$$

$$\frac{\partial Y^I_l}{\partial w^R_{c,l}} = X^I_{c,l} \qquad \frac{\partial Y^I_l}{\partial w^I_{c,l}} = X^R_{c,l}. \tag{9.5}$$

Feature extraction block

In the feature extraction block, we implemented full chain of operations which we performed to extract features in previously built systems. As depicted in Figure 9.2, these include:

- extraction of the MFCCs — this includes the absolute value, multiplying by bank of Mel filters, logarithm and Discrete cosine transform (DCT)
- CMN — speaker-dependent cepstral mean normalization
- splice — taking the +-3 context frames
- LDA+MLLT and CMLLR transforms
- splice — taking the +-5 context frames
- global CMVN — speaker-independent cepstral mean and variance normalization

This computation stays fixed during the training of the whole architecture. To be able to train the Weights estimation DNN, we need to be able to backpropagate through the whole feature extraction process. Most of the operations can be interpreted as an affine transform, thus the backpropagation is identical to backpropagating through standard layer in neural networks. The splice components concatenate vectors from their input, which means that to backpropagate through these components need to copy the errors on the output to the inputs following the same pattern. The remaining nonlinear parts of the process are the absolute value and logarithm for which the derivatives of the outputs with respect to their inputs can be derived as follows:

$$\frac{\partial \log(x)}{\partial x} = \frac{1}{x} \tag{9.6}$$

$$\frac{\partial |x|}{\partial x_R} = \frac{\partial \sqrt{x_R^2 + x_I^2}}{\partial x_R} = \frac{x_R}{|x|} \tag{9.7}$$

$$\frac{\partial |x|}{\partial x_I} = \frac{\partial \sqrt{x_R^2 + x_I^2}}{\partial x_I} = \frac{x_I}{|x|}. \tag{9.8}$$

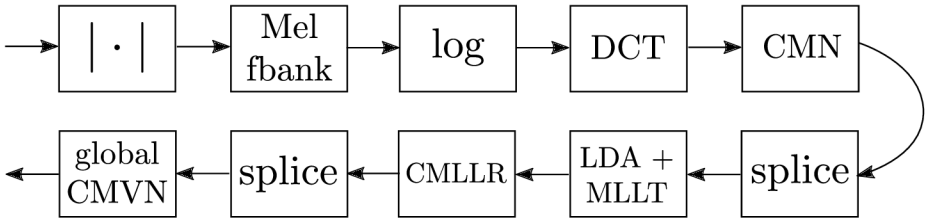


Figure 9.2: Detail of feature extraction block.

DNN classifier

The DNN classifier is the standard part of acoustic model in speech recognition system. Its input are the extracted features and outputs are the posteriors of the Hidden Markov Models states. The propagation and backpropagation process follows the usual neural network training procedure.

9.3 Description of the training

To train the whole architecture, we examined three procedures which differ by the way the Weights estimation DNN and DNN classifier blocks are initialized.

1. The first alternative initializes both blocks to be consistent with delay-and-sum beamforming. The steps are following:
   (a) The Weight estimation DNN is trained to predict the weights of the beamformer derived from the analytic delay-and-sum beamforming. This is done by minimizing mean square error objective.
   (b) The DNN classifier is trained on the analytically beamformed data.
   (c) The whole architecture is assembled and the DNN classifier network in finetuned while keeping the Weights estimation DNN fixed.
   (d) Both blocks are fine-tuned.
2. The second alternative initializes only the Weight estimation network from the delay-and-sum beamforming. The steps are the following:
   (a) (same as in 1) The Weight estimation DNN is trained to predict the weights of the beamformer derived from the analytic delay-and-sum beamforming. This is done by minimizing mean square error objective.

(b) The whole architecture is assembled and the DNN classifier network in trained while keeping the Weights estimation DNN fixed.

(c) Both blocks are fine-tuned.

3. In the third alternative, we let the whole architecture learn jointly from scratch without any initialization.

9.4 Results

Table 9.1 summarizes the results obtained by three training procedures described in the previous section and compares them to the accuracy of the Delay-and-sum beamforming which was used for initialization in the first two alternatives.

	dev simu	dev real	eval simu	eval real
DS	9.53	9.64	13.02	18.12
DNN (1)	**9.44**	**9.28**	12.97	**17.95**
DNN (2)	9.56	9.53	**12.93**	18.25
DNN (3)	12.11	13.24	16.56	21.06

Table 9.1: The results of the neural network based beamforming compared to Delay-and-sum.

The first training procedure, where both the Weight estimation DNN and DNN classifier were initialized from the Delay-and-sum beamforming managed to fine-tune the beamforming procedure to outperform (though moderately) the Delay-and-sum baseline. The second experiment, initializing only the Weight estimation DNN led to similar, slightly worse results. Training the whole architecture without any initialization resulted in notably worse results.

The results show the importance of proper initialization of the network, mainly the Weight estimation DNN. To get rid of the dependence of this initialization on previous analytic beamforming, it could be possible to initialize the network to perform simple sum of all the channels. Overall, the performance of the neural-network based beamforming shows the potential of the architecture to improve the beamforming procedure. However, the achieved improvements are quite small and the performance is worse than the best results achieved with MVDR beamformer. The results could be further improved by exploring alternative training procedures and optimizing the parameters of the whole architecture.

9.5 Future directions

To conclude the discussion of neural network based beamforming, we suggest several ways, how this approach could be further improved.

- **Tuning the training procedure** There are many aspects of the current architecture and training procedure which could be more thoroughly investigated. This includes the ways to initialize the networks, tuning the learning rates and sizes of the networks or using different features on the input of the Weights estimation DNN.

- **Treating the complex values** In the current approach the complex values of the weights are represented by separating their real and imaginary components. This may

not be ideal as the amplitude-phase form better represents the information which is important for the following beamforming. The learning of the amplitude and phase thus may be easier for the network. However, as the phase is circular, this representation could also bring some problems. One of the options to solve this could be using complex-valued networks as explored previously in the antenna array field [31].

- **Data augmentation** The training of the neural networks is highly data dependent. Simulating additional training data by artificially adding noise has been previously shown to bring significant improvements [15]. Moreover, in the case of DNN based beamforming, the simulated data could be effectively used for better initialization using the information about the clean version of the data. Apart from adding the noise, the data could be also largely extended by permuting the channels in the real data.

Chapter 10

Conclusion

10.1 Summary of performed work

In this thesis, we have presented the methods for microphone array processing for the far-field speech recognition task. Particularly, we focused on two beamforming techniques (Delay-and-sum and Minimum variance distortionless response). These techniques enable to combine signals from multiple microphones to reduce the noise. Additionally, we used Weighted error prediction method to deal with the problem of reverberation.

We evaluated the methods on three datasets (AMI, CHiME3 and REVERB) and achieved significant improvement over the single microphone case which corresponds to the published state-of-the-art. We explored modifications of these methods leading to better results. We found that the MVDR method outperforms simple Delay-and-sum and discussed the properties of both methods. We also showed how the dereverberation can further improve the results.

In the end, we investigated a novel method of performing beamforming using neural network framework, which combines the speech recognizer and the beamformer into one joint trainable system. The results of the performed experiments showed mild improvement over the Delay-and-sum method. We suggested several ways how the accuracy could be further improved.

10.2 Future directions

Short-term perspective

In the nearest future, the continuation of this work is two-fold. First, we will use the findings of this work in the projects of the BUT Speech@FIT group. This may involve solving technical problems such as dealing with ad-hoc microphone arrays or more efficient implementation. The second direction, we would like to follow, is further research of the methods to jointly train the front-end and the back-end processing as discussed in Chapter 9.

Long-term perspective

From a broader point of view, there are lots of problems in far-field speech recognition which remain unsolved. Machines still cannot match the ability of humans of separating the voice of interest in very noisy and crowded environments. One of the ways to get closer to the human performance could be higher interconnection of beamforming with speaker recognition systems which could make the beamformer able to more reliably focus on particular speaker.

Appendix A

Derivation of the Minimum Variance Distortionless Response filter

In this section, we present the derivation of the Minimum Variance Distortionless Response filter (MVDR). The notation which was introduced in Section 3.4 is following

ω	angular frequency
n_0	index of the reference microphone
$\mathbf{h}_{n_0}(j\omega)$	the beamforming filter
ξ_{nr}	noise reduction factor
v_{sd}	speech distortion factor
$X_{n_0}(jw)$	frequency-domain speech signal as received by reference microphone
$V_{n_0}(jw)$	frequency-domain noise signal as received by reference microphone
$\mathbf{x}(j\omega)$	the vector of frequency-domain speech signals received by each microphone $\mathbf{x}(j\omega) = [X_1(j\omega)\ \ X_2(j\omega)\ \ ...\ \ X_N(j\omega)]^T$
$\mathbf{v}(j\omega)$	the vector of frequency-domain speech signals received by each microphone $\mathbf{v}(j\omega) = [V_1(j\omega)\ \ V_2(j\omega)\ \ ...\ \ V_N(j\omega)]^T$
$d(\omega)$	the steering vector $d(\omega) = [e^{-i\omega\tau_1}\ e^{-i\omega\tau_2}\ ...\ e^{-i\omega\tau_N}]$, where τ_i is the delay of the speech signal at microphone i with respect to the reference microphone

The MVDR filter is defined as a solution to an optimization problem

$$\begin{aligned} \mathbf{h}_{n_0}^{MVDR}(j\omega) &= \arg\max_{\mathbf{h}_{n_0}(j\omega)} \xi_{nr}[\mathbf{h}_{n_0}(j\omega)] \\ \text{subject to} &\quad v_{sd}[\mathbf{h}_{n_0}(j\omega)] = 0. \end{aligned}$$

Since the noise reduction factor is defined as

$$\xi_{nr}[\mathbf{h}_{n_0}(j\omega)] = \frac{E\{|V_{n_0}(j\omega)|^2\}}{E\{|\mathbf{h}_{n_0}^H(j\omega)\mathbf{v}(j\omega)|^2\}}, \tag{A.1}$$

and the numerator does not depend on the filter $h_{n_0}(j\omega)$, we can rewrite the maximization as minimization of the denominator

$$\mathbf{h}_{n_0}^{MVDR}(j\omega) = \arg\min_{\mathbf{h}_{n_0}(j\omega)} E\{|\mathbf{h}_{n_0}^H(j\omega)\mathbf{v}(j\omega)|^2\}. \tag{A.2}$$

We can also reformulate the constraint. The speech distortion factor $v_{sd}[\mathbf{h}_{n_0}(j\omega)]$ is defined as

$$v_{sd}[\mathbf{h}_{n_0}(j\omega)] = \frac{E\{|X_{n_0}(j\omega) - \mathbf{h}_{n_0}^H(j\omega)\mathbf{x}(j\omega)|^2\}}{E\{|X_{n_0}(j\omega)|^2\}}, \tag{A.3}$$

thus we can derive

$$\frac{E\{|X_{n_0}(j\omega) - \mathbf{h}_{n_0}^H(j\omega)\mathbf{x}(j\omega)|^2\}}{E\{|X_{n_0}(j\omega)|^2\}} = 0 \tag{A.4}$$

$$E\{|X_{n_0}(j\omega) - \mathbf{h}_{n_0}^H(j\omega)\mathbf{x}(j\omega)|^2\} = 0 \tag{A.5}$$

$$E\{|X_{n_0}(j\omega) - \mathbf{h}_{n_0}^H(j\omega)X_{n_0}(j\omega)d(\omega)|^2\} = 0 \tag{A.6}$$

$$E\{|1 - \mathbf{h}_{n_0}^H(j\omega)d(\omega)|^2\} = 0 \tag{A.7}$$

$$\mathbf{h}_{n_0}^H(j\omega)d(\omega) = 1. \tag{A.8}$$

Together, we reformulated the optimization problem to

$$\mathbf{h}_{n_0}^{MVDR}(j\omega) = \arg\min_{\mathbf{h}_{n_0}(j\omega)} E\{|\mathbf{h}_{n_0}^H(j\omega)\mathbf{v}(j\omega)|^2\} \tag{A.9}$$

$$\text{subject to} \quad \mathbf{h}_{n_0}^H(j\omega)d(\omega) = 1. \tag{A.10}$$

To find the optimal $\mathbf{h}_{n_0}^{MVDR}(j\omega)$, we define the Lagrangian function

$$\mathcal{L}[\mathbf{h}_{n_0}(j\omega), \gamma] = E\{|\mathbf{h}_{n_0}^H(j\omega)\mathbf{v}(j\omega)|^2\} + \gamma(\mathbf{h}_{n_0}^H(j\omega)d(\omega) - 1) = \tag{A.11}$$

$$= E\{\mathbf{h}_{n_0}^H(j\omega)\mathbf{v}(j\omega)\mathbf{v}^H(j\omega)\mathbf{h}_{n_0}(j\omega)\} + \gamma(\mathbf{h}_{n_0}^H(j\omega)d(\omega) - 1) = \tag{A.12}$$

$$= \mathbf{h}_{n_0}^H(j\omega)E\{\mathbf{v}(j\omega)\mathbf{v}^H(j\omega)\}\mathbf{h}_{n_0}(j\omega) + \gamma(\mathbf{h}_{n_0}^H(j\omega)d(\omega) - 1) = \tag{A.13}$$

$$= \mathbf{h}_{n_0}^H(j\omega)\mathbf{\Sigma}_N(j\omega)\mathbf{h}_{n_0}(j\omega) + \gamma(\mathbf{h}_{n_0}^H(j\omega)d(\omega) - 1). \tag{A.14}$$

To find the extreme of the Lagrangian function, we differentiate it with respect to the coefficient of the filter

$$\frac{\partial\mathcal{L}[\mathbf{h}_{n_0}(j\omega), \gamma]}{\partial\mathbf{h}_{n_0}(j\omega)} = 2\mathbf{h}_{n_0}(j\omega)\mathbf{\Sigma}_N(j\omega) + \gamma d^H(\omega) \tag{A.15}$$

and equate this to zero

$$2\mathbf{h}_{n_0}^H(j\omega)\mathbf{\Sigma}_N(j\omega) + \gamma d^H(\omega) = 0 \tag{A.16}$$

$$\mathbf{h}_{n_0}^H(j\omega) = -\frac{1}{2}\gamma d^H(\omega)\mathbf{\Sigma}_N^{-1}(j\omega). \tag{A.17}$$

This already gives us the formula of the filter, the last step is to determine the value of γ using A.8

$$\mathbf{h}_{n_0}^H(j\omega)d^H(\omega) = 1 \tag{A.18}$$

$$-\frac{1}{2}\gamma d(\omega)\mathbf{\Sigma}_N^{-1}(j\omega)d(\omega) = 1 \tag{A.19}$$

$$\gamma = -2(d^H(\omega)\mathbf{\Sigma}_N^{-1}d(\omega))^{-1}. \tag{A.20}$$

Substituting A.20 to A.17, we get

$$\mathbf{h}_{n_0}^H(j\omega) = \frac{d^H(\omega)\mathbf{\Sigma}_N^{-1}}{d^H(\omega)\mathbf{\Sigma}_N^{-1}d(\omega)}, \tag{A.21}$$

which is the final formula for the MVDR filter.

Bibliography

[1] X. Anguera, C. Wooters, and J. Hernando. Acoustic beamforming for speaker diarization of meetings. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(7):2011–2022, 2007.

[2] J. Barker, R. Marxer, E. Vincent, and S. Watanabe. The third CHiME speech separation and recognition challenge: dataset, task and baselines. In *Proceedings of IEEE Automatic Speech Recognition and Understanding (ASRU)*, 2015.

[3] Ch. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[4] J. Carletta. Unleashing the killer corpus: experiences in creating the multi-everything AMI meeting corpus. 41(2):181–190, 2007.

[5] J. Chen, J. Benesty, and Y. Huang. Time delay estimation in room acoustic environments: An overview. *EURASIP Journal on Advances in Signal Processing*, (1), 2006.

[6] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 28:357–366, 1980.

[7] K.L. Du, A.K.Y Lai, K.K.M Cheng, and M.N.S Swamy. Neural methods for antenna array signal processing: A review. *Signal Processing*, (4):547–561, 2002.

[8] M.J.F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12:75–98, 1998.

[9] M.J.F. Gales. Semi-tied covariance matrices for hidden Markov models. *IEEE Transactions on Speech and Audio Processing*, 7(3):272–281, 1999.

[10] M.J.F. Gales and S. Young. The application of hidden Markov models in speech recognition. *Found. Trends Signal Process.*, 1(3):195–304, January 2007.

[11] F. Grézl, M. Karafiát, S. Kontár, and J. Cernocky. Probabilistic and bottle-neck features for lvcsr of meetings. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 4, 2007.

[12] E. Habets. *Single- and multi-microphone speech dereverberation using spectral enhancement*. PhD thesis, Technische Universiteit Eindhoven, 2007.

[13] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *The Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990.

[14] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, Na.Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*, 2012.

[15] M. Karafiát, F. Grézl, L. Burget, I. Szőke, and J. Černocký. Three ways to adapt a cts recognizer to unseen reverberated speech in but system for the aspire challenge. In *Proceedings of Interspeech 2015*, pages 2454–2458, 2015.

[16] K. Kinoshita, M. Delcroix, S. Gannot, E. Habets, R. Haeb-Umbach, W. Kellermann, V. Leutnant, R. Maas, T. Nakatani, B. Raj, et al. A summary of the REVERB challenge: state-of-the-art and remaining challenges in reverberant speech processing research. *EURASIP Journal on Advances in Signal Processing*, 2016.

[17] K. Kinoshita, M. Delcroix, T. Nakatani, and M. Miyoshi. Multi-step linear prediction based speech dereverberation in noisy reverberant environment. In *Interspeech*, pages 854–857, 2007.

[18] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, W. A. Sehr, Kellermann, and R. Maas. The reverb challenge: Acommon evaluation framework for dereverberation and recognition of reverberant speech. In *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*, pages 1–4, 2013.

[19] C. Knapp and G.C.Carter. The generalized correlation method for estimation of time delay. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 24(4):320–327, 1976.

[20] K. Kumatani, J. McDonough, and B. Raj. Microphone array processing for distant speech recognition: From close-talking microphones to far-field sensors. *Signal Processing Magazine, IEEE*, 29(6):127–140, 2012.

[21] M. Mohri, F. Pereira, and M. Riley. Weighted finite-state transducers in speech recognition, 2001.

[22] T. Ng, B. Zhang, L. Nguyen, S. Matsoukas, X. Zhou, N. Mesgarani, K. Veselý, and P. Matějka. Developing a speech activity detection system for the DARPA RATS program. Number 9, pages 1–4, 2012.

[23] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, et al. The kaldi speech recognition toolkit. 2011.

[24] T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals. WSJCAMO: a British English speech corpus for large vocabulary continuous speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 81–84, 1995.

[25] P. R. Roth. Effective measurements using digital signal analysis. *IEEE Spectrum*, 8(4):62–70, 1971.

[26] T. N. Sainath, R. J. Weiss, K. W. Wilson A. Senior, and O. Vinyals. Learning the speech front-end with raw waveform cldnns. 2015.

[27] T. N. Sainath, R. J. Weiss, K.W. Wilson, A. Narayanan, and M. Bacchiani. Factored spatial and spectral multichannel raw waveform cldnns. 2016.

[28] T. N. Sainath, R. J. Weiss, K.W. Wilson, A. Narayanan, M. Bacchiani, and A. Senior. Speaker location and microphone spacing invariant acoustic modeling from raw multichannel waveforms. 2015.

[29] M.L. Seltzer. *Microphone Array Processing for Robust Speech Recognition.* PhD thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 2003.

[30] M. Souden, J. Benesty, and S. Affes. On optimal frequency-domain multichannel linear filtering for noise reduction. *IEEE Transactions on Audio, Speech & Language Processing*, 18(2):260–276, 2010.

[31] A.B. Suksmono and A. Hirose. Intelligent beamforming by using a complex-valued neural network. *Journal of Intelligent and Fuzzy Systems*, 15(3-4):139–147, 2004.

[32] P. Swietojanski, A. Ghoshal, and S. Renals. Hybrid acoustic models for distant and multichannel large vocabulary speech recognition. In *ASRU*, pages 285–290. IEEE, 2013.

[33] X. Xiao, S. Watanabe, H. Erdogan, L. Lu, J. Hershey, M.L. Seltzer, G. Chen, Y. Zhang, M. Mandel, and D. Yu. Deep beamforming networks for multi-channel speech recognition. 2016.

[34] T. Yoshioka. *Speech Enhancement in Reverberant Environments.* PhD thesis, Graduate School of Informatics, Kyoto University, 2010.

[35] T. Yoshioka, N. Ito, M. Delcroix, A. Ogawa, K. Kinoshita, M. Fujimoto, Ch. Yu, W.J. Fabian, M. Espi, T. Higuchi, et al. The NTT CHiME-3 system: Advances in speech enhancement and recognition for mobile multi-microphone devices. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 436–443, 2015.

[36] T. Yoshioka and T. Nakatani. Generalization of multi-channel linear prediction methods for blind MIMO impulse response shortening. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(10):2707–2720, 2012.

[37] T. Yoshioka, T. Nakatani, M. Miyoshi, and H.G. Okuno. Blind separation and dereverberation of speech mixtures by joint optimization. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):69–84, 2011.

[38] T. Yoshioka, A. Sehr, M. Delcroix, K. Kinoshita, R. Maas, T. Nakatani, and W. Kellermann. Making machines understand us in reverberant rooms: robustness against reverberation for automatic speech recognition. *IEEE Signal Processing Magazine*, 29(6):114–126, 2012.