Modelling the Nonstationarity of Speech in the Maximum Negentropy Beamformer

Dissertation zur Erlangung des Grades eines Doktors der Philosophie
der Philosophischen Fakultäten der Universität des Saarlandes

vorgelegt von

Barbara Rauch

aus

Bonn-Duisdorf

Saarbrücken, 2010

Dekan:	Prof. Dr. Erich Steiner
Berichterstatter:	Prof. Dr. Dietrich Klakow Prof. Dr. William J. Barry
Datum der letzten Prüfungsleistung:	13.09.2010

Abstract

State-of-the-art automatic speech recognition (ASR) systems can achieve very low word error rates (WERs) of below 5% on data recorded with headsets. However, in many situations such as ASR at meetings or in the car, far field microphones on the table, walls or devices such as laptops are preferable to microphones that have to be worn close to the user's mouths. Unfortunately, the distance between speakers and microphones introduces significant noise and reverberation, and as a consequence the WERs of current ASR systems on this data tend to be unacceptably high (30-50% upwards). The use of a microphone array, i.e. several microphones, can alleviate the problem somewhat by performing spatial filtering: beamforming techniques combine the sensors' output in a way that focuses the processing on a particular direction. Assuming that the signal of interest comes from a different direction than the noise, this can improve the signal quality and reduce the WER by filtering out sounds coming from non-relevant directions. Historically, array processing techniques developed from research on non-speech data, e.g. in the fields of sonar and radar, and as a consequence most techniques were not created to specifically address beamforming in the context of ASR. While this generality can be seen as an advantage in theory, it also means that these methods ignore characteristics which could be used to improve the process in a way that benefits ASR. An example of beamforming adapted to speech processing is the recently proposed maximum negentropy beamformer (MNB), which exploits the statistical characteristics of speech as follows. "Clean" headset speech differs from noisy or reverberant speech in its statistical distribution, which is much less Gaussian in the clean case. Since negentropy is a measure of non-Gaussianity, choosing beamformer weights that maximise the negentropy of the output leads to speech that is closer to clean speech in its distribution, and this in turn has been shown to lead to improved WERs [Kumatani et al., 2009]. In this thesis several refinements of the MNB algorithm are proposed and evaluated. Firstly, a number of modifications to the original MNB configuration are proposed based on theoretical or practical concerns. These changes concern the probability density function (pdf) used to model speech, the estimation of the pdf parameters, and the method of calculating the negentropy. Secondly, a further step is taken to reflect the characteristics of speech by introducing time-varying pdf parameters. The original MNB uses fixed estimates per utterance, which do not account for the non-stationarity of speech. Several time-dependent variance estimates are therefore proposed, beginning with a simple moving average window and including the HMM-MNB, which derives the variance estimate from a set of auxiliary hidden Markov models.

All beamformer algorithms presented in this thesis are evaluated through far-field ASR experiments on the Multi-Channel Wall Street Journal Audio-Visual Corpus, a database of utterances captured with real far-field sensors, in a realistic acoustic environment, and spoken by real speakers. While the proposed methods do not lead to an improvement in ASR performance, a more efficient MNB algorithm is developed, and it is shown that comparable results can be achieved with significantly less data than all frames of the utterance, a result which is of particular relevance for real-time implementations.

Zusammenfassung (German Summary)

Automatische Spracherkennungssysteme können heutzutage sehr niedrige Wortfehlerraten (WER) unter 5% erreichen, wenn die Sprachdaten mit einem Headset oder anderem Nahbesprechungsmikrofon aufgezeichnet wurden. Allerdings hat das Tragen eines mundnahen Mikrofons in vielen Situationen, wie z.B. der Spracherkennung im Auto oder während einer Besprechung, praktische Nachteile, und ein auf dem Tisch, an der Wand oder am Laptop befestigtes Mikrofon wäre in dem Fall vorteilhaft. Bei einer größeren Distanz zwischen Mikrofon und Sprecher werden andererseits aber verstärkt Hintergrundgeräusche und Hall aufgenommen, wodurch die Wortfehlerraten häufig in einen unakzeptablen Bereich von 30–50% und höher steigen. Ein Mikrofonarray, d.h. eine Gruppe von Mikrofonen, kann hierbei durch räumliches Filtern in gewissem Maße Abhilfe schaffen: sogenannte Beamforming-Methoden können die Daten der einzelnen Sensoren so kombinieren, dass der Fokus auf eine bestimmte Richtung gerichtet wird. Wenn nun ein Zielsignal aus einer anderen Richtung als die Störgeräusche kommt, kann dieser Prozess die Signalqualität erhöhen und WER-Werte reduzieren, indem die Geräusche aus den nicht-relevanten Richtungen herausgefiltert werden. Da Beamforming-Techniken sich aus der Forschung an nicht-sprachlichen Daten wie Sonar und Radar entwickelt haben, sind die wenigsten Methoden in diesem Bereich speziell auf das Problem der Spracherkennung ausgerichtet. Während eine Anwendungsunabhängigkeit von Vorteil sein kann, bedeutet sie aber auch, dass Eigenschaften der Spracherkennung ignoriert werden, die zur Verbesserung des Ergebnisses genutzt werden könnten. Ein Beispiel für einen Beamforming-Algorithmus, der speziell für die Verarbeitung von Sprache entwickelt wurde, ist der Maximum Negentropy Beamformer (MNB). Der MNB nutzt die Tatsache, dass „saubere“ Sprache, die mit einem Nahbesprechungsmikrofon aufgenommen wurde, eine andere Wahrscheinlichkeitsverteilung aufweist als verrauschte oder verhallte Sprache: Die Verteilung sauberer Sprache unterscheidet sich von der Normalverteilung sehr viel stärker als die von fern aufgezeichneter Sprache. Der MNB wählt Beamforming-Gewichte, die den Negentropy-Wert maximieren, und da Negentropy misst, wie sehr sich eine Verteilung von der Normalverteilung unterscheidet, ähnelt die vom MNB produzierte Sprache statistisch gesehen sauberer Sprache, was zu verbesserten WER-Werten geführt hat [Kumatani et al., 2009]. Das Thema dieser Dissertation ist die Entwicklung und Evaluierung von verschiedenen Modifikationen des MNB. Erstens wird eine Anzahl von praktisch und theoretisch motivierten Veränderungen vorgeschlagen, die die Form der Wahrscheinlichkeitsverteilung zur Sprachmodellierung, die Schätzung der Parameter dieser Verteilung und die Berechnung der Negentropy-Werte betreffen. Zweitens wird ein weiterer Schritt zur Berücksichtigung der Eigenschaften von Sprache unternommen, indem die Zeitabhängigkeit der Verteilungsparameter eingeführt wird; im ursprünglichen MNB-Algorithmus sind diese für eine Äußerung konstant, was im Gegensatz zur nicht-konstanten Eigenschaft von Sprache steht. Mehrere zeitabhängige Varianz-Schätzungmethoden werden beschrieben und evaluiert, von einem einfachen gleitenden Durchschnittswert bis zum komplexeren HMM-MNB, der die Varianz aus Hidden-Markov-Modellen ableitet. Alle Beamforming-Algorithmen, die in dieser Arbeit vorgestellt werden, werden durch Spracherkennungsexperimente mit dem Multi-Channel Wall Street Journal Audio-Visual Corpus evaluiert. Dieser Korpus wurde nicht durch Simulation erstellt, sondern besteht aus Äußerungen von Personen, die mit echten Sensoren in einer realistischen akustischen Umgebung aufgenommen wurden. Die Ergebnisse zeigen, dass mit den bisher entwickelten Methoden keine Verbesserung der Wortfehlerrate erreicht werden kann. Allerdings wurde ein effizienterer MNB-Algorithmus entwickelt, der vergleichbare Erkennungsraten mit deutlich weniger Sprachdaten erreichen kann, was vor allem für eine Echtzeitimplementierung relevant ist.

Table of Contents

Acknowledgements

This thesis could not have been written without the financial, practical and moral support of many people. First of all I am grateful to my first supervisor Professor Dietrich Klakow and to Dr John McDonough for many fruitful discussions and their overall guidance and help over the years, and to Professor William Barry for agreeing to be my second reviewer. Particular thanks are also due to my colleagues Kenichi Kumatani and Friedrich Faubel for their support and patience, as well as the remaining LSV staff. This thesis was carried out as part of the DFG international research training group IRTG 715 ”Language Technology and Cognitive Systems”, and I am grateful to the funding body and the organising members for giving me the opportunity to conduct it. My fellow IRTG students are the ones who made my time in Saarbrücken particularly memorable, so I thank all of them for the coffee breaks with or without experimental cakes and all the other fun occasions. Finally, I am especially thankful to Eva Lasarcyk, Judith Köhne, Irene Cramer, Jochen Leidner, Alexander Koller and my family for good advice and continuing encouragement in pursuing this project.

Acronyms and Abbreviations

ASR	Automatic Speech Recognition
CMLLR	Constrained Maximum Likelihood Linear Regression
CMS	Cepstral Mean Subtraction
CSD	Cross (Power) Spectral Density
CVN	Cepstral Variance Normalisation
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DOA	Direction Of Arrival
DSB	Delay-and-Sum Beamformer
FFT	Fast Fourier Transform
FT	Fourier Transform
GEV	Generalised EigenValue
GG	Generalised Gaussian
GMM	Gaussian mixture model
GSC	Generalised Sidelobe Canceller
HMM	Hidden Markov Model
IDFT	Inverse Discrete Fourier Transform
LCMV	Linear Constraint Minimum Variance
LM	Language Model
MAPSSWE	Matched Pair Sentence Segment Word Error
MC-WSJ-AV	Multi-Channel Wall Street Journal Audio-Visual Corpus
MFCC	Mel Frequency Cepstral Coefficients
ML	Maximum Likelihood
MLLR	Maximum Likelihood Linear Regression
MM	Moment Method
MMSE	Minimum Mean Square Error
MPDR	Minimum Power Distortionless Response Square Error
ms	milliseconds
MSNR	Maximum Signal-to-Noise Ratio
MVDR	Minimum Variance Distortionless Response
offPE	offline Parameter Estimation
onPE	online Parameter Estimation
pdf	probability density function
PLP	Perceptual Linear Prediction

PSD	Power Spectral Density
SAT	Speaker-Adapted Training / Speaker Adaptive Training
SNR	Signal-to-Noise Ratio
STFT	Short Time Fourier Transform
ULA	Uniform Linear Array
VAD	Voice Activity Detector
VTLN	Vocal Tract Length Normalisation
VTS	Vector Taylor Series
WER	Word Error Rate

Chapter 1

Introduction

For many of today's applications in the field of language technology the input modality of choice is speech rather then text. For example, interactive programs may deal with user requests at a call centre, translate human speech from one language to another, or engage the user in educational dialogue. In other cases non-interactive processing of spoken language is required before the actual interaction with the user, as for video clip search on the Internet or the analysis of audio corpora. Since language processing steps are traditionally applied to written text, the transcription of speech to text is a central task. This task is called *automatic speech recognition (ASR)*, and an ASR module therefore constitutes an important component of the aforementioned type of applications. Sometimes ASR even fulfils the main requirement of an application, as in the case of dictation programs.

Nowadays ASR systems can achieve relatively good results of less than one error in every twenty words, but only under certain conditions, specifically when the language is relatively controlled, the acoustic conditions in which the speech is recorded are favourable, and the system has been adapted to the speaker's voice. However, when one of these factors differs from the expected setting, the number of errors can increase dramatically. The best ASR results are currently obtained for so-called *clean* speech, which is recorded with a headset or lapel microphone close to the mouth, also called *close-talking* microphone. When the microphone is far from the mouth, typically on the table or attached to the wall, the task is called *far-field ASR* or *distant ASR*. The advantage of tether-free communication with a computer system is substantial, as many situations like ASR in meetings or while driving a car, or communication with an environment like a house management system are cumbersome if the user is required to wear a close-talking microphone. The disadvantage of moving the microphone away from the speaker's mouth is that the data is much more difficult to process for current ASR systems. When applying a standard ASR system designed for close-talking data to single channel far-field input, the error rates typically increase manyfold. For example, on the MC-WSJ-AV test set used later in this thesis, the word error rate (WER) is 6.7% for the close-talking data and 28% for a single distant microphone (cf. Section 5.2

SNR	Clean	20dB	15dB	10dB	5dB	0dB
Ratio of Signal to Noise	1:0	10 : 1	5.6 : 1	3.2 : 1	1.8 : 1	1 : 1
WER	0.98%	5.99%	15.67%	36.65%	64.31%	84.68%

Table 1.1: Degradation of word error rate with signal-to-noise ratio for the Aurora-2 task [Pearce and Hirsch, 2000].

for a description of the test set and the definition of word error rate). What are the factors leading to such a significant performance deterioration? There are in fact two aspects which complicate ASR in this case:

1. Sounds other than the speech of interest are captured by the microphones:
   While stationary or nonstationary background noise like another speaker or a machine running in the background may also be picked up by close-talking microphones, when a speaker is recorded with a distant microphone, the same noise will have a much higher level of energy compared to the speech of interest, thereby possibly masking important information in the signal.

2. Reverberation leads to a temporal distortion of sound:
   Sound in an open space or recorded with a headset does not contain any reverberation (echo), but in closed spaces the waves are reflected by walls and objects, leading to multipath-propagation of the signal. The effect on the recording is that of ’smearing’ the signal over time, leading to an overlap with previously uttered sounds.

Standard ASR systems are based on deriving a statistical match between the sound classes they were trained to recognise and the sound waves they encounter in the test scenario. Due to the availability of close-talking data, most ASR systems are trained on clean speech, and in the far-field ASR scenario, the match between the training and test speech is therefore made more difficult by both noise and reverberation. As a consequence recognition performance suffers severe degradation. Table 1.1 gives an indication of the effect additive noise can have on ASR recognition rates. It shows the average WER results for the AURORA-2 connected digits recognition task when evaluated with an ASR system trained on clean speech [Pearce and Hirsch, 2000]. In this task speech from the TIdigits database, consisting of a string of digits spoken by American English talkers, were mixed with a selection of different real-world noises, for example car noise or restaurant noise. The table lists the WER averaged over all eight noise types (test set A and B) for a range of signal to noise ratios (SNR). Since SNR is a logarithmic ratio of signal power to noise power, defined formally by Eq. (3.53), the second row shows the corresponding linear amplitude ratios between the clean speech and the added noise for ease of interpretation. We can observe a dramatic increase from a very low error rate below 1% for the clean test data to more than 84% WER for an equal mix of speech and noise.

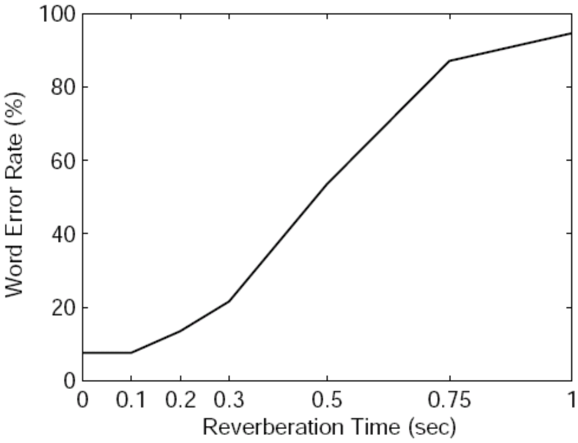


Figure 1.1: The effect of reverberation time on WER (from Seltzer [2003]).

Returning to the effect of reverberation on WER, Fig. 1.1 illustrates this issue as evaluated for the Wall Street Journal corpus by Seltzer [2003]. The horizontal axis plots the *reverberation time* T_{60}, which is defined as the time it takes for the sound level to decrease by 60 dB after the sound source has been switched off. The data was created by varying the mathematical model of a room, simulating different surface materials on the wall and thereby leading to different reverberation times. As for additive noise, we observe a steep increase of WER with increasing reverberation time.

Many different approaches have been proposed to achieve robust ASR in the presence of noise and reverberation. Most of these techniques are applied to the recording of a single distant microphone, but some make use of a *microphone array*, i.e. a group of several microphones, to improve the quality of the speech for ASR or other applications. This thesis is concerned with a method to improve the quality of far-field ASR by use of a microphone array with the technique to be introduced in the following section.

1.1 The Potential of Beamforming

Like humans, who can make use of two ears to locate the origin of sounds and improve understanding of speech, a speech processing system can leverage the availability of several sensors by combining the recorded sounds with a method called *beamforming*. Chapter 3 is dedicated to a formal explanation of beamforming, but in this section we will informally present the central idea of *spatial selectivity*, which focuses processing on sounds arriving from a particular spatial direction, with a simple example.

Consider two spatially separate microphones $M1$ and $M2$ and two sound sources, as depicted in Fig. 1.2. The source on the right labelled S represents the speech of interest, and the source on the left labelled N an interfering noise signal. Due to the different locations, signals emitted by the sound sources will be recorded at each microphone at a different time. In Fig. 1.3, this situation is modelled by showing a graph representing the recording of microphone $M1$ at the top and the equivalent graph for $M2$ below it. The signal plotted in a solid line represents a sound emitted by the speech source, and the dotted line represents a sound emitted by the noise source. Since $M1$ is further from the speech source than $M2$, the speech signal arrives at its location later than at $M2$ (let us denote this delay by d_S). Conversely, the noise source being closer to it leads to the noise signal being recorded earlier than at $M2$ (by d_N seconds). Consider now the situation depicted by the two graphs in the bottom half of the figure. If we shift the signal recorded by $M2$ by d_S seconds to align the two speech signal parts, and add the shifted signal to the original $M1$ recording, we obtain the output shown in the third graph from the top. The speech signal has been doubled, while the noise signal parts were misaligned and stayed at the same level of energy. In relative terms, the speech signal has been amplified. If we decide to normalise the recording back to the level of the original speech by multiplying the microphone signals by a weight (in this case 0.5 for both microphones), the noise signal has been attenuated, as shown in the bottom graph. While we have also introduced an additional (attenuated) noise component, if we were to use more sensors, the attenuation would be even stronger, and the effect of the noise on the overall signal would become smaller.

Instead of amplifying the speech signal, we could just as well amplify the noise instead by shifting the signal recorded at $M1$ by d_N seconds and adding it to the recording from $M2$. Depending on which delays we apply to each channel, signals arriving at the array from a particular direction will be amplified, or equivalently, signals arriving from any other direction will be attenuated, and this is how beamforming achieves spatial selectivity.

The above example constitutes an extremely simple type of beamforming, the delay-and-sum beamformer described later in Section 3.1. We can make the method of combining the microphone signals more complex and obtain more sophisticated beamformers by computing beamformer weights that are optimal with respect to a certain criterion, but the underlying principle of achieving spatial selectivity remains the same.

The application of beamforming as a pre-processing step to far-field speech recognition is becoming standard because its potential for improving the performance of ASR systems is considerable. Even the simple delay-and-sum beamformer can already reduce error rates significantly (for example, from 28% to 16.6% on the aforementioned MC-WSJ-AV test set), and more advanced approaches can improve on this even further. The development of beamforming algorithms specifically tailored to the task of speech recognitions is therefore a promising step in the direction of approximating the error rates obtained for close-talking data data.

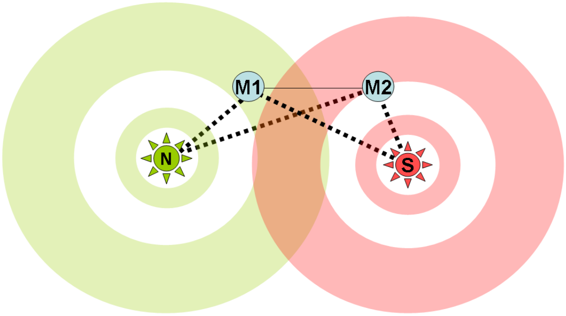


Figure 1.2: A noise (N) source and speech (S) source impeding on a microphone array with 2 sensors.

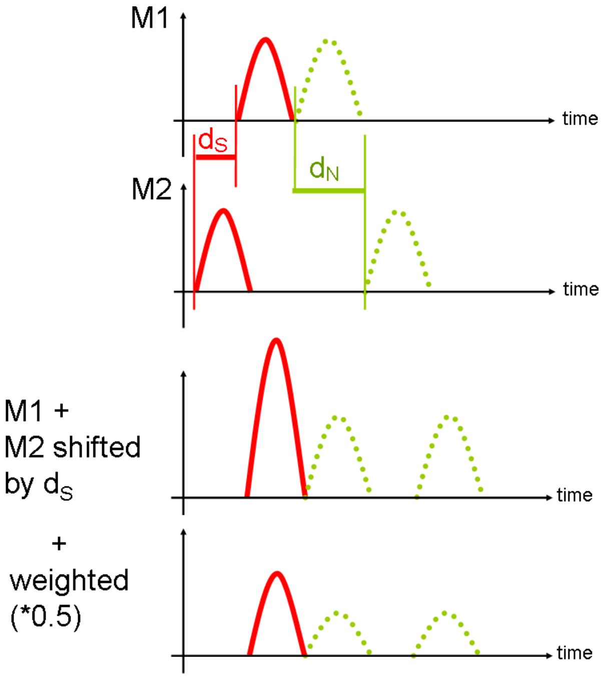


Figure 1.3: Amplification of a sound source with respect to a noise source by shifting and adding the data recorded at two microphones.

1.2 What this Thesis is About

Historically, array processing techniques developed from research on non-speech data, e.g. in the fields of sonar and radar, and as a consequence most beamforming techniques were not created specifically for the ASR context. The independence of an algorithm of the kind of data it processes can be considered an advantage, but it also means that data-specific aspects are ignored which could be used to improve the process in a way that benefits certain applications. For example, when performing beamforming as a preprocessing step to ASR, there are at least two aspects that could be exploited: the characteristics of speech on the one hand, and the internal processes of the ASR system on the other hand.

An example of one of the few approaches that aim to adapt beamforming in a way specifically tailored to ASR is the *likelihood maximisation beamformer (LiMaBeam)* proposed by Seltzer [2003]. LiMaBeam exploits information about the standard ASR process by making the beamformer's optimisation criterion dependent on a property that is central to ASR performance: the likelihood of the test utterance assuming a certain transcription hypothesis.

Another example of beamforming adapted to speech processing is the recently proposed *maximum negentropy beamformer (MNB)*, which exploits the statistical characteristics of speech in broad terms as follows. Clean headset speech differs from noisy or reverberant speech in its distribution, which is much less Gaussian in the clean case. As discussed later in Section 4.2, negentropy is a measure of non-Gaussianity, so choosing beamformer weights that maximise the negentropy of the output should lead to speech that is closer to clean speech in its distribution, and this in turn has been shown to lead to improved WER [Kumatani et al., 2009].

The calculation of the negentropy is defined in terms of a model of the beamformer output as given by a *probability density function* (pdf). The original MNB uses the generalised Gaussian pdf for this task, and the negentropy then depends on three pdf parameters: the variance, the scale parameter and the shape parameter. This thesis is concerned with the introduction of time-dependent values for these parameters, motivated by the following reasoning. In order to model the beamformer output with the GG pdf, the pdf parameters have to be estimated first, and in the original MNB process this was done on a per-utterance basis. The resulting estimate is constant for all frames of an utterance, in spite of the fact that speech is a highly non-stationary signal. By using pdf parameter estimates that vary with time we might expect to get more accurate speech sample models and as a consequence better ASR performance. Several models reflecting the nonstationarity of speech are therefore developed and evaluated in Chapter 7, ranging from a simple moving average model to more complex models reconstructing the spectral envelope with the use of an auxiliary HMM.

While the main focus of the present work is to account for the nonstationarity of speech in the pdf parameter estimates, Chapter 6 describes a number of experiments which were first conducted to ensure the validity of a number of theoretically or practically motivated changes to the original MNB. Specifically, it determines the effect on ASR performance of the following modifications:

1. a change of the algorithm used for the shape parameter estimation to a simpler and more efficient method;

2. a correction of the GG pdf model to a theoretically valid version for the complex-valued beamformer output;

3. a change in the relationship between the estimation of the scale parameter and variance, whereby the definition of negentropy is followed more closely than in the original implementation;

4. the use of the empirical differential entropy rather than the exact differential entropy in the calculation of the negentropy, which is required by the use of time-dependent parameter estimates.

By showing that these modifications do not have a detrimental effect on WER, Chapter 6 paves the way for the experimental evaluation of the non-stationary variance estimates in Chapter 7. Before the novel aspects of the thesis are described, some necessary background material is provided, with chapters 2 and 3 covering the theory of ASR and beamforming respectively, and Chapter 4 describing the original maximum negentropy beamformer.

As the evaluations will show, the overall aim of improved ASR performance could not be achieved with the time-dependent variance estimates proposed in Chapter 7, and the focus therefore shifted to attempting to explain the observed behaviour. While this was partially possible, some questions remain unanswered, and the thesis concludes with a summary of the findings and remaining open questions in Chapter 8.

Chapter 2

Signal Processing and Automatic Speech Recognition Background

The application of focus in this thesis is automatic speech recognition (ASR). This chapter will therefore describe the standard ASR process in order to lay the groundwork for later explanations of the novel aspects of this thesis, including a brief overview of the necessary signal processing background. It also includes a short review of methods developed to deal with noise and reverberation other than beamforming.

2.1 Overall Framework

The aim of an ASR system is the transcription of recorded speech, so sound waves have to be mapped to sequences of words. State-of-the art ASR systems accomplish this task in a statistical pattern matching framework, most commonly using the well-known data structure *hidden Markov model* (HMM) and the associated algorithms. There are alternatives to HMMs for this task, but since they are the established standard technology and the one used for the work of this thesis, this overview will cover HMM-based speech recognition only.

Viewed at a high level, the ASR process consists of a training phase and a testing phase, as illustrated in Figure 2.1. During training, large corpora are processed to build two databases, the *acoustic models* and the *language model* (LM). The acoustic models are HMMs storing representative statistical characteristics of sound classes, such as words, phonemes or even smaller units, extracted from many hours of transcribed speech. The language model represents the probability of word sequences and is usually extracted from large text corpora. During testing, the statistically best match between the sound class models and the recorded speech to be transcribed is determined which also scores best with respect to the language model, and the output transcription corresponding to the best sequence of words is generated. This matching process does not happen at the level of sound waves: both the training and testing data are segmented into units called *frames*, which are

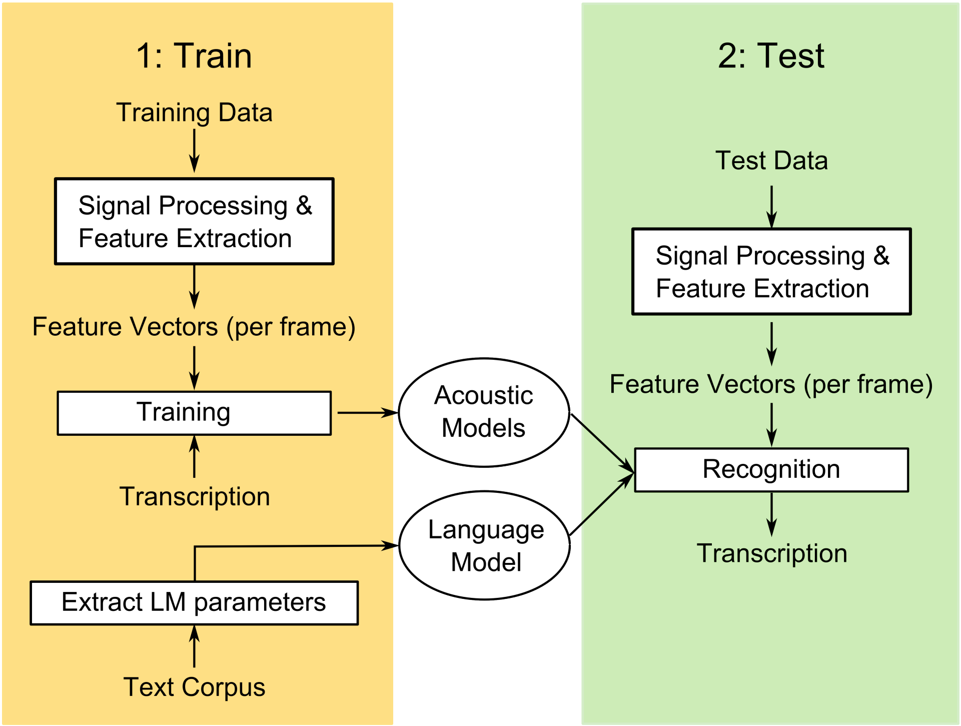


Figure 2.1: The standard high-level ASR Process.

then transformed into compact *feature vectors* by a process called *feature extraction*, to be described in more detail in Section 2.3.

Formally we can express the process as follows. Let the observation $\boldsymbol{o} = \{o_1, o_2, \ldots, o_T\}$ be the sequence of feature vectors to be decoded. We wish to map the observation to a word sequence $\boldsymbol{w} = \{w_1, w_2, \ldots, w_M\}$, specifically to the optimal sequence $\hat{w}$ which is most likely given these observations, i.e.

$$\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w} \in W} p(\boldsymbol{w}|\boldsymbol{o}) \,, \tag{2.1}$$

where W is the set of all word sequences that the ASR system could generate. Since a direct estimation of the probability would be difficult, the expression is broken down into more manageable components using Bayes' formula:

$$p(\boldsymbol{w}|\boldsymbol{o}) = \frac{p(\boldsymbol{o}|\boldsymbol{w})\ p(\boldsymbol{w})}{p(\boldsymbol{o})} \,. \tag{2.2}$$

The term in the denominator, $p(\boldsymbol{o})$, can be ignored because the probability of the given observation is constant when comparing different word sequences. The probabilities in the numerator are the ones modelled by the ASR system, using two different data structures: the probability of the observations given the word sequence $p(\boldsymbol{o}|\boldsymbol{w})$, the *acoustic likelihood*, is computed with the acoustic models (the

HMMs), and the prior probability of the word sequence itself, $p(\boldsymbol{w})$, is obtained from the language model.

The following sections will provide more detail on each of these components. We begin with some necessary signal processing background, followed by the feature extraction in Sec. 2.3. More details on the acoustic models are given in Sec. 2.4 and on the language model in Sec. 2.5.

2.2 Signal Processing Background

While a comprehensive review of the signal processing knowledge on which beamforming is based is outside the scope of this work, this section will briefly review some important concepts relevant to later chapters and introduce the terminology and notation used in the thesis. The reader is referred to the extensive literature on digital signal processing for more details and background information.

2.2.1 Signals and Sampling

In order to process a continuous time signal $x(t)$ with a computer, it needs to be discretised first. The process of *sampling* consists of measuring a continuous sample at a certain frequency, for example 16000 times per second. The *sampling rate* is denoted by f_s and measured in Hertz (samples per seconds, abbreviated Hz), or kilo-Hertz (a thousand Hertz, abbreviated kHz). In order to guarantee the faithful representation of a wave, it must be sampled more than twice per period, so the *sampling theorem* states that f_s must be greater than twice the frequency of the fastest wave component present in a signal. We will denote the sampled discrete-time version of $x(t)$ by $x[k]$, using the sample index k in square brackets to distinguish it from the continuous-time signal with the time index t.

2.2.2 Frequency Domain and Fourier Transform

Any signal can be described as a sum of simple sinusoidal waves of different frequencies, phases and amplitudes. It is often useful to analyse a process acting on a signal in terms of the effect on these frequency components, and a substantial part of signal processing is therefore said to happen in the *frequency domain*.

The analysis process which produces the description of a signal in terms of its frequency components is the *Fourier transform* (FT). For a continuous sequence $x(t)$, the FT can be defined as

$$X(\omega) \triangleq \int_{-\infty}^{\infty} x(t)\, e^{-j\omega t}\,; \tag{2.3}$$

discretising the variable to $x[k]$, we have the *discrete Fourier transform* (DFT) defined as

$$X(m) \triangleq \sum_{k=0}^{M-1} x[k]\, e^{-j(2\pi m/M)k}\,, \tag{2.4}$$

where

- $j = \sqrt{-1}$,
- ω is the continuous frequency variable in Hz,
- m is the index of the DFT output in the frequency domain (the number of the frequency component), which ranges from 0 to M-1 and is also called a frequency *bin*,
- M is the so-called length of the DFT, the number of samples of the input sequence $x[k]$ processed, and also the number of frequency points in the DFT output.

The real-valued time domain signal x and the complex-valued frequency domain representation X are said to form a *Fourier transform pair*, which will be denoted by $x \leftrightarrow X$. In practice an efficient version of the DFT is used, the *fast Fourier transform* (FFT), which restricts M to be a power of 2. When the (D)FT is applied to a short interval of time, it is called the *short time Fourier transform* (STFT).

The ensemble of M DFT outputs is called the *spectrum* of the input signal. It consists of M complex values, so visualisations generally split it into the magnitude spectrum and the phase spectrum. A plot of the magnitude spectrum against time is called a *spectrogram.*

The DFT is conjugate symmetric, meaning that for a real-valued input sequence, the DFT output for the mth bin will have the same magnitude as the $(M - m)$th bin and the negative of the phase angle of the same bin. Formally,

$$X(m) = X^*(M - m)\,, \tag{2.5}$$

where * denotes complex conjugation. Due to the conjugate symmetry of the spectrum, a complete description of the spectrum is given by specifying it for the first $(\frac{M}{2} + 1)$ bins only.

Another property of the FT is known as the shifting theorem. Expressed for a continuous sequence, it states that a shift by a units in the time domain corresponds to the scaling of the spectrum by a complex multiplicative factor of $e^{j\omega a}$ in the frequency domain. Formally,

$$X_{\text{shifted}}(\omega) = e^{j\omega a} X(\omega)\,. \tag{2.6}$$

For a discrete sequence, we have

$$X_{\text{shifted}}(m) = e^{j(2\pi m/M)a} X(m)\,. \tag{2.7}$$

To reconstruct the time domain signal from the spectrum, the *inverse DFT* (IDFT) can be applied, which is defined as

$$x[k] \triangleq \frac{1}{M} \sum_{k=0}^{M-1} X(m)\, e^{j(2\pi m/M)k}\,. \tag{2.8}$$

2.2.3 Digital Filters and Filter Banks

A *filter* is a system that changes the spectral content of a time-domain signal; in general, it reduces or filters out some unwanted spectral content. For example, a *band-pass filter* lets a certain range of frequencies (the *passband*) pass through the system unchanged, while attenuating or removing the other frequencies (the *stopband*). We will restrict the present discussion to finite impulse response filters of linear time-invariant systems, a widely used subset sufficient for our purposes. A digital filter can be implemented as a sequence of L coefficients $h[l]$, which are combined with the input signal to yield the filter output $y[k]$ by a process called *convolution*, defined as

$$y[k] = h[k] * x[k] = \sum_{l=0}^{L-1} h[k]x[k-l]\,. \tag{2.9}$$

The sequence of coefficients is also known as the *impulse response* of the filter, since it is the filter output when the input is an impulse, a single unity-valued sample. When a time-domain signal is convolved with the impulse response of a filter, the corresponding frequency-domain representations are multiplied. Formally, if $x \leftrightarrow X$ and $h \leftrightarrow H$, then

$$y = h[k] * x[k] \quad \leftrightarrow \quad Y(m) = H(m) \cdot X(m) \tag{2.10}$$

where the spectrum of the impulse response $H(m)$ is called the*frequency response* of the filter. The frequency domain part of the equation can be transformed to

$$H(m) = \frac{Y(m)}{X(m)}$$

to illustrate that the frequency response is the ratio of the output spectrum to the input spectrum.

Several band-pass filters with a common input or summed output are called a *filter bank*. The output of one filter in a filter bank with common input is called a *subband*, so filter banks provide a means to transform signal representations between the time domain and the *subband domain*. An *analysis* filter bank transforms a signal from the time domain to the subband domain, while a *synthesis* filter bank recombines a set of subband signals into a time-domain signal. An analysis filter bank is generally designed so that the passbands of all filters span a frequency range of interest contiguously (or with a small overlap), so that each filter isolates a frequency portion and the whole spectrum is represented in the output. This function may sound similar to the FT role, and in fact applying an STFT is equivalent to a certain class of filter banks called uniform DFT filter banks. For these filter banks, the same impulse response template, the *prototype*, is modulated (shifted) to each subband's centre frequency. Traditionally, filter banks were designed for subband coding [Vaidyanathan, 1993], and applications like adaptive processing and beamforming were only considered by later developments, e.g. [De Haan et al., 2003] [de Haan, 2001]. In the experiments of this thesis a Nyquist(M) filter bank [Kumatani et al., 2008] is used, which was developed specifically for beamforming.

2.2.4 Power of a Signal and Power Spectral Density

The energy E present in a discrete time signal $x[k]$ within a specific sample interval $[N_1, N_2]$ is defined as the sum of the squared magnitudes

$$E_{[N_1,N_2]} = \sum_{k=N_1}^{N_2} |x[k]|^2, \text{with } N_2 > N_1 \,.$$

Power is defined as energy per time unit. The power P_x of the signal $x[k]$ is therefore

$$P_x = \frac{1}{2K} \sum_{k=-K}^{K} |x[k]|^2 \,.$$

For a continuous signal $x(t)$, we have

$$E_{[N_1,N_2]} = \int_{N_1}^{N_2} |x(t)|^2 \, dt \,,$$

and

$$P_x = \frac{1}{2T} \int_{-T}^{T} |x(t)|^2 \, dt \,.$$

Depending on the length of the interval (the size of K or T), we obtain different estimates of the power, with the limiting value $\lim_{T\to\infty} P_x$ generally called the *average* power, and the *instantaneous* power defined as $\lim_{T\to 0} P_x$. If we consider x to be a random variable, its power is an estimate of $E\{|x|^2\}$. If furthermore the mean of x is zero, this is also an estimate of its variance σ_x^2.

When power refers to a frequency variable $X(\omega)$, P_X is called the *power spectral density* (PSD) - the amount of power per unit of frequency (hence the term spectral) as a function of frequency. It indicates how strongly a frequency component of a signal varies in energy. The term *power spectrum* is sometimes informally used as a more informal synonym for the term PSD, or to refer to the plot of the PSD versus frequency - i.e. like a spectrogram with each spectral term squared.

For a complex scalar random variable X with mean $\mu_X = 0$ the variance is defined as

$$\sigma_X^2 = E\{|X - \mu_X|^2\} = E\{|X|^2\} = E\{XX^*\} \,, \tag{2.11}$$

where $E\{\}$ is the statistical expectation operator. The PSD of a zero-mean frequency variable can therefore be defined as $P_X = E\{XX^*\}$ and constitutes a variance estimate, as power does in the time domain. If we take two different frequency-domain variables X and Y, the expression

$$P_{XY} = E\{XY^*\} \tag{2.12}$$

is known as the *cross (power) spectral density* (CSD).

Let us extend the above concepts to *vector* random variables, using two different complex column vector r.v.s $\boldsymbol{X}$ and $\boldsymbol{Y}$ with mean vectors $\boldsymbol{\mu_X} = \mathbf{0}$ and $\boldsymbol{\mu_Y} = \mathbf{0}$. Instead of the variance we then have the cross-covariance matrix

$$\boldsymbol{P_{XY}} = E\{(\boldsymbol{X} - \boldsymbol{\mu_X})(\boldsymbol{Y} - \boldsymbol{\mu_Y}^H)\} = E\{\boldsymbol{XY}^H\}\,. \tag{2.13}$$

When $\boldsymbol{X} = \boldsymbol{Y}$, we denote the resulting covariance matrix with a single subscript as in $\boldsymbol{P_X}$. Moreover, when the vectors represent a complex spectrum $\boldsymbol{X}(\omega)$ and $\boldsymbol{Y}(\omega)$, $\boldsymbol{P_X}$ is called the *PSD matrix* of $\boldsymbol{X}$, and $\boldsymbol{P_{XY}}$ the *CSD matrix* of the two vectors.

2.3 The Front-End

The term front-end generally refers to all pre-processing of the recorded sound wave files before the pattern matching stage at the feature vector level. We will focus on the actual feature extraction phase here, but note that other preparatory processes (such as beamforming) are sometimes considered part of the front-end as well. The description will be at a high level only because in contrast to the definitions given in the previous section, the details are not required for the description of the work undertaken for this thesis, except for the computation of the cepstral vectors, which is described in Sec. 7.4.1.

The purpose of feature extraction is to reduce the dimensionality of the data and transform the input into a form that is most suitable for the classification task. Ideally, the features should clearly separate the phonetic classes of interest while at the same time being invariant to irrelevant aspects such as noise and reverberation, or speaker-dependent factors like voice quality or accent. The most popular features used nowadays for ASR are *mel frequency cepstral coefficients* (MFCC) [Mermelstein, 1976] and *perceptual linear prediction* (PLP) [Hermansky, 1990]. Dynamic features estimating the change of rate from frame to frame are often added to the feature vector to improve recognition [Furui, 1986][Yang et al., 2005]. These *delta features* may be estimated in different ways, for example by simply taking the difference of consecutive frames, but more complex methods provide more reliable estimates. In addition to the first-order rate of change, *delta delta features* estimating the second-order rate of change, acceleration, are also commonly used.

The ASR features used for the experiments reported in this thesis are based on a recently proposed alternative front-end, the warped *minimum variance distortionless response* (MVDR) cepstral coefficients, which have been shown to lead to improvements in ASR performance compared to MFCC and PLP features [Wölfel and McDonough, 2005].

2.4 Acoustic Models

Sec. 2.1 introduced the idea of modelling the acoustic likelihood $p(\boldsymbol{o}|\boldsymbol{w})$ with a set of acoustic models. This section explains how HMMs can be used to perform this task. Again, the literature on this subject is extensive, and any detailed descriptions will be restricted to those aspects relevant to the experimental work presented in this thesis.

2.4.1 Hidden Markov Models

A hidden Markov model is a stochastic model comprising a set of probabilities associated with so-called *states*, which can be thought of as generating or emitting a feature vector observation o_t to be classified for ASR. An HMM can be fully described by the quintuple $\boldsymbol{\lambda} = (S, A, B, s_{\text{start}}, s_{\text{end}})$, where

- $S = \{s_1, \ldots, s_N\}$ is the set of states;
- $A = \{a_{ij}\}$ is the matrix of transition probabilities, with each a_{ij} representing the probability of transitioning from state i to state j;
- $B = \{b_i(o_t)\}$ are the emission probabilities, the likelihood that an observation o_t is generated by state i, which are usually modelled by probability density functions for the purpose of ASR;
- $s_{\text{start}} \in S$ is the initial state of the HMM, and
- $s_{\text{end}} \in S$ is its final state.

An example HMM is shown in Fig. 2.2, where the states are represented by circles, the transition probabilities are marked on the arrows between the states, and the dashed arrows point to the state-associated pdfs modelling the emission probability. The initial and final states are non-emitting, i.e. they do not generate a feature vector and have no associated emission probability. A sequence of feature vectors is generated by the model by transitioning through the HMM from the initial to the final state as governed by the transition probabilities, generating a feature vector when entering an emitting state as determined by the emission pdfs. The resulting feature vector sequence is directly observable, while the sequence of states is hidden and must be inferred from the observed feature vectors and the knowledge of the HMM parameters.

For ASR one HMM is constructed per phonetic unit, which could be a word but is more typically a phone. As mentioned in Sec. 2.1, the aim of using HMMs in ASR is to compute $p(\boldsymbol{o}|\boldsymbol{w})$, the probability of the sequence of feature vectors $\boldsymbol{o}$ assuming that one or more words $\boldsymbol{w}$ were uttered. When using one HMM per phone, a link between the word level and the phonetic level is therefore required. It is given by a *lexicon*, which lists all possible pronunciations for a word in terms of sequences of phones. A composite HMM representing a word w can then be constructed by connecting the phone-level HMMs at the non-emitting states, and by concatenating the word-level HMMs in the same way we obtain the HMM for the word sequence $\boldsymbol{w}$, whose parameters will be denoted by $\boldsymbol{\lambda}_{\boldsymbol{w}}$.

Let $\boldsymbol{s} = \{s_1, s_2, \ldots s_T\}$ be a particular state sequence of length T, also called a *path* through the model, and S the set of all possible paths through the HMM with

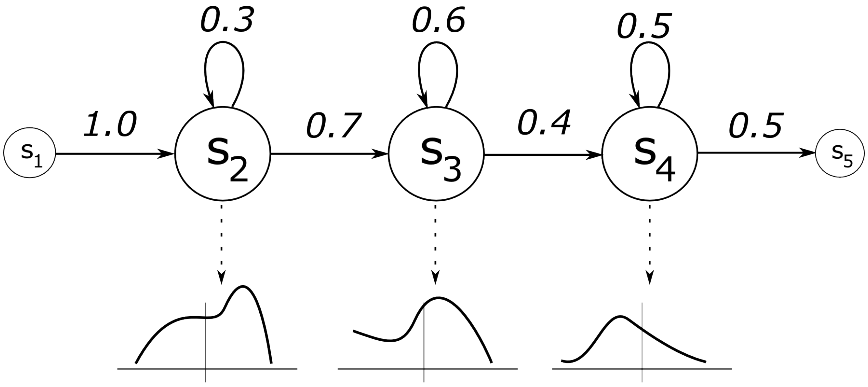


Figure 2.2: An example of a hidden Markov model with three emitting states.

parameters $\boldsymbol{\lambda_w}$. The probability that the feature vector sequence $\boldsymbol{o} = \{o_1, \ldots, o_T\}$ is generated by this HMM can then be expressed by summing over all possible paths, yielding

$$\begin{aligned} p(\boldsymbol{o}|\boldsymbol{w}) = p(\boldsymbol{o}|\boldsymbol{\lambda_w}) &= \sum_{\boldsymbol{s} \in S} p(\boldsymbol{o}, \boldsymbol{s}|\boldsymbol{\lambda_w}) \\ &= \sum_{\boldsymbol{s} \in S} p(\boldsymbol{o}|\boldsymbol{s})\, p(\boldsymbol{s}|\boldsymbol{\lambda_w}) \\ &= \sum_{\boldsymbol{s} \in S} \left[\prod_{t=1}^{T} a_{t,t+1} \right] \left[\prod_{t=1}^{T} b_t(o_t) \right] , \end{aligned} \tag{2.14}$$

since the HMM's emission probabilities determine the likelihood of the observation sequence for a given path, $p(\boldsymbol{o}|\boldsymbol{s})$, and its transition probabilities the likelihood of the path through this model, $p(\boldsymbol{s}|\boldsymbol{\lambda_w})$.

For ASR the emission probabilities are most often modelled by Gaussian pdfs or *Gaussian mixture models* (GMMs). In other words, the emission probability of state s_i for the feature vector o_t can be expressed as

$$b_i(o_t) = \sum_j c_{j,i} \cdot p_{\text{Gauss}}(o_t; \mu_j, \Sigma_j) \tag{2.15}$$

where μ_i and Σ_i are respectively the mean and covariance matrix of the Gaussian pdf p_{Gauss}, and $c_{j,i}$ is the mixture weight of the jth pdf in the state's mixture model. The covariance matrix is often assumed to be diagonal, which implies that only the variance per feature vector dimension needs to be stored.

2.4.2 Decoding

The act of mapping an input feature vector to a word sequence is also known as *decoding*. One way to find the optimal word sequence $\hat{w}$ of Eq. (2.1) would be to evaluate Eq. (2.14) for all the basic and composite HMMs we would consider for our application. For *isolated word recognition*, where one utterance is assumed to consist of a single word, this would mean evaluating one HMM per pronunciation in the lexicon, but for *continuous speech recognition*, which in principle allows sentences of any length, the approach becomes intractable. An approximation involving only the most likely state sequence(s) is therefore used, and this is efficiently computed by the *Viterbi algorithm* [Forney, 1973], which is an instance of the *dynamic programming* approach [Bellman, 1957]. Note that a state sequence provides an alignment of feature vectors to HMM states and therefore indirectly the decoded sequence of words.

The decoding process may produce the single most likely hypothesis, but often it is useful to consider several good hypotheses, which can be compactly stored in a *word lattice*. Such a lattice can be implemented as an acyclic graph whose nodes correspond to words, and whose arcs represent transitions between word ends and are associated with the probabilities determined during decoding.

Lattices are particularly useful when using large knowledge sources like vocabularies or LMs, because the decoding process can be split into two parts: an initial phase uses simple knowledge sources to create a word lattice, and a second stage called *lattice rescoring* applies more computing-intensive resources to the constrained search space of the lattice to obtain more accurate results efficiently.

When the lattice representing the search space for phone-level recognition is restricted to a single sequence of words, the decoding is known as *forced alignment* or *(forced) Viterbi alignment*. This can be used to reduce the workload involved in creating phone-labelled corpora, such as the training corpora described hereafter, since it requires only the word-level transcriptions of the recorded data and not the time-consuming labelling of the phone boundaries.

2.4.3 Training

The HMM probabilities need to be estimated before they can be used to decode a sequence of observation vectors. This estimation process is called *training*, and it involves the use of a training corpus consisting of many hours of speech which have been labelled with the correct phonetic classes. The most common estimation method is the *Baum-Welch algorithm*, which iteratively adjusts the observation likelihoods $b_i(o_t)$ and transition probabilities a_{ij} to find a local solution according to a *maximum likelihood* (ML) criterion[1]. Assume for a moment that we have a single Gaussian pdf per state and an assignment of training data feature vectors to states based on the current model parameters. Let O_i be the set of vectors assigned to state i in this way, of size $|O_i|$. The new mean and covariance matrix of the pdfs could then be updated to the familiar expressions for sample mean and covariance,

$$\hat{\mu}_i = \frac{1}{|O_i|} \sum_{t \in O_i} o_t$$

[1] The principle of ML estimation is explained in more detail in Sec. 4.4.2

and

$$\hat{\Sigma}_i = \frac{1}{|O_i|} \sum_{t \in O_i} (o_t - \mu_t)^T (o_t - \mu_t)\,,$$

where $()^T$ denotes the transpose of a vector. Now instead of a "hard" assignment of vectors to frames, the Baum-Welch algorithm makes a "soft" probabilistic assignment, resulting in the update formulae

$$\hat{\mu}_i = \frac{\sum_t c_i(t)\, o_t}{\sum c_i(t)}$$

$$\hat{\Sigma}_i = \frac{\sum_t c_i(t)\, (o_t - \mu_t)'(o_t - \mu_t)}{\sum c_i(t)}\,,$$

where $c_i(t)$ is the probability of being in state i a time t, which is computed from the so-called forward and backward probabilities. For more details on the calculation of these values and a full description of the Baum-Welch algorithm refer to [Baum, 1972],[Jelinek, 1997], or [Rabiner and Juang, 1993].

2.5 Language Model

The language model serves to compute the prior probability of the word sequence, $p(\boldsymbol{w})$, as described in Sec. 2.1. This is most commonly implemented through an N*-gram* model, which approximates the probability of a word by considering the previous $N-1$ words only. Formally, the probability of the sequence can expressed as

$$\begin{aligned} p(\boldsymbol{w}) &= p(\{w_1, w_2, \ldots, w_M\}) \\ &= p(w_1) \cdot p(w_2|w_1) \cdot p(w_3|\{w_1, w_2\}) \cdot \ldots \cdot p(w_M|\{w_1, \ldots, w_{M-1}\}) \\ &= \prod_{n=1}^{M} p(w_n|\{w_1, \ldots, w_{n-1}\})\,. \end{aligned}$$

An N-gram model then uses the approximation

$$p(w_n|\{w_1, \ldots, w_{n-1}\}) \approx p(w_n|\{w_{n-N+1}, \ldots, w_{n-1}\})\,.$$

For example, a *bigram* model ($N = 2$) calculates

$$p(\boldsymbol{w}) \approx \prod_{n=1}^{M} p(w_n|w_{n-1})\,.$$

The N-gram probabilities are estimated on a large text corpus before recognition. The higher the number of words considered in an N-gram, the more prominent the problem of data sparsity becomes, as not all combinations of N words will occur in the corpus. Methods to handle estimates for word combinations that do not occur include *smoothing* and *backoff*. The former takes away some probability mass from seen words and redistributes it to unseen ones, while the latter resorts to a shorter sequence as a knowledge source when the N-gram is not available. For more details on these methods see e.g. [Jurafsky and Martin, 2000, §6].

The LM probabilities can be incorporated into the graph that results from combining single word HMMs into a large network by associating them with the transitions between HMMs. The decoding stage can then evaluate the combined probability of the acoustic likelihood and the language model jointly. Note that it is customary to multiply the language model log probability with an *LM weight* to increase the importance of the LM as compared to the acoustic likelihood.

In a similar way, the *insertion penalty* is a value that is often used to influence the length of utterances. It is a fixed value that is added to or deducted from the log probability when transitioning from one word HMM to the next, thereby favouring longer or shorter word sequences.

2.6 Adaptation and Normalisation

The mismatch between training and test data is a fundamental problem in ASR. Differences in recording environment and equipment or voice characteristics can lead to a large increase in recognition errors. These mismatches can be reduced in two ways:

- in *model (space) adaptation*, the parameters of the acoustic models are modified to better match the test utterance features;

- in *feature (space) adaptation*, the test utterance features are transformed to fit the acoustic models more closely.

This section summarises a number of common adaptation methods which were used in the experiments conducted for this thesis.

One of the most popular feature space adaptation methods for channel distortions is *cepstral mean subtraction* (CMS), also called cepstral mean normalisation [Atal, 1974]. It involves the subtraction from each frame of the cepstral feature vector average, generally computed across the whole utterance. Another method applied to the cepstral vectors is *cepstral variance normalisation* (CVN), which consists of scaling each feature to achieve unit variance. This has been shown to reduce the effect of additive noise [Hain et al., 1998].

Speaker-dependent variability in speech can be related to a number of factors including speaking rate, accent, or timbre, some of which can be related to physical qualities, like the length of the vocal tract. For example, the latter causes formants (the spectral peaks of a sound) to shift approximately linearly. The idea behind the widely used *vocal tract length normalisation* (VTLN) is to account for this by applying a linear warping of the frequency axis before the cepstral features are extracted [Andreou et al., 1994].

2.6.1 MLLR and CMLLR

Maximum likelihood linear regression (MLLR) is a popular model adaptation method which estimates a linear transform to be applied to the parameters of a set of HMMs such that the likelihood of the adaptation data is maximised. Note that this step requires a transcription of the adaptation data. Indicating the speaker by s, we can express the transformation of an HMM mean $\boldsymbol{\mu}$ as

$$\boldsymbol{\mu}^{(s)} \triangleq \mathbf{A}^{(s)}\boldsymbol{\mu} + \mathbf{b}^{(s)} \,,$$

and for the covariance matrix we have

$$\boldsymbol{\Sigma}^{(s)} = \boldsymbol{H}^{(s)}\boldsymbol{\Sigma}\boldsymbol{H}^{(s)T} \,,$$

where $\mathbf{A}^{(s)}$ and $\boldsymbol{H}^{(s)}$ are the speaker-dependent transformation matrices and $\mathbf{b}^{(s)}$ is the cepstral bias vector.

Constrained MLLR (CMLLR) [Gales, 1998] is a variant of MLLR where the transformation of the covariance matrix is the same as the one for the means, i.e. $\boldsymbol{H}^{(s)} = \boldsymbol{A}^{(s)}$. The advantage of this constraint is that CMLLR can be implemented by operating on the feature space, which is computationally more efficient than MLLR.

There is a variant of the ML training procedure described in Sec. 2.4.3 which uses speaker adaptation parameters, known as *speaker-adapted* (or *speaker adaptive*) *training* (SAT) [Anastasakos et al., 1996]. Here the training data is divided by speaker, and the adaptation transforms are estimated accordingly. A set of speaker-independent models is then estimated based on these transforms. We will refer to the resulting HMMs as ML-SAT models.

2.7 Noise and Reverberation Compensation Techniques

The previous section presented a number of techniques to reduce the mismatch in environmental or speaker conditions in the feature and model space. These approaches provide only limited robustness against noise and reverberation, and many other methods have been specifically developed to improve the recognition of speech in noisy and reverberant conditions. While the main background to the novel work of this thesis consists of the algorithms presented in the beamforming chapter 3, this section gives a brief overview of other methods aimed at robust ASR.

As mentioned in Sec. 2.3, the ASR features extracted from speech would ideally be noise-robust by nature, and the aforementioned PLP features have in fact been shown to possess this property to a certain degree [Hermansky, 1990]. *RASTA-PLP* features are an enhanced version developed to increase this robustness further [Hermansky and Morgan, 1994]. Another approach referred to as *multi-style training* consists of the use of training data with various environmental characteristics.

According to Gales and Young [2008], both of these methods are limited in practice, and active noise compensation methods are required below SNRs of 15 dB.

Spectral subtraction is a well-known feature space noise compensation technique, which consists of the subtraction of a noise spectrum estimate from the signal spectrum [Boll, 1979]. Denoting the transform pairs of the desired signal in the time and short term frequency domain by $s(t) \leftrightarrow S(m)$, an additive noise component by $n(t) \leftrightarrow N(t)$, and the overall noise-corrupted signal at a single sensor by $x(t) \leftrightarrow X(m)$, their relationship can be modelled by

$$x(t) = s(t) + n(t) \quad \leftrightarrow \quad X(m) = S(m) + N(m)\,,$$

where as before t and m are the time index and frequency bin respectively. The clean speech spectrum is then estimated by subtracting an estimate of the noise spectrum $\hat{N}$:

$$|\hat{S}(m)|^\gamma = |\hat{X}(m)|^\gamma - |\hat{N}(m)|^\gamma\,,$$

where $\gamma = 1$ to subtract magnitude terms, or $\gamma = 2$ to use power terms. The noise estimate $N(m)$ is typically based on utterance segments without speech, which can be determined by use of a *voice activity detector* (VAD) [Beritelli et al., 2001]. Other methods exist ([Cohen, 1989],[Hirsch, 1993], [van Compernolle, 1989]), but in any case an exact estimate is difficult to obtain, and a distortion called musical noise often occurs in the output.

More sophisticated approaches to reconstructing the clean speech by modelling the effect of the corruption and then subtracting it from a noisy speech representation generally attempt to perform the process in the domain of the speech features, i.e. the log mel spectral domain when MFCC coefficients are used, or at least after the non-linear transformation of the logarithm, and may be based on more complex data models. Instead of only accounting for additive noise, the convolutive effect of the room and recording channel can be modelled with the room impulse response $h(t)$, yielding the following data model:

$$x(t) = h(t) * s(t) + n(t) \quad \leftrightarrow \quad X(m) = H(m) \cdot S(m) + N(m)\,.$$

Defining the log spectra $X_l(m) = \log X(m)$, $H_l(m) = \log H(m)$, and $S_l(m) = \log S(m)$, $N_l(m) = \log N(m)$, we can express this relationship in the log domain as

$$\begin{aligned} X_l(m) =& \log\left[H(m) \cdot S(m) + N(m)\right] \\ =& \log\left[\exp\log(H(m)) \cdot \exp\log(S(m)) + \exp\log(N(m))\right] \\ =& \log\left[\exp(H_l(m) + S_l(m)) + \exp N_l(m)\right] \\ =& \log\left[\exp(H_l(m) + S_l(m)) \cdot (1 + \exp(N_l(m) - (H_l(m) + S_l(m)))\right] \\ =& H_l(m) + S_l(m) + \log\left[1 + N_l(m) - S_l(m) - H_l(m))\right]\,, \end{aligned}$$

or more generally in the form

$$X_l(m) = S_l(m) + f(S_l, H_l, N_l, m)\,, \tag{2.16}$$

where the function f could be any function describing the transformation clean speech undergoes due to the noise and environment. If we model the offset given by f, we can subtract it from the log spectrum of the noisy speech to reconstruct the log spectrum of the clean speech [Acero, 1991]. However, this function is non-linear and non-trivial to estimate. Moreno et al. [1996] developed a popular method which approximates f as a linear function by help of a first order vector Taylor series (VTS) expansion. The compensation method assumes that the clean speech can be modelled by a GMM, which can be trained on a corpus of clean speech. It furthermore assumes that the noise is well represented by a single Gaussian pdf

$$p(n) = p_{\text{Gauss}}(n; \mu_n, \Sigma_n)\,, \tag{2.17}$$

and that H_l represents a constant channel effect which does not change. Approximating f by a linear function in this context implies that the noisy speech can also be expressed as a Gaussian mixture model, whereby the value of f depends on the mixture component. Based on an initial estimate of H_l, μ_n and Σ_n, the algorithm obtains an estimate of the noisy speech using the VTS expansion, after which the mean and covariance matrices of the noisy speech GMM can be estimated. An iteration of the expectation maximisation algorithm [Dempster et al., 1977] is then performed to re-estimate H_l, μ_n and Σ_n. This process is repeated until the likelihood of the noisy data converges. The results can be used in one of two ways: either in the feature space, by subtracting the estimate of the transformation offset based on (2.16) to reconstruct the clean speech, or in the model space, by updating the GMMs of the HMMs to reflect the noisy speech. A major advantage of this method is that, unlike spectral subtraction, it does not require a VAD step to identify periods of noise.

Regarding model-space compensation, the HMMs for clean speech are often supplemented by one or more models for noise. For example, in *HMM decomposition* [Varga and Moore, 1990] noise and speech are simultaneously recognised by sets of parallel HMMs. In *parallel model combination* [Gales and Young, 1993] on the other hand, clean speech HMMs are first combined with noise HMMs into models for corrupted speech.

As a compromise between the computationally simple but limited feature compensation and more powerful but computationally demanding model compensation methods, *uncertainty-based* approaches have been proposed. A well-known example are *missing feature techniques* [Cooke et al., 1994], which associate a binary value with each frequency bin of each frame indicating if the feature is reliable or not, thereby constituting a mask over the spectrogram of the utterance, and then compute the transcription likelihoods given the mask. *Uncertainty decoding* is another uncertainty-based approach, which instead of assigning binary confidence values attempts to incorporate the knowledge of the accuracy of the features directly into the speech recogniser [Droppo et al., 2002] [Liao and Gales, 2005]. For more details on both uncertainty-based approaches, see e.g. [Gales and Young, 2008].

2.7.1 Blind Source Separation and Independent Component Analysis

Blind source separation (BSS) attempts to recover a set of individual signals which have been mixed together. In the context of speech enhancement or recognition, at least one of these sources is the desired speech, which is to be separated from other sources or noise. The term BSS covers a group of techniques which assume very little or no information about the signals to be separated - hence the qualifier *blind*. For example, they require no knowledge about the positions of the microphones or speakers. However, they generally assume that the number of sources is known, and they work by making certain assumptions about the characteristics of the signals to be separated. A popular subset of BSS algorithms is based on *Independent Component Analysis* (ICA), which generally assumes that the observed signal is a linear mixture of statistically independent sources with a non-Gaussian pdf. To separate the signals, ICA algorithms estimate a *demixing matrix* $\boldsymbol{W}$ which produces separated signals by maximising the statistical independence or non-Gaussianity of the output. Assuming we have n independent source signals $\boldsymbol{s} = \{s_1, \ldots, s_n\}$, mixed together and observed as n output signals $\boldsymbol{x} = \{x_1, \ldots, x_m\}$, both column vectors, the problem can be expressed as

$$\boldsymbol{x} = \sum_{i=1}^{n} \boldsymbol{a}_i s_i = \boldsymbol{A}\boldsymbol{s}\,, \tag{2.18}$$

where $\boldsymbol{A}$ is the unknown *mixing matrix* with columns $\boldsymbol{a_i}$ whose effect the demixing matrix is supposed to reverse. Slightly different definitions exist, but the above is the most popular one [Hyvärinen, 1999]. Using only the knowledge of $\boldsymbol{x}$, ICA attempts to estimate the demixing matrix $\boldsymbol{W}$ so that it approximates $\boldsymbol{A^{-1}}$ in order to obtain an estimate of the sources $\hat{\boldsymbol{s}}$ from $\boldsymbol{x}$:

$$\hat{\boldsymbol{s}} = \boldsymbol{W}\boldsymbol{x} \approx \boldsymbol{A}^{-1}\boldsymbol{A}\boldsymbol{s} = \boldsymbol{s}\,. \tag{2.19}$$

This is done by adaptively calculating the rows of $\boldsymbol{W}$ and maximising an optimisation criterion defined for $\hat{\boldsymbol{s}}$. Various criteria have been used in ICA, for example mutual information, a measure of statistical independence, or kurtosis and negentropy, two measures of non-Gaussianity [Gallager, 1968]. The reasoning behind the use of non-Gaussianity as a criterion is two-fold:

1. The central limit theorem states that the pdf of the sum of independent random variables of any pdf will approach a Gaussian pdf as more and more components are added, so the pdf of the sum of several random variables will be more Gaussian than the pdf of the individual addends [Rice, 2001].

2. As discussed further in Sec. 4.2.1, Gaussian variables can be considered the least predictable of all random variables. Signals of interest however are often more predictable than a Gaussian r.v. due to the structured information they convey, and therefore their pdf generally follows a non-Gaussian pdf.

In the context of ICA both points imply that the sources can be identified by choosing the demixing matrix such that the source pdfs are maximally non-Gaussian.

Two-well known ambiguities arise from the fact that both $\boldsymbol{A}$ and $\boldsymbol{s}$ in Eq. 2.18 are unknown:

- Multiplying a source s_i by a factor can be cancelled by dividing the corresponding column of $\boldsymbol{A}$ by the same value, implying that the real magnitude (or variance, when speaking of the random variable) of the sources cannot be determined. This issue, known as the *scaling ambiguity*, is usually addressed by simply assuming unit variance for each source. While the negative or positive sign of the values is still ambiguous in this case, this is not a significant problem for many applications.

- The *permutation ambiguity* refers to the fact that changing the order of the source vector rows can be compensated by changing the order of the mixing matrix columns. Various methods have been proposed to solve this problem, e.g. by taking into account the geometry of the array [Saruwatari et al., 2006].

ICA depends on the statistical independence and therefore uncorrelatedness of the sources. This assumption may not hold in practice; for example, in case of reverberation, the echos are delayed and attenuated signal of the desired source, and the components may therefore be correlated. Like many ICA algorithms, the maximum negentropy beamformer which this work builds upon also employs negentropy as a criterion to achieve maximum non-Gaussianity, albeit in a beamforming context, which does not make the independence assumption. Before the details of this algorithm are introduced in Chapter 4, the theory of beamforming will be described in more detail in the next chapter.

Chapter 3

Standard Beamformers

This chapter presents the core of the background material, the theory underlying beamforming. Chapter 1 introduced the basic idea of using several microphones to focus the processing on sounds coming from a particular direction. In this chapter the idea is formalised and a number of well-known beamformer types are presented.

We will begin by examining the simplest of all beamformers, the delay-and-sum beamformer. This will allow us to introduce a number of basic concepts used in a more general description of beamforming, the filter-and-sum beamformer. Sections 3.6 to 3.8 will then present and compare compare several variants of the filter-and-sum beamformer, which differ in the optimisation criteria that are used to determine the beamforming parameters. Finally, we will look at the generalised sidelobe canceller, a convenient implementation of certain beamformer types. The problems associated with the default configuration will be covered, including some attempts to address them. The generalised sidelobe canceller is important because it is also the basis of the maximum negentropy beamformer to be described in Chapter 4.

3.1 Delay-and-Sum Beamformer

The *delay-and-sum beamformer (DSB)*, sometimes called conventional beamformer, is the simplest type of beamformer. As the name suggests, it combines the channels of the array by applying a time shift to each microphone's signal and then summing and dividing by the number of sensors to average the time-aligned signals. We can express this formally as follows. Let N be the number of microphones, x_n the signal received at microphone n, and τ_n the time shift applied to the signal received at microphone n. Formally, the output y of the microphone array can then be described as

$$y(t) = \frac{1}{N} \sum_{n=0}^{N-1} x_n(t + \tau_n) \,, \tag{3.1}$$

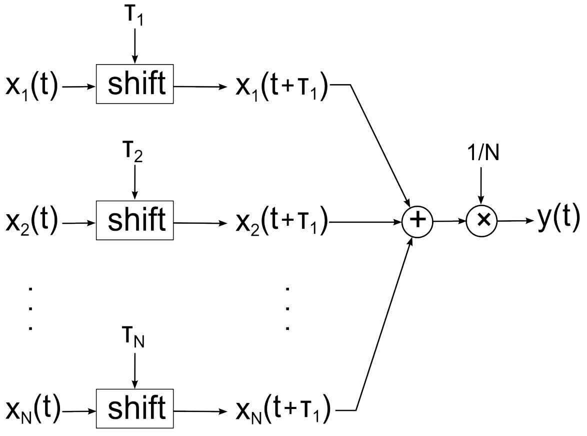


Figure 3.1: Delay-and-Sum beamformer in the time domain.

where t is the continuous time variable. Figure 3.1 depicts the DSB operation.

As explained in Chapter 1, the goal of the DSB is to emphasise a desired signal coming from a target direction by aligning the signals from the channels appropriately so the signals of interest are added constructively, while noise coming from other directions is suppressed. We will now examine in more detail how the DSB accomplishes this spatial selectivity. In order to do so, we will introduce a concept called the *array frequency response*: an expression that relates the beamformer's input, expressed in terms of a signal coming from a specific direction, to its output. By examining a graphical representation of the beamformer response, the *beam pattern*, we will see that for certain directions the incoming signals will be amplified, and for others they will be attenuated. This effect is called *spatial filtering*.

3.1.1 Propagation Delay and Steering Angle

The first important concept to consider is the *propagation delay*, the time it takes a sound to travel from one place to another. The signals x_n arriving at each microphone are related to each other in a way determined by the physical laws of sound wave propagation and the array geometry.

A microphone array is a multiple-input single-output device, but is usually modelled as a single-input single-output device by relating the microphone signals to a single *source signal* $s(t)$, which could for example be the voice of a speaker, and whose restoration or approximation we consider to be of real interest to our application. There are many different ways of modelling the relationship between the source and microphone signals. For example, we could make the simplifying assumption that each $x_n(t)$ is a delayed version of $s(t)$. This would account for the propagation delay, but it would ignore other aspects that influence the signal in the real world, for example the attenuation of energy over distance, the effect of the room or the microphone characteristics on the signal, or the influence of noise.

However, we will use this approximation for now to simplify the introduction of the basic beamforming concepts. In Sec. 3.4 we will introduce a more complex model including the effect of additive noise. Formally, the simple delay approximation can be described by

$$x_n(t) = s(t - \delta_{s,n}) \, , \tag{3.2}$$

where $\delta_{s,n}$ is the time the signal takes to travel from the source to microphone n.

Beamforming can be considered an attempt to reconstruct the source signal s. However, instead of reconstructing the source signal, it is generally sufficient to reconstruct a *reference signal r(t)* received at a reference point, for example the middle of the array or the first microphone. This is because r itself is considered a delayed version of s, and the delay does not matter in many cases. Denoting by $\delta_{s,r}$ the propagation delay between the source and the reference point, we have

$$r(t) = s(t - \delta_{s,r}) \, . \tag{3.3}$$

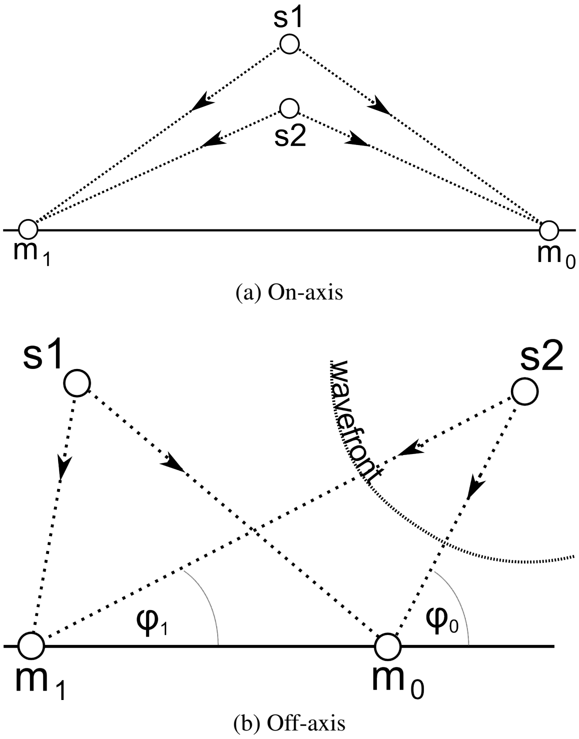
Denoting by $\delta_{r,n}$ the propagation delay between the reference point and microphone m_n, we can use the difference in propagation delays

$$\Delta_{r,n} = \delta_{s,n} - \delta_{s,r} \tag{3.4}$$

to express the microphone signals as delayed versions of the reference signal:

$$x_n(t) = r(t - \Delta_{r,n}) \, . \tag{3.5}$$

The set of propagation delays for the microphone array will depend on the relative location of the source with respect to the array. Consider the microphone arrays depicted in Figure 3.2. In Fig. 3.2(a) the sources s1 and s2 are both *on-axis*, i.e. at the same distance to both microphones, and therefore a wave emitted by either of these sources will arrive at the same time at both microphones. In Fig. 3.2(b) on the other hand, the sources are *off-axis*, and the delays will be different for sources s1 and s2. If we wanted to express the delays in terms of the angle of the incident wave, this would be different for each microphone of the array. These *direction of arrival (DOA)* angles ϕ_n are shown in Fig. 3.2(b) for the source s2. Both configurations in Fig. 3.2 depict a source close to the array. Sometimes the so-called*far-field assumption* is made, according to which the source is sufficiently far away from the microphone array so that the DOA angles can be considered the same for all microphones. In contrast to the near-field case, where sound waves are modelled with a spherical wavefront, sound is now modelled as *plane waves*, as depicted in Figure 3.3. Even though the far-field assumption is rarely justified in the case of using microphone arrays for speech recognition, we will use it in this chapter to simplify the explanations. All equations can be extended to the near-field case though (see e.g. Doclo [2003]). Given the far-field assumption, the DOA angles ϕ_n are all equal, and the delay differences $\Delta_{r,n}$ between the reference point and each microphone can therefore be expressed as



(a) On-axis

(b) Off-axis

Figure 3.2: A signal from near sources arriving at different microphone array configurations.

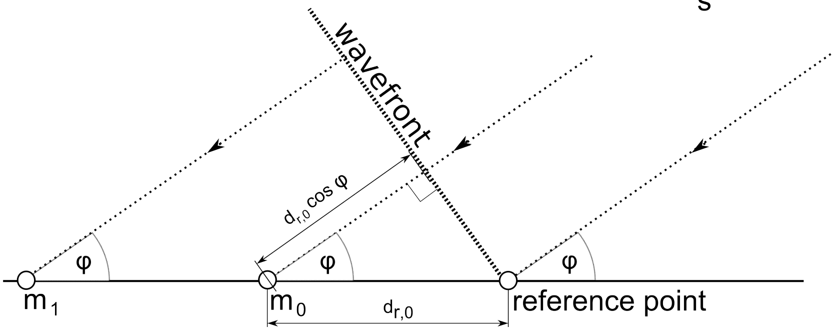


Figure 3.3: A signal from a distant source arriving at a microphone array.

$$\Delta_{r,n}(\phi) = \frac{d_{r,n} \cdot \cos(\phi)}{c} \,. \tag{3.6}$$

Substituting this into Eq. (3.4) we obtain the delays with respect to the source

$$\delta_{s,n}(\phi) = \frac{d_{r,n} \cdot \cos(\phi)}{c} + \delta_{s,r} \,. \tag{3.7}$$

In order to emphasise the dependence of the delays on ϕ, we refer to them as $\delta_{r,n}(\phi)$ and $\delta_{s,n}(\phi)$ from now on. Note that in order to reconstruct the source signal, we need to set the delays τ_n we applied in Equation (3.1) so that they cancel out the propagation delays for the DOA "target" angle ϕ_t corresponding to the source location. This means that τ_n is dependent on ϕ_t and is given by

$$\tau_n(\phi_t) = \frac{d_{r,n} \cdot \cos(\phi_t)}{c} + \delta_{s,r} \,. \tag{3.8}$$

The angle ϕ_t is called the *steering angle*, and the corresponding "target" direction the *look direction* of the array.

3.1.2 Array Frequency Response and Beam Pattern

Let us now examine how the output of the beamformer is affected by our choice of beamformer parameters: more specifically, how the relationship between the input $s(t)$ and the output $y(t)$ varies as a function of the delays $\tau_n(\phi_t)$ that we choose for our beamformer.

We begin in the time domain by putting together Equations (3.1) and (3.2), so we can express the DSB output as a linear combination of shifted versions of the source signal:

$$y(t) = \frac{1}{N} \sum_{n=0}^{N-1} s(t + \tau_n(\phi_t) - \delta_{s,n}(\phi)) \,. \tag{3.9}$$

Since a shift by a units in the time domain corresponds to the scaling of the spectrum by a complex multiplicative factor of $e^{j\omega a}$ in the frequency domain (cf. Sec. 2.2.2), it is convenient to express the last relationship in the frequency domain. Making use of the mentioned time shifting property and the linearity of the Fourier transform, the spectrum of the output can be expressed as

$$Y(\omega) = \frac{1}{N} \sum_{n=0}^{N-1} e^{j\omega(\tau_n(\phi_t) - \delta_{s,n}(\phi))} \cdot S(\omega) \,, \tag{3.10}$$

where $S(\omega)$ is the spectrum of the source signal $s(t)$. We will now develop the *array frequency response function*, which plays the same role for an array as the frequency response does for an LTI filter. If we assume that the transformation which the source signal undergoes when travelling through the room and being processed by the microphone array is an LTI system, we can fully characterise that system by specifying its frequency response, which is defined as the ratio of the output to the input spectrum. Given our single-input single-output simplification, this is

$$H(\omega) = \frac{Y(\omega)}{S(\omega)}\,. \tag{3.11}$$

To obtain the array frequency response for the DSB under our simple delay propagation model of the relationship between source and microphone signal, we substitute (3.10) into (3.11):

$$H(\omega, \phi) = \frac{1}{N} \sum_{n=0}^{N-1} e^{j\omega(\tau_n(\phi_t) - \delta_{s,n}(\phi))}\,. \tag{3.12}$$

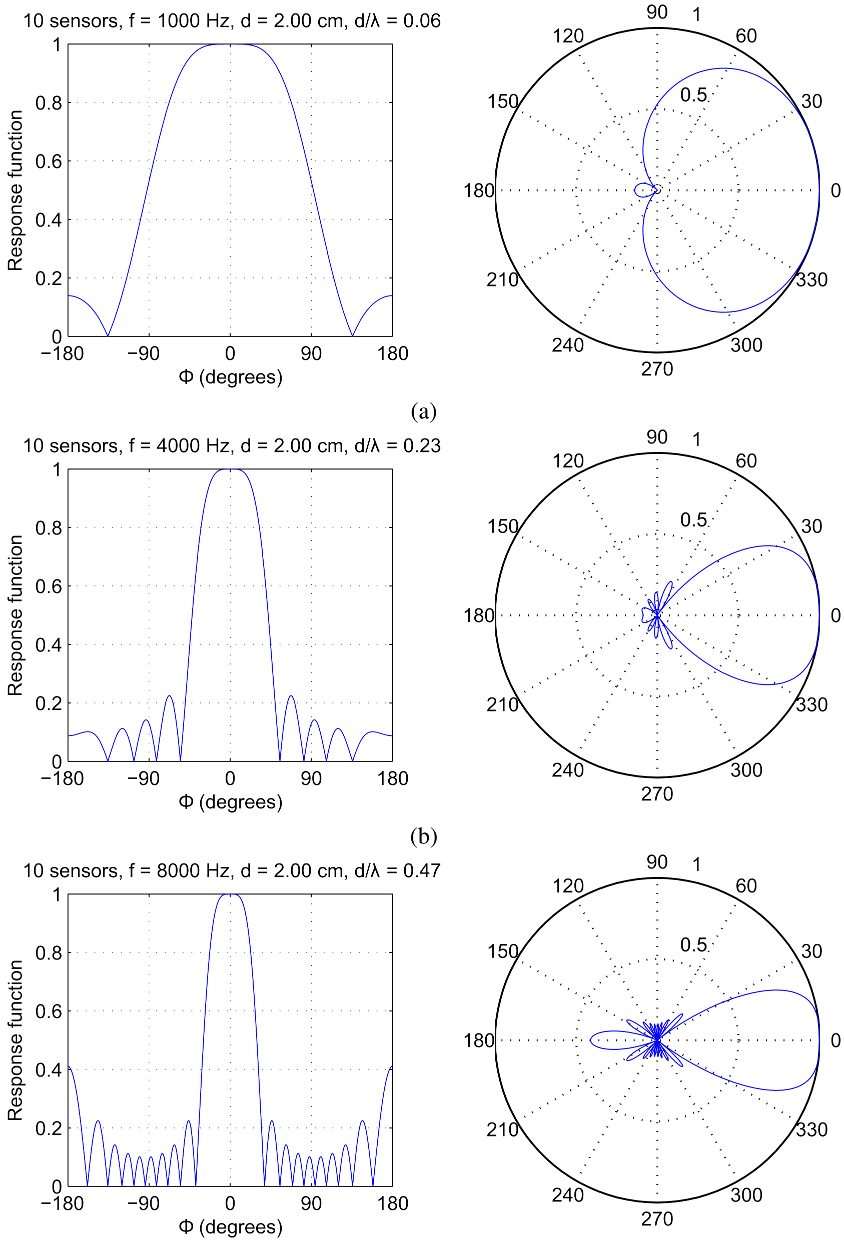
With the far-field assumption, we can express the delays in terms of the DOA angle ϕ and the distance $d_{r,n}$ from the first microphone, as in (3.7):

$$\begin{aligned} H(\omega, \phi) =& \frac{1}{N} \sum_{n=0}^{N-1} e^{j\omega(d_{r,n} \cos(\phi_t)/c + \delta_{s,r} - (d_{r,n} \cos(\phi)/c + \delta_{s,r}))} \\ =& \frac{1}{N} \sum_{n=0}^{N-1} e^{j\omega d_{r,n}(\cos(\phi_t) - \cos(\phi))/c}. \end{aligned} \tag{3.13}$$

We observe that $H(\omega, \phi)$ depends on five variables: the number of sensors N, the array geometry specified by the microphone distances $d_{r,n}$, the fixed target angle ϕ_t as well as the varying angle of an incident wave ϕ, and finally the frequency ω of the component wave of the input signal. For a given choice of all of these parameters, $H(\omega, \phi)$ gives us a complex weight that models how the part of the input spectrum corresponding to a signal component of frequency ω is scaled by the beamformer. To visualise this, the magnitude of the frequency response $|H(\omega, \phi)|$ is generally used to plot the array frequency response, and this is called the *beam pattern*, or *spatial directivity pattern*[1]. Before we examine some example beam patterns, it is worth pointing out again that the array frequency response and therefore the beam pattern are only simplified models of the real transformations sound waves undergo when being processed by the array. For example, the actual room and array system may contain non-linearities, and for Eq. (3.13) we have additionally made the simplifying far-field and simple delay propagation assumptions. A beam pattern is only as accurate as the data model it is based on; however, in many cases such simplifying assumptions are accurate enough to obtain a reasonably good model and enhance our understanding of the process.

To plot the pattern we fix all parameters except ϕ so we can illustrate the effect of an incident wave from various DOA angles on the beamformer output. For

[1] Some authors use the terms beam pattern, directivity pattern and array frequency response interchangeably. In this thesis the use of "pattern" refers to the visual representation of the magnitude of the array frequency response only.



(a)

(b)

(c)

Figure 3.4: DSB beam patterns for uniform linear array of 10 microphones with an inter-microphone spacing of 2cm and a steering angle ϕ_t of 0°.

example, let us assume a uniformly spaced linear array with 10 microphones, an inter-sensor spacing d of 2 cm and a steering angle of $\phi_t = 0°$. For such a *uniform linear array* (ULA) the distance from the first reference microphone m_0 is a multiple of the inter-sensor spacing d:

$$d_{r,n} = n \cdot d\,. \tag{3.14}$$

Substituting this into (3.13), we can simplify the formula for the beamformer response of a ULA:

$$\begin{aligned} H(\omega, \phi) &= \frac{1}{N} \sum_{n=0}^{N-1} e^{j\omega n d(\cos(\phi_t) - \cos(\phi))/c} && (3.15) \\ &= \frac{1}{N}\, e^{-j(N-1)\alpha}\, \frac{\sin(N\alpha)}{\sin(\alpha)}, && (3.16) \end{aligned}$$

with

$$\alpha = \omega d(\cos(\phi_t) - \cos(\phi))/2c\,. \tag{3.17}$$

For a proof of the equivalence of (3.15) and (3.16) see Appendix A.

Since the magnitude of the term $e^{-j(N-1)\alpha}$ in (3.16) is unity, the beam pattern is given by the following function:

$$|H(\omega, \phi)| = \frac{1}{N} \left| \frac{\sin(N\alpha)}{\sin(\alpha)} \right|\,. \tag{3.18}$$

Figure 3.4 shows three DSB beam patterns that plot $|H(\omega, \phi)|$ for the example array as a function of ϕ, with each subplot illustrating the effect on the input spectrum component of a specific frequency (1000, 4000 or 8000 Hz). The beam pattern is displayed in two ways: as a linear plot to the left and a polar plot to the right. The information is the same, the polar plot simply emphasises the correspondence of the angle ϕ to a physical direction by plotting the angle around a circle.

The plots illustrate a number of issues. First of all, the beamformer behaves in the expected way: it emphasises signals arriving from 0° as compared to signals coming from other directions, which are attenuated. At the steering angle the response is highest and it assumes a value of unity, which means that the spectral component's magnitude is neither amplified nor attenuated by the beamformer. The resulting peak, the global maximum, can be seen as a "lobe" or "beam" shape in the two lower polar plots, and this is the origin of the name beamforming. We observe that for higher frequencies, the beams become narrower.

Secondly, the beam patterns show local maxima at angles that are not the steering angle. These so-called *sidelobes* are still quite small in this example, but under certain conditions they can become undesirably large, and we will now see some examples of this effect. Figures 3.5 and 3.6 show six DSB beam patterns, calculated for a uniformly spaced linear array with the same number of sensors and steering angle as before. This time the distance between the microphones d is varied from 4.29 cm to 17.15 cm, and the frequency response is shown for 1000 Hz and 2000 Hz. In the cases 3.5(c), 3.6(b) and 3.6(c) the beam patterns have sidelobes of the same height as the main lobe. This situation is not desirable, since it means that sounds coming from these directions will not be attenuated. Such lobes with unit gain, a frequency response value of unity, are called *grating lobes*. Their appearance is due to an effect called *spatial aliasing*. In conventional signal processing the term *aliasing* is used to describe the phenomenon of being unable to distinguish certain frequencies when sampling a continuous signal at a rate that is too low with respect to the period of the signal. Microphone arrays can be viewed as performing spatial sampling of the signal, and spatial aliasing is the analogous problem of being unable to distinguish certain directions of arrival. It occurs when the inter-microphone spacing is too large with respect to the wavelength of the incident wave. The wavelength λ is the distance a wave travels in one cycle, and is calculated as the speed of sound divided by the frequency f:

$$\lambda = c/f\,. \tag{3.19}$$

In order to avoid spatial aliasing, the distance d between microphones must satisfy

$$d/\lambda < 1/2\,. \tag{3.20}$$

In fact, for a ULA the ratio $d/\lambda = d \cdot f/c$ completely determines the effect of frequency and inter-sensor spacing on the beampattern, as the plots demonstrate: the beam pattern pairs with the same ratio, for example 3.6(a) and 3.5(b), or 3.6(b) and 3.5(c), are identical. This fact can also be derived from Eq. (3.17), since a given ratio $\omega d/c$ leaves the term α dependent on the DOA and target angle only.

3.1.3 Discrete-time Version

So far we have expressed all equations in this chapter in terms of the continuous time variable t in order to simplify the presentation. However, since implementing beamforming on a computer requires data to be represented as discrete values, the signals need to be expressed in terms of the discrete sample variable k. It is assumed that all signals are sampled with the sampling rate f_s Hz (samples per second), i.e. a measurement is taken every $1/f_s$ seconds.

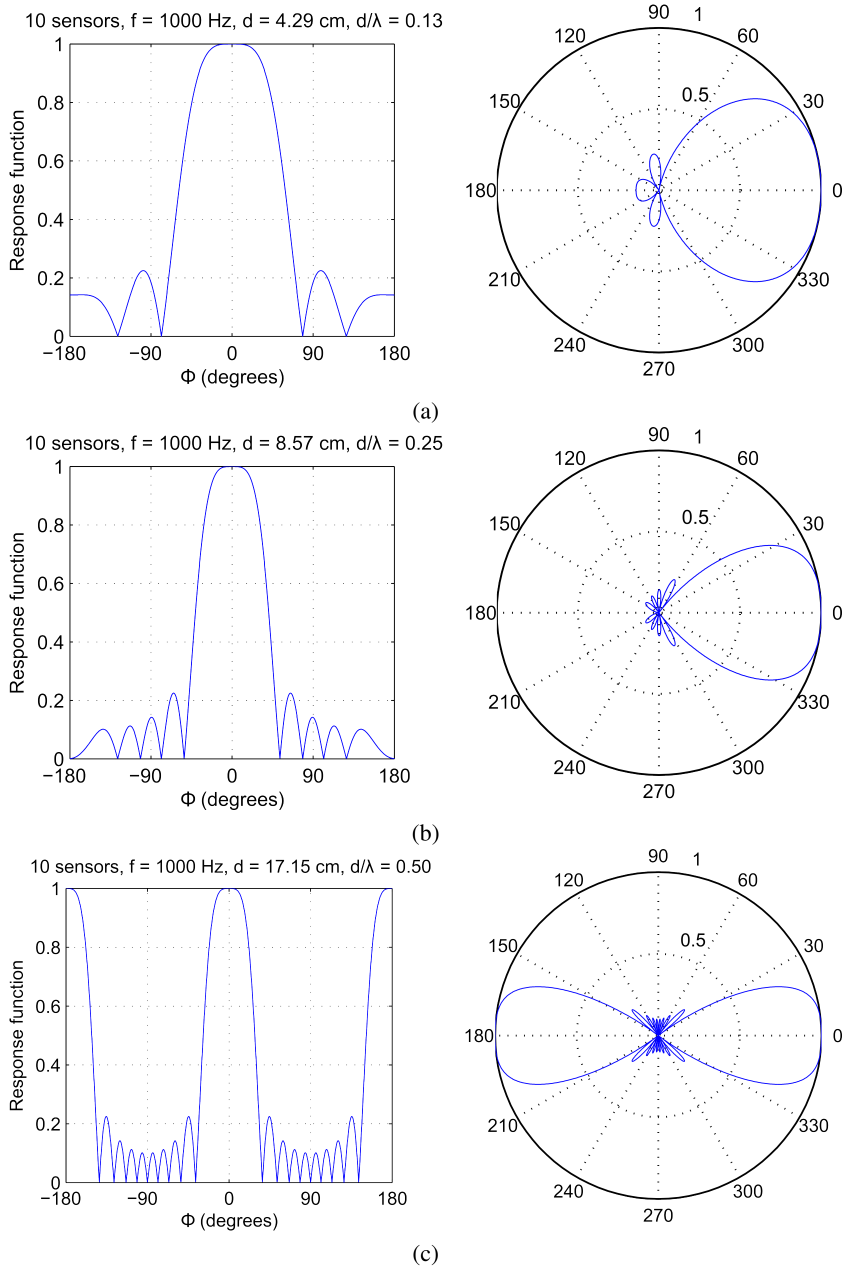
The discrete version of the DSB equation (3.1) is then

$$y[k] = \frac{1}{N} \sum_{n=0}^{N-1} x_n[k + \tau_n]\,, \tag{3.21}$$

where now τ_n is the delay in number of samples, calculated by

$$\tau_n = \left(\frac{d_{r,n} \cdot \cos(\phi_t)}{c} + \delta_{s,r} \right) \cdot f_s\,. \tag{3.22}$$

We are using the same symbol for the delays in seconds as the delays in number of samples in this thesis as the context will always make clear which one is meant. In

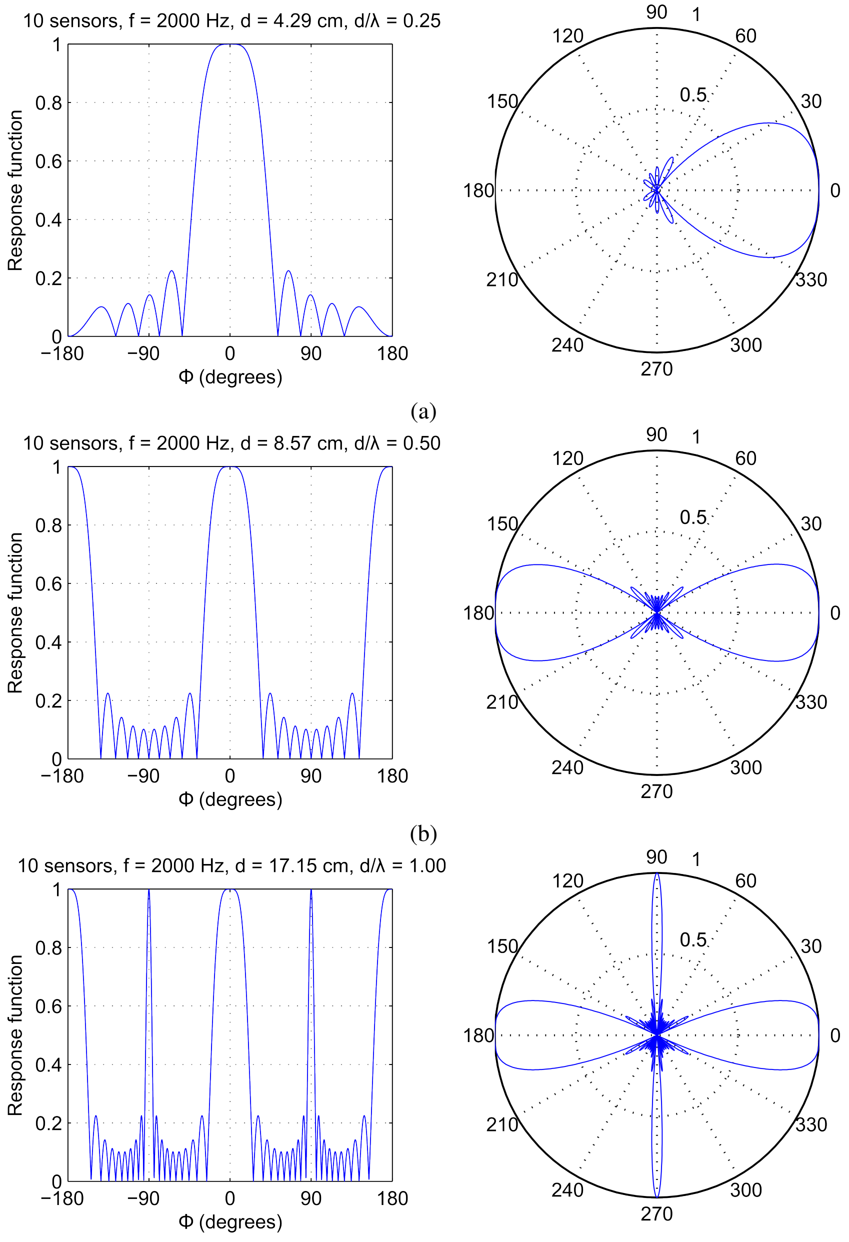


(a)

(b)

(c)

Figure 3.5: DSB beam patterns for a uniform linear array of 10 microphones with a steering angle $\phi_t = 0°$, $f = 1000$ Hz, and varying microphone distances d.



(a)

(b)

(c)

Figure 3.6: DSB beam patterns for a uniform linear array of 10 microphones with a steering angle $\phi_t = 0°$, $f = 2000$ Hz, and varying microphone distances d.

fact, a direct implementation of (3.21) is problematic since the delays would have to be integers. For example, at the typical sampling frequency of 16 kHz and a distance of 2 cm from the reference point, the largest delay (coming from $\phi_t = 0°$) with respect to the reference point would be

$$\begin{aligned}\tau_n &= \left(\frac{0.02m \cdot \cos(0°)}{343m/s}\right) \cdot 16000 \text{ samples}/s \\ &= 0.93 \text{ samples.}\end{aligned}$$

All other angles would lead to even smaller delays, so a reasonable delay in number of samples cannot be implemented in this case. We will shortly develop beamforming in the frequency domain, which does not suffer from this problem. However, for explanatory purposes we will continue in the time domain for the moment.

In the current representation the delays are added, since they are meant to counteract the propagation delays, as mentioned previously. If τ_n is positive, this means we would need access to future samples of the input signals with respect to the current output counter k. However, note that by accepting a delayed output sample, we can ensure the use of non-positive delays, i.e. only access current or past samples. Shifting the output by τ_{max}, the longest delay of all τ_n, we obtain

$$y'[k] = y[k - \tau_{\text{max}}] = \frac{1}{N} \sum_{n=0}^{N-1} x_n[k + \tau_n - \tau_{\text{max}}] \,. \tag{3.23}$$

Since $\tau_n - \tau_{\text{max}}$ is always less than or equal to zero, we can define the nonnegative values $\tau_n' = \tau_{\text{max}} - \tau_n$. The DSB can then be expressed as

$$y'[k] = \frac{1}{N} \sum_{n=0}^{N-1} x_n[k - \tau_n'] \,,\ \forall n : \tau_n' \geq 0 \,. \tag{3.24}$$

We can think of the representation in (3.24) as follows: for the current output sample $y'[k]$, each sensor contributes one weighted input sample, either the current one (if $\tau_n^{'} = 0$) or a previous sample, depending on $\tau_n^{'}$. Based on this observation we will now generalise our beamformer description to account for other beamforming types than the DSB.

3.2 A General Description: Filter-and-sum Beamformer

In Equation (3.24) each channel is multiplied with an equal weight of $1/N$. We can express a more general form of beamforming by allowing an *individual* weight per microphone:

$$y[k] = \sum_{n=0}^{N-1} a_n x_n[k - \tau_n] \,. \tag{3.25}$$

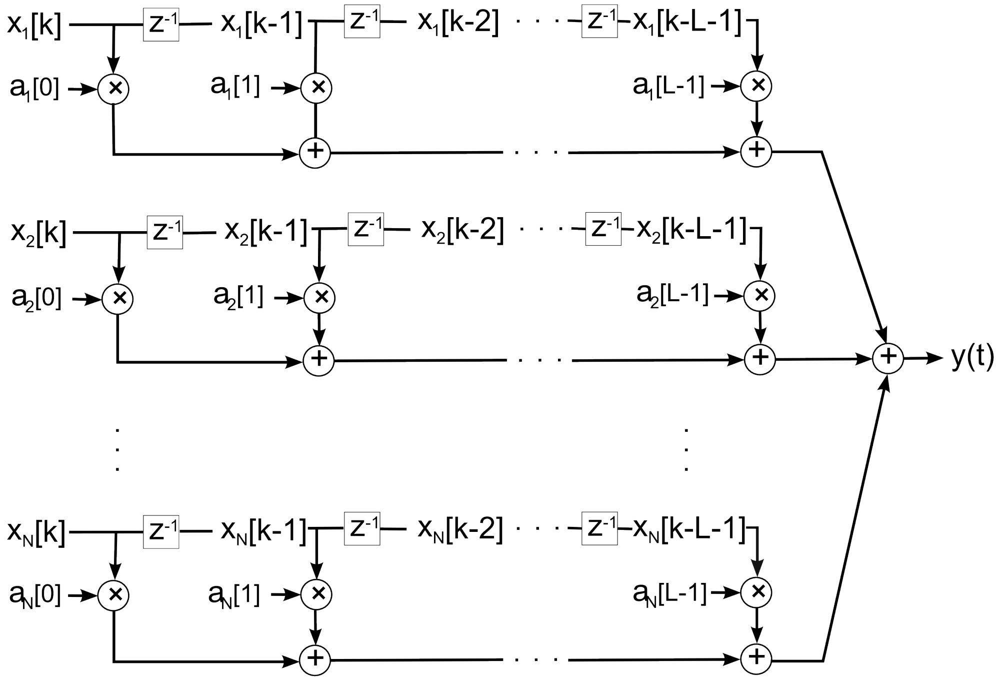


Figure 3.7: The filter-and-sum beamformer in the time domain.

For each sensor, the resulting beamformer associates a real-valued weight with one input sample, either the current or a previous one. We can generalise the operation even further by associating one weight with *each* of the last L inputs at every sensor and adding them, obtaining the structure depicted in Figure 3.7. The z^{-1} operation in this diagram denotes a delay by one sample. Formally, the output can be expressed as

$$y[k] = \sum_{n=0}^{N-1} \sum_{l=0}^{L-1} a_n[l] \cdot x_n[k-l]\,, \tag{3.26}$$

where $a_n[l]$ is the weight associated with the $(k-l)$th sample of the nth microphone. Note that the inner sum in Eq. (3.26) involves the convolution operation defined in Eq. (2.9). The above structure therefore corresponds to an LTI filter of length L applied to each sensor, and it is correspondingly called the *filter-and-sum beamformer (FSB)*. Using the convolution symbol, the FSB can be expressed as

$$y[k] = \sum_{n=0}^{N-1} a_n[k] * x_n[k]\,. \tag{3.27}$$

The sequence a_n is the impulse response of the FIR filter associated with the nth microphone, and the structure of Figure 3.7 is known as a *tapped delay line*. Observe that the DSB given in (3.24) is a special case of the FSB, with $a_n[\tau_n] = 1/N$ and all other weights set to zero. The FSB is important because it is a very general beamforming structure; in fact, all the beamformers presented in this thesis can be described as a filter-and-sum beamformer.

The FSB has been presented in the time domain, and a corresponding expression for it will now be developed in the frequency domain, where beamforming is generally done. As shown in Eq. (2.10), convolution of two signals in the time domain is equivalent to the multiplication of the spectra of the signals in the frequency domain. Making use of this fact and of the linearity of the Fourier transform again, the FSB in the frequency domain can then be specified by

$$Y(m) = \sum_{n=0}^{N-1} A_n(m) \cdot X_n(m) \,. \tag{3.28}$$

Here m is the frequency bin, and $A_n(m)$ is the frequency response of the filter associated with the nth sensor, i.e. $a_n \leftrightarrow A_n$ and similarly $x_n \leftrightarrow X_n$ are transform pairs as defined in Section 2.2.2. It is customary to express the beamformer equations using the complex conjugate of the weight A_n, so we will use the following notation from now on:

$$Y(m) = \sum_{n=0}^{N-1} W_n^*(m) \cdot X_n(m) \,, \tag{3.29}$$

where

$$W_n(m) = A_n^*(m) \,. \tag{3.30}$$

3.2.1 Beamforming in the Subband Domain

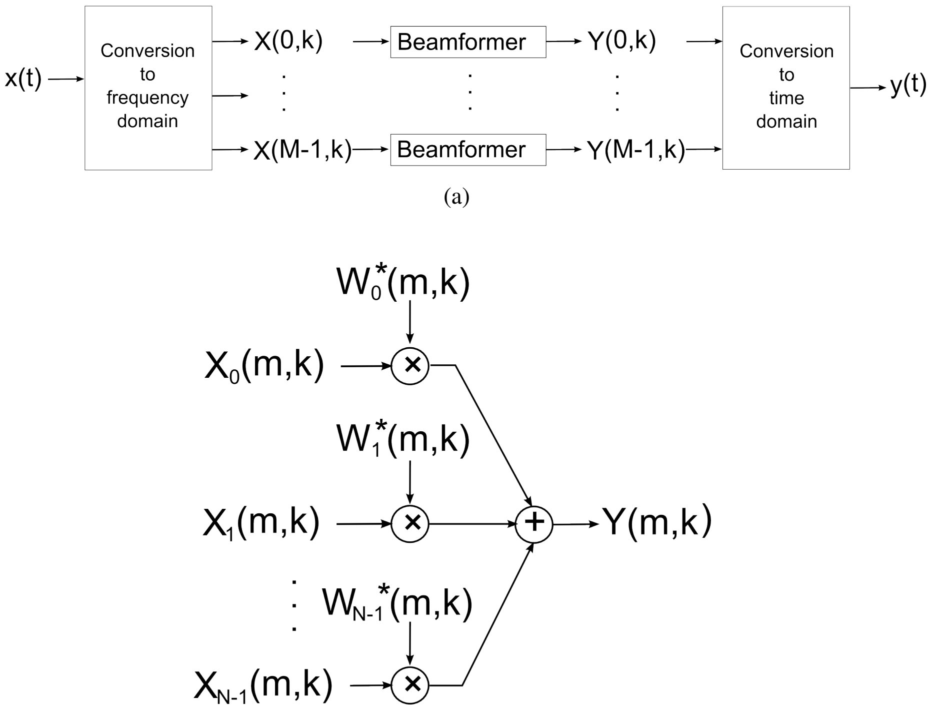
Beamforming can be performed in the time domain or in the frequency domain. Equation (3.29) shows us how to implement it in the latter:

1. The input is first converted to the frequency domain, resulting in M frequency bands or *subbands* with N complex values each per frame k. Since the subbands are approximately uncorrelated, the beamformer weights can be optimised for each subband independently. For this reason a subband-domain beamformer is much more efficient in terms of computation time than a time-domain filter-and-sum beamformer.

2. The subband samples of one bin per frame are called *snapshots*. Each subband bin m is processed separately by a narrowband beamformer, which multiplies each of its N components with a complex weight $W_n^*(m)$ and then sums over all components to obtain the output's spectrum.

3. The output spectrum $Y(m)$ can then be converted back to the time domain if needed.

Figure 3.8 illustrates the FSB in the frequency domain, with a high-level overview in part (a), and the details of one of the narrowband beamformers given in part (b). Note that the conversions between time and frequency domain in both directions cannot be performed by a standard (inverse) DFT in this case due to a problem called circular aliasing. Alternatives are the application of the *overlap-save* or



(a)

(b)

Figure 3.8: The filter-and-sum beamformer in the frequency domain. (a) High-level view, based on p. 335 of [Van Trees, 2002]. (b) Details of a subband beamformer.

overlap-add methods, or the use of a filter bank; more details on this issue are given by [Wölfel and McDonough, 2009, §3.2].

Defining the vectors

$$\boldsymbol{W}(m,k) = \begin{bmatrix} W_0(m,k) \\ W_1(m,k) \\ \vdots \\ W_{N-1}(m,k) \end{bmatrix} \tag{3.31}$$

and

$$\boldsymbol{X}(m,k) = \begin{bmatrix} X_0(m,k) \\ X_1(m,k) \\ \vdots \\ X_{N-1}(m,k) \end{bmatrix}, \tag{3.32}$$

we can express Equation (3.29) in vector notation as

$$Y(m,k) = \boldsymbol{W}^H(m,k,\phi)\boldsymbol{X}(m,k)\,, \tag{3.33}$$

where the Hermitian operator $()^H$ denotes conjugated transposition. Where convenient we may suppress the subband subscript m and the dependency of $\boldsymbol{W}$ and $\boldsymbol{X}$ on the snapshot number k and the frequency m or ω altogether. In short form, Equation (3.33) is then reduced to

$$Y = \boldsymbol{W}^H\boldsymbol{X}\,. \tag{3.34}$$

3.2.2 Array Frequency Response in Terms of the Complex Weight Vector

We will now derive an expression of the array frequency response function $H(m,\phi)$ for the FSB in terms of the frequency domain weight vector $\boldsymbol{W}$. Substituting (3.29) into the definition of $H(m,\phi)$ (3.11), we obtain

$$\begin{aligned} H(m,\phi) &= Y(m)/S(m) && (3.35)\\ &= \frac{\sum_{n=0}^{N-1} W_n^*(m)\cdot X_n(m)}{S(m)} \\ &= \sum_{n=0}^{N-1} W_n^*(m)\cdot\frac{X_n(m)}{S(m)}\,. && (3.36)\end{aligned}$$

In order to simplify this equation, let us model each x_n as a delayed version of the source signal, as done in Eq. (3.2) for a continuous signal. Denoting the delay from source to microphone n for a given DOA ϕ in samples by $\delta_{s,n}(\phi)$, the discrete version is

$$x_n[k] = s[k - \delta_{s,n}(\phi)]\,.$$

In the frequency domain, this translates to

$$X_n(m) = e^{-j(2\pi m/M)\delta_{s,n}(\phi)} S(m)\,, \tag{3.37}$$

leading to

$$\frac{X_n(m)}{S(m)} = e^{-j(2\pi m/M)\delta_{s,n}(\phi)}\,. \tag{3.38}$$

Let us further define the *array manifold vector* $\boldsymbol{d}(m, \phi)$, also known as the steering vector or the direction vector [Van Veen and Buckley, 1988], as

$$\boldsymbol{d}(m,\phi) = \begin{bmatrix} e^{-j(2\pi m/M)\delta_{s,0}(\phi)} \\ e^{-j(2\pi m/M)\delta_{s,1}(\phi)} \\ \vdots \\ e^{-j(2\pi m/M)\delta_{s,N-1}(\phi)} \end{bmatrix}. \tag{3.39}$$

As before, we may suppress the dependence of this vector on m and ϕ where convenient and simply use $\boldsymbol{d}$ in the notation. In vector form the microphone signals for a plane wave from angle ϕ are then given by

$$\boldsymbol{X}(m) = \boldsymbol{d}(m,\phi) S(m)\,. \tag{3.40}$$

Substituting (3.38) into (3.36) and using $\boldsymbol{d}$, we can express $H(m, \phi)$ for the filter-and-sum beamformer as

$$H(m,\phi) = \sum_{n=0}^{N-1} W_n^*(m) \cdot e^{-j(2\pi m/M)\delta_{s,n}(\phi)} = \boldsymbol{W}^H(m,\phi)\boldsymbol{d}(m,\phi)\,. \tag{3.41}$$

The array frequency response $H(m, \phi)$ models the overall effect of a signal passing through the microphone array and being processed by a filter-and-sum beamformer, where the array manifold vector $\boldsymbol{d}$ accounts for the interaction between the array geometry and incident plane waves, and the $\boldsymbol{W}$ term describes the effects of the chosen beamformer filter weights.

Note that in the above notation, the DSB array frequency response of Eq. (3.12) can be expressed as

$$\begin{aligned} H(m,\phi) &= \frac{1}{N}\sum_{n=0}^{N-1} e^{j(2\pi m/M)(\tau_n(\phi_t)-\delta_{s,n}(\phi))} \\ &= \frac{1}{N}\boldsymbol{d}^H(m,\phi_t)\,\boldsymbol{d}(m,\phi)\,, \end{aligned} \tag{3.42}$$

and the complex weight vector for the DSB is therefore

$$\boldsymbol{W}^H_{\text{DSB}} = \frac{1}{N}\boldsymbol{d}^H(m,\phi_t)\,. \tag{3.43}$$

3.3 Beamformer Categories

We have seen that the array frequency response is influenced both by the array geometry and the beamformer weights. Beamformer design therefore consists of finding a geometry and weight vector $\boldsymbol{W}$ which result in an array frequency response $H(m, \phi)$ with desirable qualities. In this thesis we will focus on design by weight selection, which has the advantage of not requiring a physical change in array geometry. There are many different ways of determining $\boldsymbol{W}$, and beamformers can be classified according to how the weights are chosen. Commonly used categories include the following:

1. *Data independent* beamformers:
   The weights of beamformers in this category, which are sometimes also called *fixed* beamformers, do not depend on the array data. Rather, they are chosen to yield a specified response [Van Veen and Buckley, 1988]. An example of this class is the delay-and-sum beamformer. While its parameters depend on the steering angle, they do not depend on the array data as such. Another example are super-directive beamformers [Brandstein and Ward, 2001].

2. *Statistically optimum* beamformers:
   In this case the beamformer weights are chosen based on the statistics of the array data, in order to optimise the array response with respect to a certain criterion. A statistically optimum beamformer is *adaptive* if the data characteristics have to estimated from input data as it comes in, either recursively or non-recursively. It is *non-adaptive* if the characteristics are known from the start. However, since this situation is rather rare, most statistically optimum beamformers are adaptive, and some people therefore use the terms "statistically optimum beamformers" and "adaptive beamformers" interchangeably.

The remainder of this chapter will present some statistically optimum beamformers, which may be implemented as adaptive or non-adaptive versions. The next chapter will then present the basis of the experimental part of this thesis: the maximum negentropy beamformer, which is a statistically optimum adaptive beamformer.

3.4 Signal Model With Additive Noise

In order to introduce the beamformers in the remainder of this chapter, we need to modify our simple signal model of Equation (3.2) to account for the presence of additive noise. From now on, let us assume that our input subband snapshots at the nth microphone consist of a signal component X_{Sn} and an additional additive noise component X_{Nn}:

$$X_n(m) = X_{Sn}(m) + X_{Nn}(m)\,. \tag{3.44}$$

Depending on how we model X_{Nn}, this noise could represent e.g. uncorrelated noise present at each microphone noise, but also interference signals arriving from other directions than the look direction.
For the whole array we can express the new model with the N-dimensional vectors

$$\boldsymbol{X}(m) = \begin{bmatrix} X_{S0}(m,k) \\ X_{S1}(m,k) \\ \vdots \\ X_{SN-1}(m,k) \end{bmatrix} + \begin{bmatrix} X_{N0}(m,k) \\ X_{N1}(m,k) \\ \vdots \\ X_{NN-1}(m,k) \end{bmatrix} = \boldsymbol{X_S}(m) + \boldsymbol{X_N}(m)\,. \tag{3.45}$$

With the simple propagation delay model and far-field assumption, each subband snapshot can be modelled by

$$X_n(m) = e^{-j(2\pi m/M)\delta_{s,n}(\phi)} S(m) + X_{Nn}(m)\,, \tag{3.46}$$

which implies

$$\boldsymbol{X_S}(m) = \boldsymbol{d}(m,\phi)S(m) \tag{3.47}$$

and

$$\boldsymbol{X}(m) = \boldsymbol{d}(m,\phi)S(m) + \boldsymbol{X_N}(m)\,. \tag{3.48}$$

Furthermore, we will model our data with random variables. We assume the noise and signal component of the subband snapshots to be zero-mean and uncorrelated. Since we will make use of the variance and covariance matrices of our data in several beamformer derivations, we will now define these statistics for the beamformer input and output.

Substituting (3.44) into (3.33), the output of the filter-and-sum beamformer is given by

$$\begin{aligned} Y(m) &= \boldsymbol{W}^H(m)\boldsymbol{X}(m) \\ &= \boldsymbol{W}^H(m)(\boldsymbol{X_S}(m) + \boldsymbol{X_N}(m)) \\ &= (\boldsymbol{W}^H(m)\boldsymbol{X_S}(m)) + (\boldsymbol{W}^H(m)\boldsymbol{X_N}(m)) \\ &= Y_S(m) + Y_N(m)\,, \end{aligned} \tag{3.49}$$

where we define $Y_S(m) = \boldsymbol{W}^H(m)\boldsymbol{X_S}(m)$ as the signal component in the beamformer output and $Y_N(m) = \boldsymbol{W}^H(m)\boldsymbol{X_N}(m)$ as the noise component. Omitting the dependence on m for clarity, the output variance is then

$$\begin{aligned} \sigma_Y^2 &= E\{|Y|^2\} = E\{|Y_S + Y_N|^2\} = E\{|\boldsymbol{W}^H\boldsymbol{X_S} + \boldsymbol{W}^H\boldsymbol{X_N}|^2\} \\ &= E\{\boldsymbol{W}^H\boldsymbol{X_S}\boldsymbol{X_S}^H\boldsymbol{W} + \boldsymbol{W}^H\boldsymbol{X_S}\boldsymbol{X_N}^H\boldsymbol{W} \\ &\quad +\boldsymbol{W}^H\boldsymbol{X_N}\boldsymbol{X_S}^H\boldsymbol{W} + \boldsymbol{W}^H\boldsymbol{X_N}\boldsymbol{X_N}^H\boldsymbol{W}\} \\ &= \boldsymbol{W}^H E\{\boldsymbol{X_S}\boldsymbol{X_S}^H\}\boldsymbol{W} + \boldsymbol{W}^H E\{\boldsymbol{X_S}\boldsymbol{X_N}^H\}\boldsymbol{W} \\ &\quad +\boldsymbol{W}^H E\{\boldsymbol{X_N}\boldsymbol{X_S}^H\}\boldsymbol{W} + \boldsymbol{W}^H E\{\boldsymbol{X_N}\boldsymbol{X_N}^H\}\boldsymbol{W} \\ &= \boldsymbol{W}^H E\{\boldsymbol{X_S}\boldsymbol{X_S}^H\}\boldsymbol{W} + \boldsymbol{W}^H E\{\boldsymbol{X_N}\boldsymbol{X_N}^H\}\boldsymbol{W} \\ &= \boldsymbol{W}^H\boldsymbol{P_{X_S}}\boldsymbol{W} + \boldsymbol{W}^H\boldsymbol{P_{X_N}}\boldsymbol{W}\,, \end{aligned} \tag{3.50}$$

where $\boldsymbol{P}_{\boldsymbol{X_S}}$ and $\boldsymbol{P}_{\boldsymbol{X_N}}$ are the PSD matrices (cf. Sec. 2.2.4) of the noise and signal components received at the complete array. Note that the cross-spectral matrices $E\{\boldsymbol{X_S}\boldsymbol{X_N}^H\}$ and $E\{\boldsymbol{X_N}\boldsymbol{X_S}^H\}$ are zero due to our assumption that noise and signal are uncorrelated. Defining $\sigma^2_{Y_S} = \boldsymbol{W}^H\boldsymbol{P}_{\boldsymbol{X_S}}\boldsymbol{W}$ and $\sigma^2_{Y_N} = \boldsymbol{W}^H\boldsymbol{P}_{\boldsymbol{X_N}}\boldsymbol{W}$ as the signal and noise components of the beamformer output respectively, we can express the variance of the output more compactly as

$$\sigma^2_Y = \sigma^2_{Y_S} + \sigma^2_{Y_N} \, . \tag{3.51}$$

Using Eq. (3.47) we can further specify the PSD matrix of the signal components at the array as

$$\begin{aligned}
\boldsymbol{P}_{\boldsymbol{X_S}}(m) &= E\{\boldsymbol{X_S}\boldsymbol{X_S}^H\} \\
&= E\{S(m)\boldsymbol{d}(m,\phi_t)\boldsymbol{d}^H(m,\phi_t)\, S^*(m)\} \\
&= E\{S(m)S^*(m)\}\boldsymbol{d}(m,\phi_t)\boldsymbol{d}^H(m,\phi_t) \\
&= \sigma^2_S(m)\boldsymbol{d}(m,\phi_t)\boldsymbol{d}^H(m,\phi_t) \, .
\end{aligned} \tag{3.52}$$

The following sections will utilise these definitions to present a number of well-known optimisation criteria in beamformer design.

3.5 SNR as Quality Measure and Optimisation Criterion

The first statistically optimum beamformer to be presented in this chapter is the maximum signal-to-noise ratio beamformer, which assumes that we have some way of measuring the average noise energy present at the array. If that is the case, we can derive a weight vector that will minimise the ratio of signal to noise energy.

3.5.1 Power of a Signal and Signal-to-noise Ratio

A common quality measurement for signals is the *signal-to-noise ratio (SNR)*, which is defined as the ratio of the signal power to the power of the noise corrupting the signal:

$$\text{SNR(noisy signal)} = \frac{P_{\text{signal}}}{P_{\text{noise}}} \, . \tag{3.53}$$

The SNR is often expressed in logarithmic terms and measured in decibels (dB), in which case it is defined as

$$\text{SNR}_{dB}\text{(noisy signal)} = 10\log_{10}\frac{P_{\text{signal}}}{P_{\text{noise}}} \, . \tag{3.54}$$

To judge a noise removal mechanism the improvement in SNR of the output compared to the input is therefore often measured. For arrays we obtain such a measure by dividing the SNR at any given sensor, e.g. the first one, by the SNR at the output of the beamformer. This is called the *array gain*:

$$\text{array gain} = \frac{\text{SNR}_{\text{out}}}{\text{SNR}_{\text{in}}} \tag{3.55}$$

Let us now develop expressions for the input and output SNR and the array gain in terms of our data and beamforming models. In Sec. 2.2.4 we saw that the power of a zero-mean random variable is an estimate of its variance σ_x^2. In this thesis we assume that the mean of the data received at the sensors is zero; if that is not the case, we can estimate the mean in some way and subtract it. Now if the beamformer input has a mean of zero, so does the beamformer output, i.e. $\mu_y = 0$. Finally, Parseval's Theorem [Oppenheim and Schafer, 1989] states that the power of a signal in the time domain equals its power in the frequency domain. The average power of the beamformer output is therefore equal to the output subband snapshot variance σ_Y^2 as given by Equation (3.50).

We can now express the input SNR of Equation (3.53) as the quotient of the signal component's variance and the noise component's variance as defined in (2.11), both measured at one sensor, in this case the first microphone:

$$\mathrm{SNR_{in}} = \frac{\sigma^2_{X_{S0}}}{\sigma^2_{X_{N0}}} \,. \tag{3.56}$$

Using Eq. (2.11) and (3.51), the output SNR is given by

$$\mathrm{SNR_{out}} = \frac{\sigma^2_{Y_S}}{\sigma^2_{Y_N}} = \frac{\boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_S}} \boldsymbol{W}}{\boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_N}} \boldsymbol{W}} \,. \tag{3.57}$$

Substituting (3.56) and (3.57) into (3.55) we obtain

$$\text{array gain} = \frac{\mathrm{SNR_{out}}}{\mathrm{SNR_{in}}} = \frac{\sigma^2_{Y_S}}{\sigma^2_{Y_N}} \Bigg/ \frac{\sigma^2_{X_{S0}}}{\sigma^2_{X_{N0}}} = \frac{\boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_S}} \boldsymbol{W}}{\boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_N}} \boldsymbol{W}} \cdot \frac{\sigma^2_{X_{N0}}}{\sigma^2_{X_{S0}}} \,. \tag{3.58}$$

3.5.2 Maximum SNR beamformer

In beamforming, the SNR has not only been used as a quality measure but also as an optimisation criterion, leading to the formulation of the *maximum SNR (MSNR) beamformer*. The MSNR beamformer sets the weights so as to maximise the array gain. Since the input SNR is fixed, this amounts to maximising the output SNR as given by Eq. (3.57), i.e. the optimisation criterion is

$$\max_W \mathrm{SNR_{out}} = \max_W \frac{\boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_S}} \boldsymbol{W}}{\boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_N}} \boldsymbol{W}} \,. \tag{3.59}$$

Maximising the quotient of this criterion is equivalent to maximising its numerator while keeping its denominator equal to unity or some other constant, i.e. it is equivalent to the constrained optimisation problem

$$\max_W \boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_S}} \boldsymbol{W} \quad \text{subject to} \quad \boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_N}} \boldsymbol{W} = 1 \,. \tag{3.60}$$

This problem can be solved with the method of Lagrange multipliers, which involves introducing the Lagrange variable λ to define the following function f to be maximised:

$$f = \boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_S}} \boldsymbol{W} - \lambda(\boldsymbol{W}^H \boldsymbol{P}_{\boldsymbol{X_N}} \boldsymbol{W} - 1) \,. \tag{3.61}$$

To find the maximum of f, the derivative of this function is set equal to zero, yielding

$$\boldsymbol{P_{X_S}} \boldsymbol{W} = \lambda \boldsymbol{P_{X_N}} \boldsymbol{W} \,. \tag{3.62}$$

This problem is the so-called *generalised eigenvalue problem*, which for two given matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ involves finding the eigenvectors v that fulfil the equation

$$\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{B}\boldsymbol{v} \,. \tag{3.63}$$

If $\boldsymbol{B}$ is invertible, it is equivalent to the *standard eigenvalue problem*

$$\boldsymbol{A'}\boldsymbol{v} = [\boldsymbol{B}^{-1}\boldsymbol{A}]\boldsymbol{v} = \lambda \boldsymbol{v} \,, \tag{3.64}$$

in our case

$$[\boldsymbol{P_{X_N}^{-1}} \boldsymbol{P_{X_S}}]\boldsymbol{W} = \lambda \boldsymbol{W} \,. \tag{3.65}$$

Any eigenvector of $[\boldsymbol{P_{X_N}^{-1}} \boldsymbol{P_{X_S}}]$ will fulfil Equation (3.65), and to maximise the output SNR the one corresponding to the largest eigenvalue λ is chosen. Substituting (3.52) into (3.65) gives

$$\boldsymbol{P_{X_N}^{-1}} \, \sigma_S^2(m) \, \boldsymbol{d}(m, \phi_t) \boldsymbol{d}^H(m, \phi_t) \, \boldsymbol{W} = \lambda \boldsymbol{W} \,, \tag{3.66}$$

which has the solution [Van Trees, 2002, §6.2.3]

$$\boldsymbol{W}_{\text{MSNR}}^H = \boldsymbol{P_{X_N}^{-1}} \boldsymbol{d}(m, \phi_t) \,. \tag{3.67}$$

Comparing this to the DSB solution (3.43) we see that in addition to the target angle ϕ_t, the MSNR beamformer requires knowledge of the (inverse) noise PSD matrix. Using a voice activity detector [Beritelli et al., 2001] to identify periods of speech, the remaining noise-only periods can be used to obtain an estimate of $\boldsymbol{P_{X_N}}$ and its inverse. The VAD may introduce errors though, especially when the SNR is low and reliable voice activity detection becomes difficult [Wölfel and McDonough, 2009].

3.6 MPDR and MVDR Beamformers

There are two well-known beamformers that are closely related to the MSNR beamformer: the MVDR beamformer which will be described in this section, and the minimum mean square error beamformer to be presented in Section 3.7. The exact nature of the relationship between the three beamformers will be discussed in Section 3.8, but let us begin by deriving the definition of the MVDR beamformer. There are actually two related but slightly different definitions of the MVDR beamformer, and we will follow Van Trees [2002] in presenting both variations, referring to one as the *minimum power distortionless response (MPDR)* beamformer and the other as the MVDR beamformer, even though some textbooks define the former as the MVDR beamformer.

3.6.1 Distortionless Constraint

In Section 3.5.2 we saw an example of a constrained optimisation problem that was introduced to solve an unconstrained optimisation problem. In fact, some beamformers have been designed by formulating a constrained optimisation problem to start with, and not as an auxiliary tool. One common way of doing this is to minimise the variance of the array output subject to a so-called *distortionless constraint*: the constraint that in the absence of noise $H(m, \phi)$ should be unity for the steering angle, so that a plane wave arriving from that direction is not distorted, i.e. neither amplified nor attenuated. Formally, we can express this constraint as

$$\boldsymbol{W}^H(m, \phi)\boldsymbol{d}(m, \phi_t) = 1\,, \tag{3.68}$$

which implies (cf. (3.35) and (3.41)) that the beamformer output is the desired source signal:

$$Y(m) = S(m)\,. \tag{3.69}$$

The reason for wanting to minimise the output variance is explained in the following section.

3.6.2 Minimum Power Distortionless Response Beamformer

In order to retain only signals coming from the look direction, ideally our array frequency response would be unity for the steering angle, and zero or at least very low anywhere else. Non-zero values for angles other than the steering angle mean that signals coming from these other directions will not be fully suppressed. One way of measuring the strength of those undesirable components is to measure the average energy of the output signal, i.e. its power, and that is why some methods aim to minimise the output power while still fulfilling the distortionless constraint. Since the power of a zero-mean signal is equivalent to its variance, these approaches can also be said to minimise the variance.

We will now formalise this idea. Given the distortionless constraint in (3.68), and the aim to minimise the variance σ_Y^2 as given in Equation (3.50), we have the following constrained optimisation problem:

$$\min_{W} \boldsymbol{W}^H \boldsymbol{P_X} \boldsymbol{W} \quad \text{subject to} \quad \boldsymbol{W}^H \boldsymbol{d}(m, \phi_t) = 1 \tag{3.70}$$

As before, the method of Lagrange multipliers can be applied to this problem, resulting in the following solution for the optimal weight vector:

$$\boldsymbol{W}_{\text{MPDR}}^H = \left[\boldsymbol{d}^H(m, \phi_t)\ \boldsymbol{P_X}^{-1}\ \boldsymbol{d}(m, \phi_t)\right]^{-1}\ \boldsymbol{d}^H(m, \phi_t)\ \boldsymbol{P_X}^{-1}\,. \tag{3.71}$$

The details of the computation can be found in [Doclo, 2003] or [Wölfel and McDonough, 2009], for example.

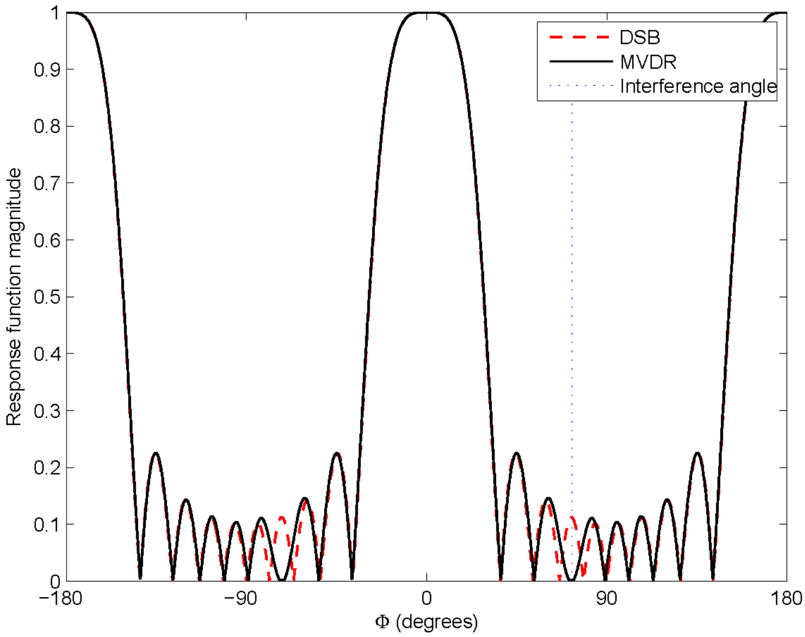


Figure 3.9: DSB and MVDR beamformer beam patterns with single plane wave interference from 72.5°.

3.6.3 Minimum Variance Distortionless Response Beamformer

If we have more specific knowledge of the type and strength of noise present in the signal received at the microphone array, we can use that knowledge to formulate a slightly different beamformer. Instead of minimising the output power, we can aim to minimise only the noise power, after all it is the noise that we would like to suppress. In that case, the optimisation problem becomes

$$\min_{W} \boldsymbol{W}^H \boldsymbol{P}_{X_N} \boldsymbol{W} \quad \text{subject to} \quad \boldsymbol{W}^H \boldsymbol{d}(m, \phi_t) = 1 \tag{3.72}$$

and the weight vector becomes

$$\boldsymbol{W}^H_{\text{MVDR}} = \left[\boldsymbol{d}^H(m,\phi_t)\, \boldsymbol{P}^{-1}_{\boldsymbol{X_N}}\, \boldsymbol{d}(m,\phi_t)\right]^{-1} \boldsymbol{d}^H(m,\phi_t)\, \boldsymbol{P}^{-1}_{\boldsymbol{X_N}} \,. \tag{3.73}$$

Unlike the DSB, the MVDR beamformer can react to noise sources directly by placing nulls in the beam pattern. We can demonstrate this for the simple case of a single plane wave interference under white noise. Denoting the spectrum of the interference wave by N_I, and assuming that the white noise is stationary, equal at all microphones and of variance σ_w^2, we can express the PSD matrix of the overall noise as

$$\boldsymbol{P}_{X_N} = \sigma_w^2 \boldsymbol{I} + N_I \boldsymbol{d}(m,\phi_I)\boldsymbol{d}^H(m,\phi_I) \,, \tag{3.74}$$

where ϕ_I is the angle of interference and $\boldsymbol{I}_N$ is the identity matrix of dimension N, the number of sensors. Starting from this definition we can derive $\boldsymbol{P}_{\boldsymbol{X_N}}^{\boldsymbol{-1}}$ and substitute this into (3.73) to obtain the MVDR weight vector; for details see [Van Trees, 2002, §6.3.1]. Using Eq. (3.41), we can then derive the beam pattern from the weight vector. Figure 3.9 plots such a beam pattern for a 2 kHz plane wave arriving at a ULA with 10 microphones 8.57 cm apart, so that $d/\lambda = 0.5$, with a steering angle of 0° and interference from 72.5°. For purposes of comparison, the DSB beam pattern is also plotted as a dashed line. We observe that the two beam patterns are very similar, except for the region around the interference angle, where the DSB has a sidelobe but the MVDR places a null, thereby achieving better interference suppression.

3.7 Minimum Mean Square Error Beamformer

We stated before that the goal of beamforming is to restore the desired source signal $s(t)$. If the desired signal were known, we could find beamformer weights which minimise the difference between it and the actual beamformer output $y(t)$. While we do not usually have access to the desired signal directly – otherwise beamforming would not be necessary – we may be able to approximate it sufficiently. The *minimum mean square error (MMSE)* filter, more commonly known as the *Wiener filter*, implements this approach by minimising the difference between the desired signal and the beamformer output in the mean square error sense. Denoting the desired signal by $d(t)$, this means that we define the error $e(t)$ as

$$e(t) = d(t) - y(t)\,, \tag{3.75}$$

and optimise the beamformer parameters to minimise the mean squared error:

$$\min_{W} E\{|e(t)|^2\}\,. \tag{3.76}$$

Assuming a zero mean again, expression (3.76) corresponds to minimising the average power in the error signal, and as before the solution is preserved when moving to the frequency domain. Assuming $e(t) \leftrightarrow E(m)$ and $d(t) \leftrightarrow D(m)$, substituting the frequency domain version of Eq. (3.75) into (3.76) and using Eq. (3.34) yields

$$\begin{aligned}
E\{|E(m)|^2\} &= E\{|D(m) - Y(m)|^2\} \\
&= E\{(D - \boldsymbol{W}^H\boldsymbol{X})(D^* - \boldsymbol{X}^H\boldsymbol{W})\} \\
&= E\{\boldsymbol{W}^H\boldsymbol{X}\boldsymbol{X}^H\boldsymbol{W} - \boldsymbol{W}^H\boldsymbol{X}D^* - \boldsymbol{X}^H\boldsymbol{W}D + DD^*\} \\
&= \boldsymbol{W}^H E\{\boldsymbol{X}\boldsymbol{X}^H\}\boldsymbol{W} - \boldsymbol{W}^H E\{\boldsymbol{X}D^*\} \\
&\quad -E\{\boldsymbol{X}^H D\}\boldsymbol{W} + E\{DD^*\} \\
&= \boldsymbol{W}^H\boldsymbol{P_X}\boldsymbol{W} - \boldsymbol{W}^H\boldsymbol{\sigma^2}_{XD} - \boldsymbol{\sigma^2}^H_{XD}\boldsymbol{W} + \sigma_D^2\,,
\end{aligned} \tag{3.77}$$

where $\boldsymbol{\sigma^2}_{XD} = E\{\boldsymbol{X}D^*\}$ and $\sigma_D^2 = E\{DD^*\}$. The MMSE optimisation criterion can therefore be stated as

$$\min_W \left(\boldsymbol{W}^H \boldsymbol{P_X} \boldsymbol{W} - \boldsymbol{W}^H \boldsymbol{\sigma^2}_{XD} - \boldsymbol{\sigma^2}^H_{XD} \boldsymbol{W} \right) + \sigma_D^2 \, . \tag{3.78}$$

By setting the complex derivative of the optimisation criterion with respect to $\boldsymbol{W}$ equal to zero we obtain the optimal weight vector [Van Trees, 2002, §6.2.2.1]

$$\boldsymbol{W}^H_{\text{MMSE}} = \boldsymbol{\sigma^2}^H_{XD} \boldsymbol{P_X^{-1}} \, , \tag{3.79}$$

an equivalent form of which is known as the Wiener-Hopf Equation [Oppenheim and Schafer, 1989]. With our noise and signal model and the plane wave assumption, the MMSE beamformer's weight vector can be specialised to

$$\boldsymbol{W}^H_{\text{MMSE}} = \frac{\sigma_S^2}{\sigma_S^2 + \Lambda(m)} \Lambda(m) \cdot \boldsymbol{d}(m, \phi_t) \boldsymbol{P}^{\boldsymbol{-1}}_{\boldsymbol{X_N}} \, , \tag{3.80}$$

where

$$\Lambda(m) = [\boldsymbol{d}^H(m, \phi_t) \, \boldsymbol{P}^{\boldsymbol{-1}}_{\boldsymbol{X_N}} \boldsymbol{d}(m, \phi_t)]^{-1} \tag{3.81}$$

The derivation of this weight vector from the more general form in (3.79) is given in Appendix B.

Since the desired signal $s(t)$ is not available in practice, it is usually estimated from the data. This can be done by assuming a model of the relationship between signal and noise, then estimating the noise in some way, and finally removing the latter from the measured noisy signal to obtain an estimate of the clean signal. For example, in the case of an additive and stationary noise assumption we can estimate the noise during a period of speech inactivity, then estimate the noisy speech signal during a period of speech activity, and by subtracting one from the other obtain an estimate of the source signal.

3.8 Comparison of MSNR, MVDR and MMSE Beamformers

A comparison of the weight vectors for the MSNR, MVDR and MMSE beamformers reveals that all three are closely related to each other:

$$\boldsymbol{W}^H_{\text{MSNR}} = \boldsymbol{d}(m, \phi_t) \boldsymbol{P}^{\boldsymbol{-1}}_{\boldsymbol{X_N}} \, ,$$

$$\begin{aligned} \boldsymbol{W}^H_{\text{MVDR}} &= \Lambda(m) \boldsymbol{d}^H(m, \phi_t) \, \boldsymbol{P}^{\boldsymbol{-1}}_{\boldsymbol{X_N}} \\ &= \Lambda(m) \boldsymbol{W}^H_{\text{MSNR}} \, , \end{aligned}$$

$$\begin{aligned}\boldsymbol{W}^H_{\text{MMSE}} &= \frac{\sigma_S^2}{\sigma_S^2+\Lambda(m)}\Lambda(m)\cdot\boldsymbol{d}(m,\phi_t)\boldsymbol{P}^{\boldsymbol{-1}}_{\boldsymbol{X_N}} \\ &= \frac{\sigma_S^2}{\sigma_S^2+\Lambda(m)}\boldsymbol{W}^H_{\text{MVDR}} \\ &= \Lambda_2(m)\boldsymbol{W}^H_{\text{MSNR}}\,,\end{aligned}$$

where $\Lambda(m)$ is defined as in Eq. (3.81), and

$$\Lambda_2(m) = \frac{\sigma_S^2}{\sigma_S^2+\Lambda(m)}\Lambda(m)\,.$$

Note that both $\Lambda(m)$ and $\Lambda_2(m)$ are one-dimensional, i.e. they are scalar frequency-dependent factors. This means that both the MVDR and MMSE beamformers can be factored into two parts, a multi-channel MSNR beamformer and a single-channel complex weight per frequency. This latter spectral weight is equivalent to a single-channel filter, and is therefore also called a *post-filter*. For example, an MMSE beamformer consists of a multi-channel MVDR beamformer followed by a single-channel Wiener Filter as the post-filter [Brandstein and Ward, 2001][Van Trees, 2002][Wölfel and McDonough, 2009]. We can formulate a generic beamformer version with the arbitrary complex scalar ζ:

$$\boldsymbol{W}^H_{\text{MSNR+}} = \zeta(m)\cdot\boldsymbol{d}(m,\phi_t)\boldsymbol{P}^{\boldsymbol{-1}}_{\boldsymbol{X_N}}\,, \tag{3.82}$$

The scalar factor implies that the beam patterns of all the beamformers which can be expressed in this form have the same basic shape, with the post-filter emphasising certain frequencies over others. Note that due to this scaling, the MSNR and MMSE beamformers do not necessarily satisfy the distortionless constraint.

The generalised eigenvalue beamformer, presented in more detail in Sec. 3.11, exploits the structure of (3.82) by optimising not the full weight vector, but only the post-filter after an MSNR beamformer. This way it can implement several optimisation criteria, including the MSNR, MVDR and MMSE beamformers. In the work presented in Chapters 4 and 5 we will use the *Zelinski* post-filter, a refinement of the Wiener filter, to further enhance the speech [Zelinski, 1988][Marro et al., 1998].

3.9 Linearly Constrained Minimum Variance Beamformer

The single distortionless constraint in Equation (3.68) can be generalised by allowing several linear constraints for further control over the beamformer response. In that case it is also useful to allow target gains other than unity. For example, if we know of an interfering noise source in a specific direction, we can force the gain in that direction to zero, while still fulfilling a desired response g for the look direction. Formally, in the single constraint case we have

$$\boldsymbol{W}^H\boldsymbol{d}(m,\phi_t) = g^*\,, \tag{3.83}$$

where g^* is a complex constant, while for the case with the additional null constraint in direction ϕ_2 this becomes

$$\boldsymbol{W}^H \begin{bmatrix} \boldsymbol{d}(m, \phi_t) & \boldsymbol{d}(m, \phi_2) \end{bmatrix} = \begin{bmatrix} g^* & 0 \end{bmatrix} . \tag{3.84}$$

The weight vector $\boldsymbol{W}$ is of dimension $N \times 1$, and if there are $J < N$ constraints we can write this as

$$\boldsymbol{W}^H \boldsymbol{C} = \boldsymbol{g}^H \text{ or equivalently } \boldsymbol{C}^H \boldsymbol{W} = \boldsymbol{g} , \tag{3.85}$$

where the $N \times J$-dimensional matrix $\boldsymbol{C}$ is called the *constraint matrix* and the $J \times 1$-dimensional vector $\boldsymbol{g}$ the *response vector* [Van Veen and Buckley, 1988].

The *linear constraint minimum variance* (LCMV) beamformer computes the beamformer weights using this general constraint equation, i.e. it solves the following minimisation problem:

$$\min_{W} \boldsymbol{W}^H \boldsymbol{P_X} \boldsymbol{W} \quad \text{subject to} \quad \boldsymbol{C}^H \boldsymbol{W} = \boldsymbol{g} . \tag{3.86}$$

The closed form solution to this problem (see [Doclo, 2003][Wölfel and McDonough, 2009]) is

$$\boldsymbol{W}^H_{\text{LCMV}} = \boldsymbol{g}^H \left(\boldsymbol{C}^H \boldsymbol{P_X}^{-1} \boldsymbol{C}\right)^{-1} \boldsymbol{C}^H \boldsymbol{P_X}^{-1} . \tag{3.87}$$

In most cases $\boldsymbol{P_X}$ is not known and must be estimated from data. Frost [Frost, 1972] suggested a well-known adaptive version of the LCMV beamformer.

3.10 Generalised Sidelobe Canceller

Griffiths and Jim [1982] proposed an efficient adaptive implementation of the LCMV beamformer which breaks the constrained optimisation problem into two parts: a non-adaptive pre-processor that consists of a fixed beamformer and a *blocking matrix*, and an adaptive beamformer with an unconstrained optimisation criterion.

In order to obtain such a structure, we begin by splitting the weight vector $\boldsymbol{W}$ into the two orthogonal parts $\boldsymbol{W_q}$ and $-\boldsymbol{V}$, with $\boldsymbol{W_q}$ required to lie in the range of the constraint matrix $\boldsymbol{C}$, and $\boldsymbol{V}$ in its null space. Since the range and null space of a matrix span the entire space, such a decomposition exists for any weight vector $\boldsymbol{W}$. Mathematically, the decomposition is expressed as

$$\boldsymbol{W} = \boldsymbol{W_q} - \boldsymbol{V} , \tag{3.88}$$

the orthogonality of $\boldsymbol{W_q}$ and $\boldsymbol{V}$ as

$$\boldsymbol{V}^T \boldsymbol{W_q} = 0 , \tag{3.89}$$

and the requirement that $\boldsymbol{V}$ should lie in the null space of the constraint matrix as

$$\boldsymbol{C}^H\boldsymbol{V} = \boldsymbol{0}_{J,1}\,, \tag{3.90}$$

where $\boldsymbol{0}_{J,1}$ is a $J \times 1$-dimensional vector of zeros. Under these conditions, the general LCMV constraints of Equation (3.85) lead to

$$\boldsymbol{C}^H\boldsymbol{W} = \boldsymbol{C}^H(\boldsymbol{W_q} - \boldsymbol{V}) = \boldsymbol{C}^H\boldsymbol{W_q} - \boldsymbol{C}^H\boldsymbol{V} = \boldsymbol{C}^H\boldsymbol{W_q} - \boldsymbol{0}_{J,1} = \boldsymbol{g}$$

In this representation it becomes clear that regardless of our choice of $\boldsymbol{V}$, the constraints specified by $\boldsymbol{C}$ and $\boldsymbol{g}$ will be satisfied by $\boldsymbol{W}$ as long as $\boldsymbol{W_q}$ satisfies

$$\boldsymbol{C}^H\boldsymbol{W_q} = \boldsymbol{g}\,. \tag{3.91}$$

The implication is that the choice of $\boldsymbol{W_q}$ is determined by the constraints, and that the choice of $\boldsymbol{V}$ can be made in such a way as to optimise another criterion, for example to minimise output power. The vector that satisfies Equation (3.91) is [Wölfel and McDonough, 2009]

$$\boldsymbol{W_q}^H = \boldsymbol{g}^H(\boldsymbol{C}^H\boldsymbol{C})^{-1}\boldsymbol{C}^H\,. \tag{3.92}$$

The computation of $\boldsymbol{V}$ is still constrained by Equation (3.90). The real advantage of the structure only comes into play when $\boldsymbol{V}$ is split further into two parts, in order to turn the constrained optimisation into an unconstrained one, at the same time reducing the dimensionality of the vector to be optimised. To that purpose $\boldsymbol{V}$ is expressed as a linear combination of the columns of an $N \times (N-J)$-dimensional matrix $\boldsymbol{B}$ which form a basis for the null space of $\boldsymbol{C}$. Formally,

$$\boldsymbol{V} = \boldsymbol{B}\boldsymbol{W_a} \quad \Rightarrow \quad \boldsymbol{W} = \boldsymbol{W_q} - \boldsymbol{B}\boldsymbol{W_a}\,, \tag{3.93}$$

where $\boldsymbol{W_a}$ is an $(N-J) \times 1$-dimensional vector, and

$$\boldsymbol{C}^H\boldsymbol{B} = \boldsymbol{0}_{J,N-J}\,, \tag{3.94}$$

where $\boldsymbol{0}_{J,N-J}$ is a $J \times (N-J)$-dimensional matrix of zeros. The matrix $\boldsymbol{B}$ can be derived by a number of orthogonalisation procedures such as QR decomposition or singular value decomposition [Van Veen and Buckley, 1988]. Substituting the left side of Eq. (3.93) into Eq. (3.90) yields

$$\boldsymbol{C}^H\boldsymbol{V} = \boldsymbol{C}^H\boldsymbol{B}\boldsymbol{W_a} = \boldsymbol{0}_{J,N-J}\boldsymbol{W_a} = \boldsymbol{0}_{J,1}\,. \tag{3.95}$$

This shows that regardless of the choice of $\boldsymbol{W_a}$, the constraint is satisfied. With the decomposition of $\boldsymbol{V}$ into $\boldsymbol{B}$ and $\boldsymbol{W_a}$, the choice of $\boldsymbol{W_a}$ can therefore be made by unconstrained optimisation.

The output of the beamformer of Equation (3.33) with the given weights is

$$\begin{aligned} Y(m) &= (\boldsymbol{W_q} - \boldsymbol{B}\,\boldsymbol{W_a})^H\boldsymbol{X(m)} & (3.96)\\ &= \boldsymbol{W_q}^H\boldsymbol{X(m)} - \boldsymbol{W_a}^H\,\boldsymbol{B}^H\,\boldsymbol{X(m)}\,. & (3.97)\end{aligned}$$

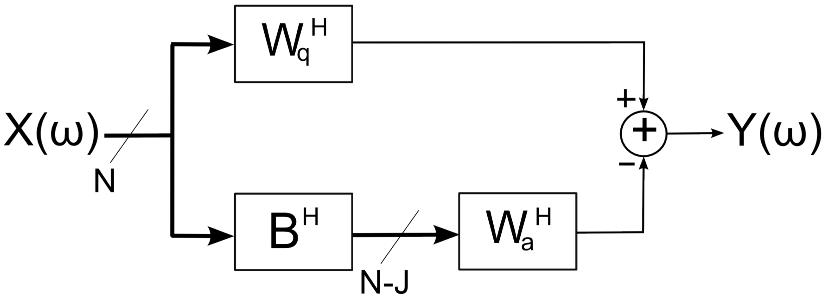


Figure 3.10: The generalised sidelobe canceller. Bold lines indicate vectors, whose dimensionality is indicated below the diagonal line.

Alternatively, in terms of the subband snapshots as defined in Section 3.2.1, we have

$$Y[k] = (\boldsymbol{W_q} - \boldsymbol{B}\,\boldsymbol{W_a})^H \boldsymbol{X}[\boldsymbol{k}]\,. \tag{3.98}$$

The beamformer with such a structure is known as the *generalised sidelobe canceller (GSC)* and illustrated in Figure 3.10. Note that individual vector components are not shown in this diagram; instead the lines representing vector inputs are drawn in bold, and the dimensionality of the vector is indicated by the number below the diagonal line through the bold line. The fixed beamformer's weight vector in the upper branch, $\boldsymbol{W_q}$, is called the *quiescent weight vector*, while the weight vector that is actually adapted to optimise a certain criterion, $\boldsymbol{W_a}$, is called the *active weight vector*.

We can interpret the function of the GSC's components in the following way. The quiescent weight vector in the upper branch creates an estimate of the desired signal, which is often called the *speech reference*. The speech reference will be determined by the constraints one specifies, for example the distortionless constraint. However, it will also contain noise coming from disturbances in directions other than the look direction, entering through the sidelobes of the beampattern. The role of the lower branch is to remove this remaining noise. To do so, the blocking matrix should first block the desired signal from entering its output. In other words, this output should contain only non-speech, that is noise components, of the signal, and is therefore also called the *noise references*. The role of the adaptive beamformer following the blocking matrix is now to match the noise references with the noise remaining in the output of the upper branch, so that subtracting one from the other will eliminate noise entering through the sidelobes from the output of the fixed beamformer. This sidelobe cancellation procedure, depicted schematically in Figure 3.11, is where the GSC takes its name from.

3.10.1 Signal Cancellation Problem

The GSC structure as presented above is based on relatively strong assumptions: accurate knowledge of the desired signal's direction of arrival on the one hand, and

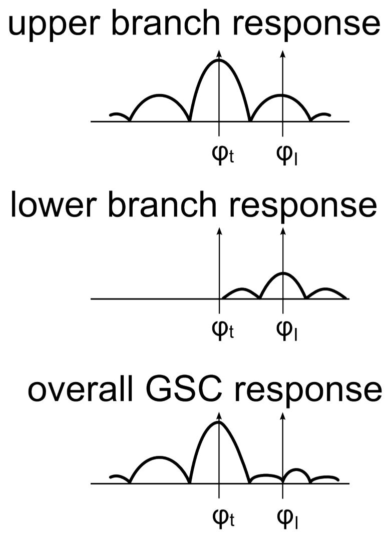


Figure 3.11: The sidelobe cancellation mechanism (based on figure 4.1 in [Van Veen and Buckley, 1988]). ϕ_t is the steering angle, ϕ_I an angle by which interference or noise enters.

a data model that assumes uncorrelated additive noise on the other hand. In reverberant rooms we have multi-path propagation of the desired signal, which leads to a noise component that is correlated with the desired signal. Both inaccurate knowledge of the correct steering angle and interference correlated to the desired signal can lead to problems. In either of these cases it may happen that the desired signal is not fully blocked by the blocking matrix. Since this means that the desired signal also contributes to the energy received in that path, the minimisation of the power then leads to a weight vector that aims to eliminate those parts of the signal leaking into the lower path. This effect is called *signal cancellation.* One possible way of dealing with this problem is to restrict adaptation of the weight vector to periods where the desired signal is not present. For example, in the context of speech recognition, based on VAD the weight vector can be adapted during periods of speech absence only. Many other approaches have been developed to address the signal cancellation problem. Kumatani et al. [2009] classify these approaches into one of the following categories:

- updating the active weight vector only when noise signals are dominant [Cohen et al., 2003, Herbordt and Kellermann, 2003, Nordholm et al., 1993];
- constraining the update formula for the active weight vector with the leaky least mean square (LMS) algorithm [Claesson and Nordholm, 1992, Nordebo et al., 1994] or with power of outputs of the blocking matrix [Hoshuyama et al., 1999];
- using multi-channel target signals received by the microphone array and cor-

relation matrices of the clean and noise corrupted target signals in a calibration phase [Grbić, 2001];

- blocking the leakage of desired signal components into the sidelobe canceller by appropriately designing the blocking matrix [Hoshuyama et al., 1999, Herbordt and Kellermann, 2002, Herbordt et al., 2007, Warsitz et al., 2008];
- taking into account speech distortion due to the leakage of a target signal using a multi-channel Wiener filter which aims at minimising a weighted sum of residual noise and speech distortion terms [Doclo et al., 2007]; and
- using a more general data model, in which an acoustic frequency response models the transformation of the desired source by the room and microphones, instead of merely compensating for the time delays [Cohen et al., 2003, Warsitz et al., 2008, Gannot et al., 2001, Gannot and Cohen, 2004].

Out of the above approaches, the Hoshuyama beamformer [Hoshuyama et al., 1999] has often been used as a baseline. The generalised eigenvalue beamformer is a more recent beamformer that was demonstrated to achieve comparable results to the Hoshuyama beamformer [Warsitz and Haeb-Umbach, 2007][Warsitz et al., 2008]. It therefore represents a state-of-the-art baseline that the algorithm presented in Chapter 5 will be compared to. The following description of the generalised eigenvalue beamformer concludes the background chapter on beamforming.

3.11 Generalised Eigenvalue Beamformer

In Section 3.5.2, we showed that the weight vector of the MSNR beamformer can be determined by solving the generalised eigenvalue problem, from which the *generalised eigenvalue* (GEV) beamformer takes its name. As described by Warsitz *et al.*, the beamformer uses an adaptive method of determining the largest eigenvalue, a problem which is called *dominant* or *principal eigenvalue tracking*. However, there are two aspects that differentiate the GEV beamformer from the MSNR beamformer as given by Eq. (3.67). The first difference is related to the fact that a maximum SNR beamformer followed by a post-filter can be used to realise many different optimisation criteria, as explained in Section 3.8. Exploiting this, the GEV beamformer uses such a post-filter. The second difference is a more general data model, which also accounts for the effect of the room on the signal. As commonly done, this effect is modelled as a linear system with an associated room impulse response $h(t)$ and corresponding frequency response $H(m)$. Instead of the data model in (3.45) we then have

$$\boldsymbol{X}(m) = \boldsymbol{H}(m)\boldsymbol{X}_{\boldsymbol{S}}(m) + \boldsymbol{X}_{\boldsymbol{N}}(m)\,, \tag{3.99}$$

where

$$\boldsymbol{H}(m) = \begin{bmatrix} H_0(m) \\ H_1(m) \\ \vdots \\ H_{N-1}(m) \end{bmatrix} \tag{3.100}$$

is the vector of room frequency responses $H_i(m)$ from the source to each microphone i, and m is the frequency bin. Note that there is no dependence on the frame index k in (3.100) since the room frequency responses are assumed to change slowly enough to ignore the effect of the changes. The effect of the above generalisation on the weight vector equation (3.82) is the replacement of the array manifold vector by the frequency response:

$$\boldsymbol{W}^H_{\text{MSNR+}} = \zeta(m) \cdot \boldsymbol{P}^{-1}_{\boldsymbol{X_N}} \boldsymbol{H}(m)\,. \tag{3.101}$$

When no reverberation is present and the plane-wave assumption is made, the frequency response simplifies to the array manifold vector.

Warsitz *et al.* do not use Eq. (3.101) to determine $\boldsymbol{W}^H_{\text{MSNR+}}$, since that would require estimating the frequency response vector $\boldsymbol{H}$, but the formulation serves to illustrate how the GEV beamformer can account for the room characteristics. Instead, they compute it directly from the noise and signal PSD matrices in the standard eigenvalue problem

$$[\boldsymbol{P}^{-1}_{\boldsymbol{X_N}} \boldsymbol{P_X}]\boldsymbol{W} = \lambda \boldsymbol{W}\,. \tag{3.102}$$

Compared with the previous formulation (3.65), this uses $\boldsymbol{P_X}$ instead of $\boldsymbol{P_{X_S}}$, which is justified when speech and noise are uncorrelated and both zero-mean [Warsitz et al., 2008]. The authors propose two different methods of estimating the post-filter scalar ζ, also without knowledge of $\boldsymbol{H}$, and a novel solution to the dominant eigenvector tracking problem [Warsitz et al., 2008]. They furthermore embed the GEV beamformer in a GSC configuration, using it in place of the usual DSB in the upper branch and constructing the adaptive blocking matrix by orthogonal projection to the upper branch, given by

$$\boldsymbol{B}^H(m) = \boldsymbol{I}_N - \boldsymbol{Q}(m)\boldsymbol{W}^H_{\text{MSNR+}}\,, \tag{3.103}$$

where $\boldsymbol{I}_N$ is the identity matrix of dimension N (the number of sensors), and

$$\boldsymbol{Q}(m) = \frac{\boldsymbol{P_{X_N}} \boldsymbol{W}_{\text{MSNR+}}}{\boldsymbol{W}^H_{\text{MSNR+}} \boldsymbol{P_{X_N}} \boldsymbol{W}_{\text{MSNR+}}}\,.$$

In order to implement the GEV beamformer, the PSD matrices $\boldsymbol{P_{X_N}}$ and $\boldsymbol{P_X}$ need to be estimated. As before, based on a VAD's separation of periods of speech and noise from those with noise only, $\boldsymbol{P_X}$ can be estimated during the former periods, and $\boldsymbol{P_{X_N}}$ during the latter.

Chapter 4

The Original Maximum Negentropy Beamformer

As explained in Sec. 1.2, the novel work of this thesis builds on the MNB. This chapter is therefore dedicated to the details of the original MNB implementation, so that the novel aspects of the work presented in Chapters 5 to 7 can be identified clearly. We begin by examining the statistical distribution of speech samples in the subband domain and the probability density function which was used to model them in the original MNB work in Section 4.1. The optimisation criterion of the MNB, negentropy, will be described further in Section 4.2.2, and Section 4.3 puts all the parts together and describes the overall MNB process in detail.

4.1 Probability Density Functions for Speech Modelling

A *probability density function* (abbreviated pdf or density) provides the likelihood that a continuous random variable X takes a particular value. For example, the pdf of the univariate Gaussian pdf with mean μ and variance σ^2 is given by

$$p_{\text{Gauss}}(X) = p_{\text{Gauss}}(X; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(X-\mu)^2}{2\sigma^2}\right). \tag{4.1}$$

While the *probability* of the r.v. taking on a value cannot be obtained from the pdf, we can calculate the probability of X lying within an interval (a, b) by integrating the pdf over this interval:

$$p(a < X < b) = \int_a^b p(x)dx\,.$$

In order to be a valid pdf, the function p must integrate to unity over all possible values of X, e.g. for a real-valued r.v. with allowed minimum values x_{min} and maximum value x_{max}, it must satisfy

$$\int_{x_{\text{min}}}^{x_{\text{max}}} p(x)dx = 1\,.$$

In Sec. 2.4 it was mentioned that the HMM emission probabilities in ASR are often modelled by a Gaussian pdf, or a mixture of Gaussian pdfs. While this may be a suitable choice for typical feature vectors like cepstral coefficients, other pdfs have been shown more suited to the modelling of speech in the subband domain. For example, Kumatani et al. [2009] presented theoretical arguments and empirical evidence that subband samples of speech are not Gaussian-distributed but can be modelled well with the *generalised Gaussian pdf.*

4.1.1 Generalised Gaussian pdf: model 1

The *generalised Gaussian (GG)* pdf is a well-known pdf which finds frequent application in the fields of blind source separation and independent component analysis. Moreover, it subsumes the Gaussian and Laplace pdfs as special cases.

The MNB work assumes that the mean of the data being modelled is zero, which can be achieved by the subtraction of the real mean if this is not naturally the case. For a zero mean, real-valued r.v. X the GG pdf can be expressed as

$$p_{\text{GG,r}}(X) = \frac{1}{D_1} \exp\left\{ - \left(\frac{X}{\hat{\sigma}\ C_1} \right)^f \right\}, \tag{4.2}$$

with

$$C_1 = \left[\frac{\Gamma(1/f)}{\Gamma(3/f)} \right]^{1/2} \text{ and } D_1 = 2\Gamma(1 + 1/f)\ C_1 \hat{\sigma}\,, \tag{4.3}$$

where f is the *shape parameter*, $\hat{\sigma}$ is the *scale parameter*, and $\Gamma(.)$ the Gamma function[1].

The **scale parameter** $\hat{\sigma}$ controls the spread of the distribution, and the normalisation constant C_1 ensures that the scale parameter $\hat{\sigma}$ is equal to the square root of the variance σ^2 [Domínguez-Molina et al., 2008], formally

$$\hat{\sigma}^2 = E\{|X|^2\} = \sigma^2\,. \tag{4.4}$$

For a proof of this relationship see Appendix C.

The **shape parameter** f determines how the probability mass is distributed around the mean. Note that the GG pdf with $f = 1$ corresponds to the Laplace pdf, and that $f = 2$ yields the Gaussian pdf, whereas in the case of $f \to +\infty$ the GG pdf converges to a uniform distribution. A GG pdf with $f < 2$ is referred to as a *super-Gaussian* pdf, and one with $f > 2$ as *sub-Gaussian*. A smaller shape parameter yields a pdf with a spikier peak and heavier tail than a GG pdf with a large shape parameter, i.e. it has more probability mass around the mean and regions far away from the mean, and less in the intermediate regions. Fig 4.1 shows the likelihood of the GG pdf with the same scaling factor $\hat{\sigma} = 1$ and different shape parameters values from the set $\{0.5, 1.0, 2.0, 4.0\}$. During MN beamforming, the

[1] The Gamma function is an extension of the factorial function to real and complex numbers, whereby the argument is shifted by one so that e.g. $\Gamma(1) = \Gamma(2) = 1! = 1$, and $\Gamma(3) = 2! = 2 \cdot 1! = 2$.

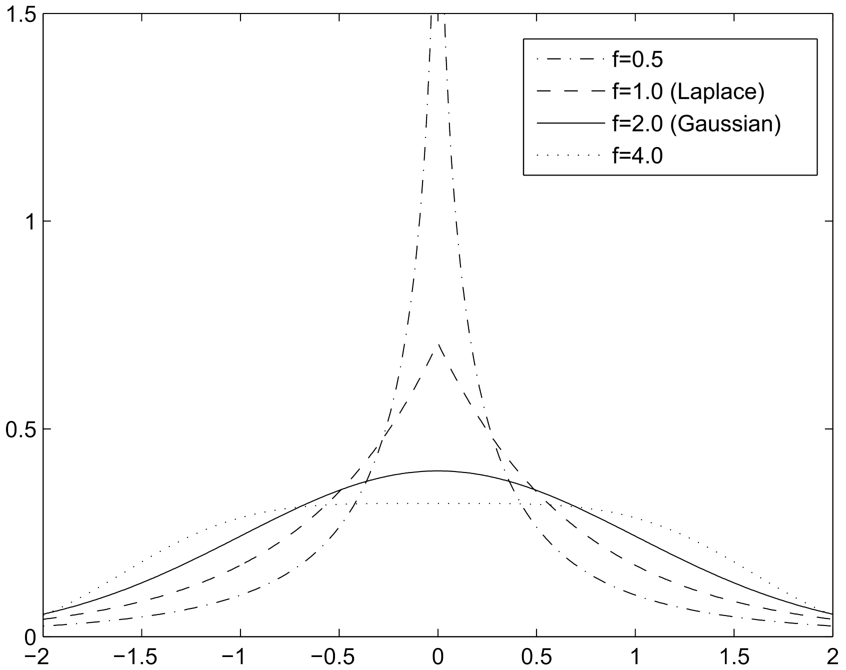


Figure 4.1: The generalised Gaussian (GG) pdf with various shape parameters.

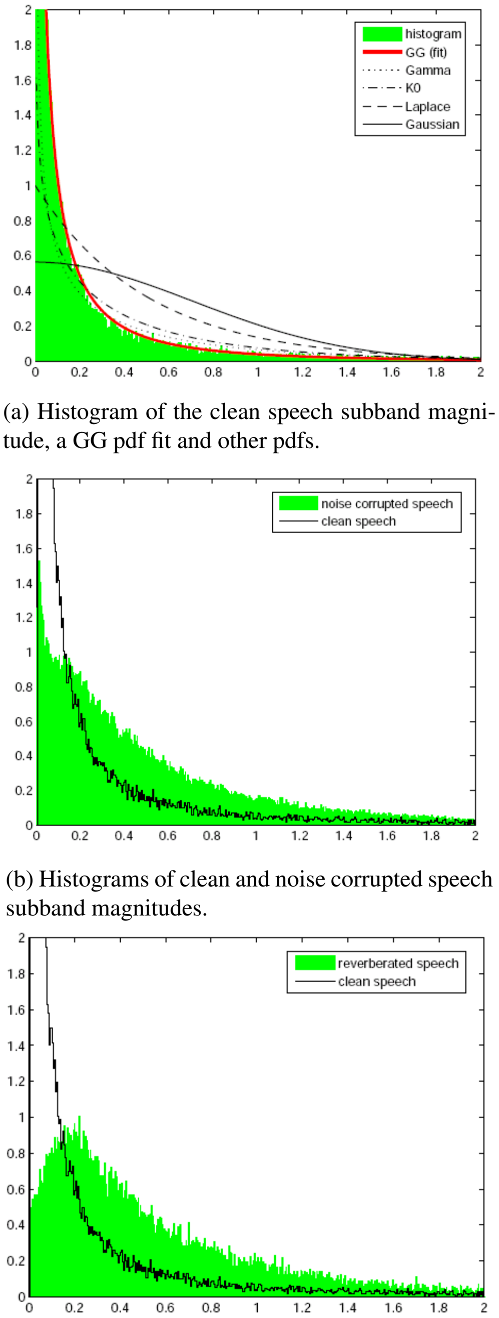
GG pdf is used to model the *complex-valued* subband samples at the output of the beamformer Y, so in order to use Eq. (4.2), these samples had to be transformed into real values. Two different approaches were evaluated in [Kumatani et al., 2009]:

a) modelling the real and imaginary part as independent components by using two r.v.'s, and

b) using the magnitude, i.e. setting $X = |Y|$.

Option (b) led to better ASR results and will henceforth be the only method considered in the description of the MNB. Since a modification of the pdf will be proposed later, we will refer to the use of Eq. (4.2) with the magnitude use (b) as pdf *model 1*.

Figure 4.2 compares histograms of the subband sample magnitudes for noisy and reverberant speech accumulated from 43.9 minutes of the development set of the PASCAL Speech Separation Challenge, Part 2 (SSC2), which is part of the MC-WSJ-AV database described in more detail in Sec. 5.2 [Lincoln et al., 2005]. The clean data is the speech recorded with headsets, while the noisy speech was created by adding several types of noise randomly chosen from the Aurora-2 database [Pearce and Hirsch, 2000] so that the SNR was 0 dB. The reverberant speech is the result of convolving the clean speech with an impulse response measured in the CSTR meeting room, in which the MC-WSJ-AV database was recorded. Part (a) of the figure shows the GG pdf obtained by fitting the scale and



(a) Histogram of the clean speech subband magnitude, a GG pdf fit and other pdfs.

(b) Histograms of clean and noise corrupted speech subband magnitudes.

(c) Histograms of clean and reverberated speech subband magnitudes.

Figure 4.2: Subband magnitude histograms of clean, reverberant and noisy speech, taken from [Kumatani et al., 2009].

shape parameters to the data. The plots clearly demonstrate that the distribution of the clean data is super-Gaussian rather than Gaussian, and that the noisy and reverberant data exhibits a distribution closer to Gaussian.

The fact that noisy and reverberant speech is more Gaussian than clean data is consistent with the aforementioned central limit theorem, which in broad terms states that the sum of a sufficiently large number of independent random variables approaches a Gaussian distribution [Rice, 2001]. The combination of clean speech and noise in noisy speech, and that of the direct path and reflections of the speech in reverberant speech, can be considered a sum of different variables, so it is not surprising that the combined signal is more Gaussian than the clean signal by itself.

Kumatani et al. [2009] tested a variety of pdfs to model the subband samples of speech in the MNB. Since they achieved best results with the GG pdf presented above, we will restrict all further developments reported in this work to that pdf.

4.2 The Optimisation Criterion: Negentropy

The MNB's optimisation criterion is negentropy, a well-known measure of non-Gaussianity. Before we examine in more detail how the MNB uses negentropy in the beamforming process, this section will define it formally and apply it to the GG pdf. Since negentropy is itself defined in terms of *entropy*, a basic measure of information in information theory [Gallager, 1968], we begin with the definition of this measure.

4.2.1 Entropy

The *differential entropy* for a continuous-valued r.v. Y is defined as

$$H(p(Y)) = H(p) \triangleq -\mathcal{E}\{\log p(Y)\} = -\int p(y) \log p(y)\, dy. \tag{4.5}$$

The entropy of a r.v. indicates how much information the observation of the variable provides; a high entropy therefore indicates that a random variable is highly unpredictable. Note that a Gaussian variable has the largest entropy among all random variables of equal variance [Gallager, 1968, Thm. 7.4.1] [Hyvärinen and Oja, 2000], which is also true for complex Gaussian r.v.s [Neeser and Massey, 1993, Thm. 2].

By substituting Eq. (4.2) into (4.5) we obtain the differential entropy for the generalised Gaussian pdf (model 1):

$$H(p_{\text{GG,r}}) = \log D_1 + 1/f\,, \tag{4.6}$$

where D_1 is defined as in Eq. (4.3). The differential entropy for a zero-mean r.v. with a Gaussian pdf can either be obtained by substituting Eq. (4.1) into the entropy definition Eq. (4.5), or by specialising the GG pdf entropy for the Gaussian case by setting $f = 2$ in Eq. (4.6). Either way yields

$$H(p_{\text{Gauss,r}}) = \frac{1}{2}\left(1 + \log(2\pi\sigma^2)\right) . \tag{4.7}$$

When a closed-form solution to the integral in Eq. (4.5) does not exist, or the parameters of the densities change over time, an approximation of the differential entropy expressed directly for a set of K samples $\mathcal{Y} = \{Y(k)\}_{k=0}^{K-1}$ is sometimes used, the *empirical* differential entropy $H_{\text{e}}(\mathcal{Y})$, or short, empirical entropy. It averages the negative loglikelihood over the sample set and is defined as

$$H_{\text{e}}(Y) = -\mathcal{E}\left\{\log p_Y(v)\right\} \approx -\frac{1}{K}\sum_{k=0}^{K-1} \log p_Y(Y(k)) . \tag{4.8}$$

To obtain the empirical entropy for the GG pdf (model 1), we substitute the GG pdf (4.2) with $X = |Y|$ into Eq. (4.8) and simplify to obtain

$$\begin{aligned}
H_{\text{e}}(p_{\text{GG}}, \mathcal{Y}) &= -\frac{1}{K}\sum_{k=0}^{K-1} \log p_Y(Y(k)) \\
&= -\frac{1}{K}\sum_{k=0}^{K-1} \log\left[\frac{1}{D_1}\exp\left\{-\left(\frac{|Y(k)|}{\hat{\sigma}\; C_1}\right)^f\right\}\right] \\
&= -\frac{1}{K}\sum_{k=0}^{K-1} \log\left[\frac{1}{D_1}\right] - \left(\frac{|Y(k)|}{\hat{\sigma}\; C_1}\right)^f \\
&= \log D_1 + \frac{1}{(K\hat{\sigma}\; C_1)^f}\sum_{k=0}^{K-1} |Y(k)|^f \qquad (4.9) \\
&= \log\left(2\Gamma(1+1/f)\;\left[\frac{\Gamma(1/f)}{\Gamma(3/f)}\right]^{1/2}\hat{\sigma}\right) + \frac{\sum_{k=0}^{K-1}|Y(k)|^f}{K\hat{\sigma}^f\;\left[\frac{\Gamma(1/f)}{\Gamma(3/f)}\right]^{f/2}} . \qquad (4.10)
\end{aligned}$$

To distinguish the (exact) differential entropy from the empirical differential entropy, we may refer to the exact differential entropy as H_{d}.

4.2.2 Negentropy

There are two popular criteria of non-Gaussianity, kurtosis and negentropy, both of which are frequently used in the field of independent component analysis. Hyvärinen and Oja [2000] noted that negentropy is generally more robust for outliers than kurtosis.

The differential *negentropy* J for a real-valued or complex-valued r.v. Y is defined as the difference of entropies

$$J(Y) \triangleq H(Y_{\text{Gauss}}) - H(Y) , \tag{4.11}$$

where Y_{Gauss} is a Gaussian variable which has the same variance σ_Y^2 as the variable Y. Since a Gaussian variable has the largest entropy among all random variables of equal variance, negentropy is non-negative, and it is zero if and only if Y has a Gaussian distribution. Assuming we use the GG pdf as the non-Gaussian pdf, we can specialise negentropy to the expression

$$J(Y) \triangleq H(Y_{\text{Gauss}}) - H(Y_{\text{GG}})\,, \tag{4.12}$$

where Y_{GG} is a r.v. with a generalised Gaussian pdf. Substituting Eq. (4.6) and (4.7) into (4.12) gives us the differential negentropy for the GG pdf model 1:

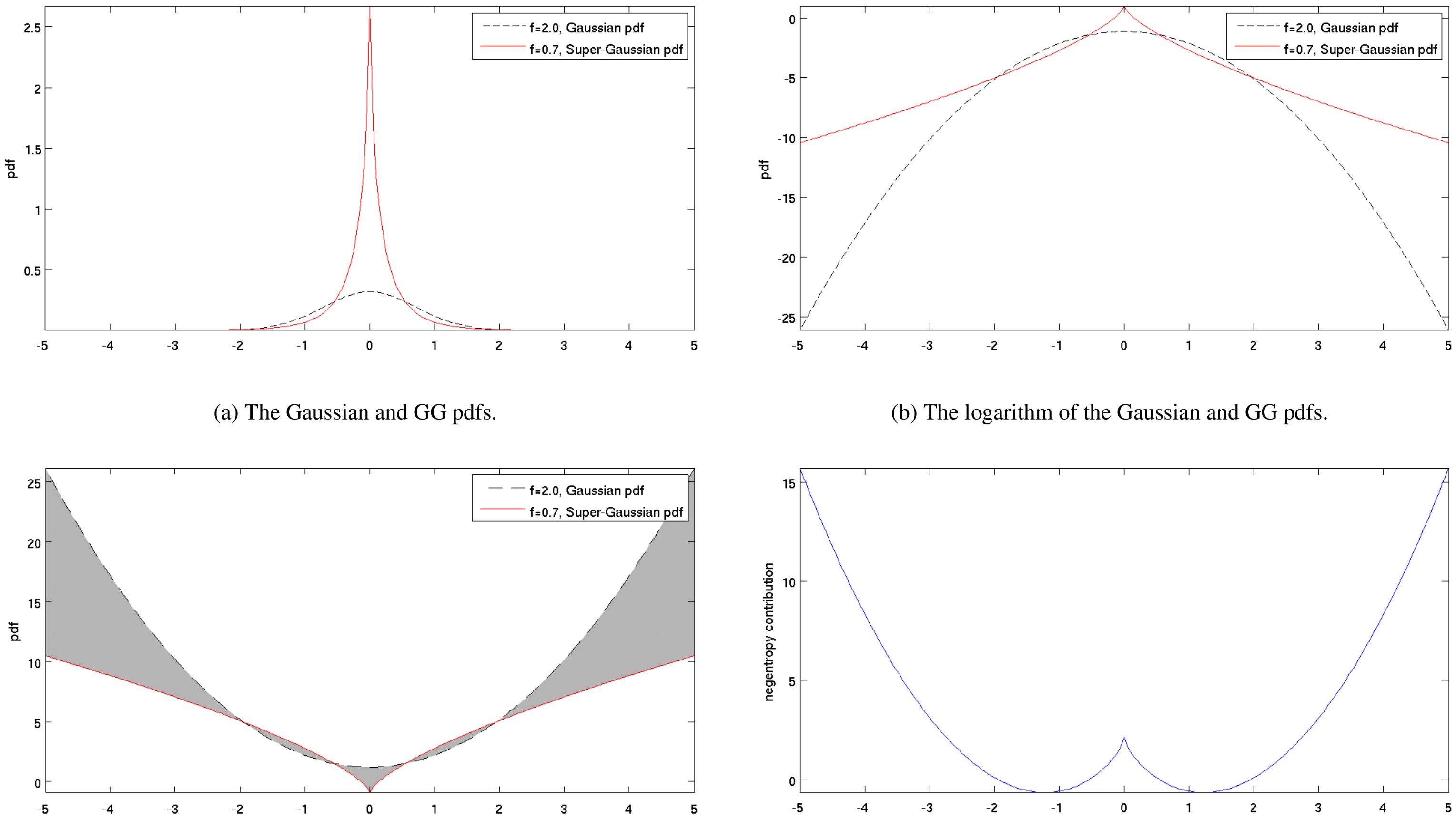
$$J_1(Y) = \frac{1}{2} - \frac{1}{f} + \log \frac{\sqrt{2\pi}\,\sigma}{D_1}\,. \tag{4.13}$$

If we use the difference between the empirical rather than exact differential entropies, we obtain the *empirical* differential negentropy, or simply empirical negentropy, which can be expressed as

$$\begin{aligned} J_{\text{e}}(Y) &\triangleq H_{\text{e}}(Y_{\text{gauss}}) - H_{\text{e}}(Y) \\ &= \frac{1}{K}\sum_{k=0}^{K-1} [\log(p_{\text{Y}}(Y(k)) - p_{\text{Gauss}}(Y(k))] \\ &= \frac{1}{K}\sum_{k=0}^{K-1} \left[\log \frac{p_{\text{Y}}(Y(k))}{p_{\text{Gauss}}(Y(k))}\right]\,, \end{aligned} \tag{4.14}$$

where as before K is the size of the sample set we calculate the negentropy for.

We can visualise negentropy for a r.v. with a GG pdf as the expected distance between the two entropy curves of a GG and Gaussian pdf. Since entropy is defined as the negative of the expected value of the loglikelihood (cf. Eq. (4.5)), Fig. 4.3 shows the curves of a Gaussian and generalised Gaussian pdf, using a scale parameter of 1.0 in all plots. Part (a) plots the original pdf as given by Eq. (6.1), which is a slightly modified version of the GG pdf (4.2) to be motivated later. The logarithm of the same pdfs is shown in part (b), and the negative of both log-pdfs in part (c). Negentropy is the average distance between the Gaussian and GG pdfs depicted in part (c), so the area between the two curves has been shaded, and the difference has been plotted in part (d). This final plot visualises the contribution that a single sample $|Y|$ would make to the negentropy. We observe that the size of the contribution depends on the magnitude of Y, and that the range of values can be divided into three zones: if the sample lies close to the mean, it will make a relatively small but positive contribution to the negentropy; if it lies in a middle zone, the contribution can actually be negative; and if it lies far away from the mean in the tail of the distribution, it will make a large positive contribution, the further, the larger. Where exactly these three zones begin and end, and how large the contributions are in each zone, depends on the scale and shape parameter though. This is illustrated by Fig. 4.4, where the negentropy contribution per sample is plotted for different shape and scale parameter values.



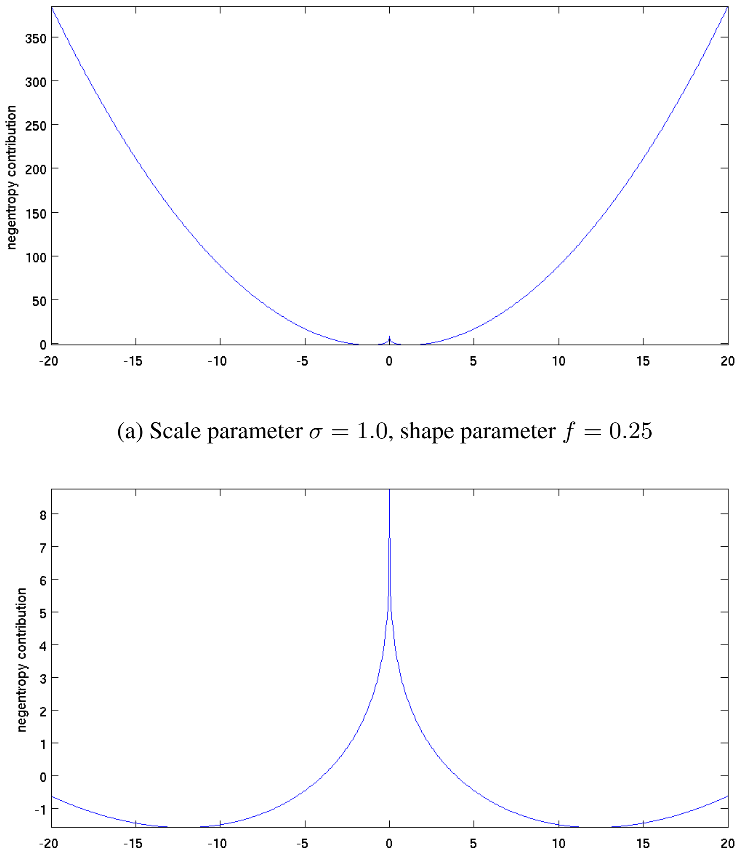
(a) The Gaussian and GG pdfs.

(b) The logarithm of the Gaussian and GG pdfs.

(c) The negative of the log-pdfs.

(d) The difference between the two graphs in (c).

Figure 4.3: The GG and Gaussian pdf with different transformations to visualise the negentropy calculation.



(a) Scale parameter $\sigma = 1.0$, shape parameter $f = 0.25$

(b) Scale parameter $\sigma = 10.0$, shape parameter $f = 0.25$

Figure 4.4: The contribution to empirical negentropy for one subband sample.

4.3 Maximum Negentropy Beamforming

We saw in Section 4.1 that the distribution of clean speech is less Gaussian than reverberant and noisy speech and that it can be modelled well by the GG pdf presented in Section 4.1.1. The basic idea underlying the MNB is therefore to

a) select beamformer weights that make the output speech as little Gaussian as possible by choosing weights that maximise the negentropy of the subband speech samples;

b) use super-Gaussian pdfs (specifically, the GG pdf) to model the subband samples of speech in the calculation of the negentropy.

The MNB does this using a subband beamformer in a GSC configuration (cf. Section 3.10), with a fixed DSB in the upper branch and a statistically optimum beamformer in the lower branch whose active weight vector is chosen in a way that maximises the negentropy of the beamformer output. Recall from Eq. (3.98) that the GSC output at frame k is given by the expression

$$Y[k] = (\boldsymbol{W_q} - \boldsymbol{B}\,\boldsymbol{W_a})^H \boldsymbol{X}[\boldsymbol{k}]\,, \tag{4.15}$$

where $\boldsymbol{W_q}$ denotes the quiescent weight vector of the upper branch chosen to satisfy some constraints (like the distortionless constraint), $\boldsymbol{B}$ is the blocking matrix orthogonal to $\boldsymbol{W_q}$, and $\boldsymbol{W_a}$ the active weight vector. Note that for N microphones at the input of the beamformer and J constraints, the length of $\boldsymbol{W_a}$ is $N - J$, since there is one complex weight per microphone and the upper GSC branch determines J of these weights. For the MNB, $J = 1$ because the DSB in the upper branch determines only the distortionless constraint for the steering angle corresponding to an estimated speaker location.

In the experiments reported in [Kumatani et al., 2009], a Zelinski postfilter [Zelinski, 1988][Marro et al., 1998] was additionally used to scale the beamformer output (cf. Section 3.8). Its weights are given by

$$W_z = \frac{\frac{2}{(N-1)} \left| \sum_{a=1}^{N-1} \sum_{b=a+1}^{N} P_{X_a X_b} \right|}{\sum_{a=1}^{N} P_{X_a}} \tag{4.16}$$

where N is the number of microphones, P_{X_a} the PSD of the time-aligned input at microphone a and $P_{X_a X_b}$ the CSD between microphones a and b.

The final output of beamformer and post-filter is then given by

$$Y_z[k] = W_z\, Y[k] = W_z\, (\boldsymbol{W_q} - \boldsymbol{B}\,\boldsymbol{W_a})^H \boldsymbol{X}[k]\,. \tag{4.17}$$

4.3.1 Overview of the Beamforming Process

Before the beamforming stage, the speaker's position is estimated with the automatic speaker tracking system described by Gehrig et al. [2006]. Based on the average speaker position estimated for each utterance, the quiescent weight vectors

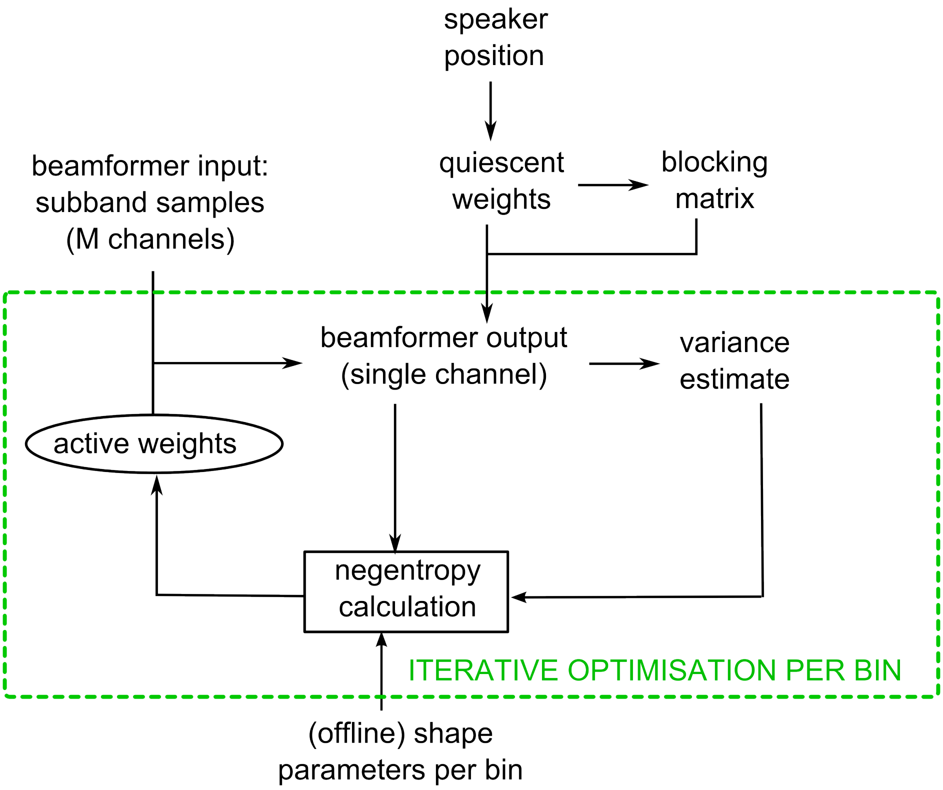

speaker position
beamformer input: subband samples (M channels)
quiescent weights
blocking matrix
beamformer output (single channel)
variance estimate
active weights
negentropy calculation
ITERATIVE OPTIMISATION PER BIN
(offline) shape parameters per bin

Figure 4.5: The MNB optimisation process for one bin of a test utterance.

and corresponding blocking matrix for each frequency bin m are calculated. For each bin, the utterance-dependent active weight vectors $\boldsymbol{W}_{\mathrm{a,m}}$ are then estimated by performing iterations of an optimisation algorithm on the entire utterance until convergence of the negentropy value is achieved. Figure 4.5 shows the stages of the optimisation process for a single bin of one utterance. More details on the actual optimisation routine used are given in Section 5.3.

Once the active weight vectors for all bins have been found, the overall weight vector is applied to the input data and the resulting output of the GSC beamformer is filtered with the Zelinski post-filter. Finally the output is converted from the subband domain back to the time domain. When the beamforming for all utterances of a given speaker is complete, the ASR decoding stage is initiated.

4.3.2 Regularisation

The MNB's optimisation criterion for the choice of the active weight vector is the differential negentropy $J(Y)$ as defined in Equation (4.11). In order to prevent excessively large weights a *regularisation term* is added, yielding the final MNB optimisation criterion

$$\max_{W_a} \mathcal{J}(Y, \alpha) = \max_{W_a}(J(Y) - \alpha \|\boldsymbol{W_a}\|^2); \quad \alpha \in \mathbb{R}_+ . \tag{4.18}$$

Since we aim for active weights that maximise the negentropy, deducting a term involving the magnitude of the weight vector will lead to smaller weights. Kumatani et al. [2009] tried several values for the regularisation factor α and found a value of 0.01 to be suitable. In the experiments conducted for this thesis the same value will therefore be used, unless stated specifically otherwise.

In the absence of a closed-form solution for the optimal $\boldsymbol{W_a}$, non-linear optimisation techniques can be used to find a (local) maximum in the optimisation space. Section 5.3 describes the optimisation method used in this thesis.

4.4 Estimation of the PDF Parameters

Three pdf parameters are needed to calculate the negentropy (4.12) for the beamformer output Y of a test utterance with the zero-mean Gaussian and GG pdf models:

- two parameters for the generalised Gaussian (4.2): the scale parameter $\hat{\sigma}$ and the shape parameter f;
- one parameter for the Gaussian pdf (4.1): the variance σ^2.

Kumatani et al. [2009] estimated each of these parameters independently for each subband, as the optimal pdf is frequency-dependent. In this thesis we will continue the subband-specific parameter estimation, but how exactly and on which data the estimation is done is an aspect that will be modified. In this section we will describe the parameter estimation as it was done in the original MNB work.

4.4.1 Estimation Data: Offline vs. Online Estimation

Two different data sets can be used to estimate the pdf parameters. We therefore distinguish between the following two estimation types:

- *online parameter estimation* (onPE) is done during beamforming, and based on the beamformer output for the test utterance. Note that this implies that the Gaussian and GG pdf parameter estimates change whenever the active weight vector changes in an iteration of the beamforming, since the beamformer output $Y[k]$, which is dependent on $\boldsymbol{W_a}$, influences the parameter estimates.
- *offline parameter estimation* (offPE) estimates the parameter on development data in a pre-processing step, i.e. a corpus of training utterances. The advantage of the offPE approach is the robustness of the estimate due to the availability of much more data.

The same estimation methods can in principle be applied in offPE and onPE. However, when choosing a combination of estimation methods, we need to make sure the beamforming process is still dependent on the test utterance in some way - if we only used offline parameters, then no change of the active weight vector would affect the negentropy values (except for the regularisation term).

The original MNB work always estimated the shape parameter in an offline fashion and the variance and scale parameter in an online fashion. This way, the number of parameters that are estimated on the relatively small number of test utterance frames is limited to reduce the problem of data sparsity. While this offPE/onPE combination is only one possible configuration, the present work does not change this aspect of the beamformer. The following sections will explain in more detail how the MNB offline and online estimation was done. Since we will introduce several estimation methods, we will introduce abbreviations for each of them for ease of reference. These identifiers are given in bold face within square brackets at the point of their definition.

4.4.2 MNB Variance and Scale Parameter Estimation (onPE)

The original MNB sets the Gaussian variance σ^2 to the moment estimate over all frames of the utterance. The *moment method* [**MM**] equates the distribution moments with the sample moments and then solves the equations for the parameters to be estimated. For the variance it is simply equal to the second moment, and so like the original MNB work we always set σ^2 to the sample variance

$$\sigma^2 = \frac{1}{K} \sum_{k=0}^{K-1} |Y[k]|^2 \,, \tag{4.19}$$

where K is the number of subband samples used for estimation – the original MNB uses the total number of frames of the utterance. For the Gaussian pdf, the moment estimate of the variance is actually equivalent to the maximum likelihood estimate, which will be explained below. We therefore refer to the variance estimate (4.19) with the identifier [**ML=MM**].

The MM estimate of the GG scale parameter is the square root of the moment estimate of the variance. The original MNB work used a different estimate though, the *maximum likelihood* [**ML**] estimate [Aldrich, 1997], which maximises the likelihood function, or equivalently the loglikelihood function, over some data. The intention is to find the set of parameters which makes the data ”more likely” than any other values would make them, assuming independence of data points. The loglikelihood function is defined as

$$L(p, \mathcal{Y}) \triangleq \log \prod_{k=0}^{K-1} p(Y[k]) \tag{4.20}$$

of a pdf p over a data set $\mathcal{Y} = \{Y[k]\}_{k=0}^{K-1}$. Substituting the pdf (4.2) into this definition gives us the GG pdf loglikelihood function:

$$L_{GG}(\mathcal{Y};\hat{\sigma},f) = -K\log D - \frac{1}{(C\,\hat{\sigma})^f}\sum_{k=0}^{K-1}|Y[k]|^f\,, \tag{4.21}$$

where $C = C_1$ and $D = D_1$ from Eq. (4.3) for pdf model 1.

The ML estimate of $\hat{\sigma}$ can be derived by finding the point that maximises (4.21), e.g. by setting the partial derivative of L with respect to $\hat{\sigma}$ to zero and solving for $\hat{\sigma}$, yielding

$$\hat{\sigma}_{\mathrm{ML}} = \frac{1}{C}\left(\frac{f}{EK}\sum_{k=0}^{K-1}|Y[k]|^f\right)^{1/f}\,, \tag{4.22}$$

where C is defined as before and $E = 1$ for model 1; see Appendix D for a derivation.

4.4.3 MNB Shape Parameter Estimation (offPE)

Kumatani et al. [2009] determined the offline estimate of the shape parameter f on the single speaker data from the aforementioned PASCAL Speech Separation Challenge, part 2. In short, the shape parameter was set to its ML estimate. However, since a closed form solution does not exist for this estimate, the value of f that maximises (4.21) is found iteratively. The initial shape and scale parameter used in this iterative process are obtained with the moment method. In other words, the estimation consisted of two steps:

1. Parameter initialisation:
   The scale parameter was set to the moment estimate (4.19) over the complete training corpus. For the shape parameter, the moment estimate equation is more complicated and does not always have an exact solution. Kumatani et al. therefore follow the method proposed by Domínguez-Molina et al. [2008] to determine an approximation for the initial value of f.

2. ML estimation of f with a gradient ascent procedure:
   In each iteration (for a maximum of eight iterations), the ML estimate of the scale parameter (4.22) is computed, using the previously computed shape parameter. Using this new scale parameter estimate, the likelihood gradient expression with respect to f given by Varanasi and Aazhang [1989] is computed, and the current value of f is updated by taking a step in the direction of the gradient.

Since the above estimation method is relatively complicated, we will show in Section 6.1 how a simpler and more efficient method can be used while still obtaining comparable ASR results.

Chapter 5

Experimental Framework

The purpose of this chapter is to describe the framework common to all the experiments reported in this thesis, namely the experimental configuration, the evaluation methodology and the implementation details concerning the beamforming module, ASR system, and optimisation routine.

5.1 Beamforming

The beamforming module covering the optimisation and application of the final weights was implemented as extension modules of the Python scripting language, which accessed some core functionality of the open-source BTK beamforming toolkit implemented in C++. The original MNB code was incorporated into a newly created object-oriented class hierarchy, and the functionality was extended by the novel parameter estimation methods to be described in later chapters.

The beamforming front-end is the same as the one used in the original MNB work. Subband analysis and synthesis were performed for 256 frequency bins with a uniform DFT filter bank based on the modulation of a single Nyquist(M) prototype impulse response [Kumatani et al., 2008]. Since bin 0 corresponds to the signal's average or DC value, it can be ignored, and due to the symmetry of the frequency bins, only the bins up to bin 128 need to be optimised. In the experiments for Kumatani et al. [2009], the first author found that every second frame could be left out to speed up the beamforming without a loss of performance. This strategy was copied, and together with a sampling frequency of 16 kHz and a frame shift of 128 samples it resulted in an effective frame shift of 256 samples for beamforming. The optimisation routine used to determine the optimal active weights for each bin is described below in Sec. 5.3.

5.2 ASR Evaluation

Beamforming applications have been evaluated by a variety of methods, including human judgement scores, SNR improvement, or distance from clean speech in terms of measures like the Itakura-Saito distance [Basseville, 1989]. At this point in time not many large databases of noisy and reverberant speech recorded with microphone arrays are readily available, so small data sets or artificially created data are often used, e.g. clean speech played through a loudspeaker and recorded with a microphone array, or convolved with room impulse responses measured for an array in a given room. While such artificial conditions may be useful for analytical reasons, they generally simplify and differ from natural settings; for example, speech played through a loudspeaker does not contain the variations caused by the head movement of human speakers. Regarding the evaluation metric, ideally it should be chosen with the target application in mind. While an algorithm may increase the SNR or other quality measure of speech, such an improvement does not always carry over to ASR performance. In this work, each proposed modification to the MNB will therefore be evaluated by ASR experiments on utterances spoken by humans and recorded with a microphone array. The standard evaluation metric for ASR systems is the *word error rate (WER)*, which is determined by three types of errors observed in the set of test utterance transcriptions when compared to a set of reference transcriptions: deletions, insertions, and substitutions. A deletion occurs when a word from the reference transcription is missing in the ASR output, an insertion when a word was added, and a substitution when a word was replaced by an incorrect word. To determine the WER, the output of an ASR system is aligned with the reference transcription by dynamic programming, and the error rate in percent is then given by the expression

$$\mathrm{WER} = 100 \cdot \frac{\text{deletions + insertions + substitutions}}{\text{total number of word tokens in the reference}} . \tag{5.1}$$

The number of insertions can be influenced by changing the insertion penalty mentioned in Sec. 2.5.

5.2.1 ASR System and Corpora

All far-field ASR experiments were conducted with the Millennium automatic speech recognition system. As in the original MNB work, this thesis uses a test set from the *Multi-Channel Wall Street Journal Audio-Visual Corpus* (MC-WSJ-AV) recorded in the Edinburgh CSTR meeting room and described by Lincoln et al. [2005]. In the single speaker stationary scenario of the MC-WSJ-AV, a speaker was asked to sit or stand in front of a presentation screen and read Wall Street Journal (WSJ) sentences from different positions. The task lies between simple digit recognition and large vocabulary conversational speech recognition in difficulty and therefore provides a suitable test set for current distant ASR research. The speech was recorded with a headset microphone, a lapel microphone and two circular, eight-channel table-top microphone arrays, all with a sampling frequency of 16 kHz. The room exhibits significant reverberation, with the reverberation time T60 estimated by Kumatani et al. [2009] as 0.38 seconds. In addition to the reverberation, the recordings include background noise from air fans or nonstationary noise like the sound of a bus passing outside the building. Refer to [Lincoln

			std.	percentiles		
minimum	maximum	mean	deviation	25th	50th (median)	75th
186	1813	834.23	298.78	633.5	818.5	1027

Table 5.1: Statistics of the number of frames per utterance of the test set.

et al., 2005] for further details of the data collection apparatus. Our test data set for the experiments contains 10 speakers recorded with the first array where each speaker reads approximately 40 sentences taken from the 5,000 word vocabulary WSJ task. This provided a total of 352 utterances which correspond to approximately 43.9 minutes of speech and consist of a total of 5852 word tokens in the reference transcriptions.

Table 5.1 describes the number of frames per utterance in the test set statistically by giving the minimum and maximum number of frames as well as the mean, standard deviation and the quartiles. Note that the number of frames used by the optimisation is half of the number of frames given here, as only every second frame was used for beamforming. A variance estimate during beamforming using all frames of the utterance is therefore based on approximately 417 frames on average.

The speech recognition system configuration is identical to the one used by Kumatani et al. [2009]. The ASR engine was trained on parts of the ICSI, NIST, and CMU meeting corpora, as well as the Transenglish Database (TED) corpus, using a total of 100 hours of training material. In addition to these corpora, approximately 12 hours of speech from the WSJCAM0 corpus [Fransen et al., 1994] was used for HMM training in order to cover the British accents for the speakers. Two different training schemes were used: conventional ML training and ML-SAT (cf. Sec. 2.6.1). The resulting ASR system is a fully continuous context-dependent triphone system with 1,743 mixture models and a total of 67,860 Gaussian components.

The ASR front-end was based on cepstral features estimated with a warped MVDR [Wölfel et al., 2005] spectral envelope of model order 30. Since the MVDR provides an increased resolution in low-frequency regions relative to the conventional Mel filters, neither the Mel filter bank nor any other filter bank was needed. For each frame 20 cepstral coefficients were extracted, followed by CMS and CVN (cf. Sec. 2.6). The resulting frame-wise feature vectors were then concatenated into groups of 15 consecutive frames, to which a linear discriminant analysis [Haeb-Umbach and Ney, 1992] was applied to yield more compact features of length 42, followed by a further application of CMS. The final step consisted of a semi-tied covariance (STC) transform [Gales, 1999], which optimises the use of the covariance matrices by employing a set of full covariance matrices which are shared by many distributions in addition to diagonal matrices associated with each pdf.

Four decoding passes were run on the final waveforms, with each pass using either a different speaker adaptation scheme or acoustic model. Specifically, the passes are:

1. Decode with the unadapted, conventional ML acoustic model.
2. Estimate VTLN and CMLLR parameters for each speaker, then redecode with the conventional ML acoustic models.
3. Estimate VTLN, CMLLR, and MLLR parameters for each speaker, then redecode with the conventional models.
4. Estimate VTLN, CMLLR, MLLR parameters for each speaker, then redecode with the ML-SAT models.

The language model used in each pass was the standard 5K bigram LM provided with the original WSJ corpus [Paul and Baker, 1992]. Speaker adaptation parameters in each pass were estimated using the word lattices generated during the prior pass. For the experiments reported here, the transformation matrix and cepstral bias vector computed during speaker adaptation belong to a set of sparsely parameterised *all-pass transforms* [McDonough, 2000].

5.2.2 Statistical significance test

The final WERs were tested for statistical significance with the *matched pair sentence segment word error* (MAPSSWE) test. This parametric test divides the word string into segments specific to the output of the two systems being compared and assumes that errors are independent across segments, but not within each segment. The MAPSSWE test was suggested for the evaluation of ASR results by Gillick and Cox [1989] and adopted by the National Institute of Standards and Technology (NIST), which used it in such important evaluations as the Rich Transcription 2007 Meeting Recognition Evaluation [Fiscus et al., 2008]. To determine the MAPSSWE results, version 2.4.0 of the NIST SCTK toolkit implementing this test was used.

5.2.3 Baseline Results For DSB, MMSE and GEV Beamformers

Table 5.2 gives the WER results for the baseline beamformers when using the eight channels from the first array. The DSB, MMSE and GEV beamformers as implemented by Mr Kenichi Kumatani were compared with each other. The results show that both the MMSE and GEV beamformer provide a substantial and statistically significant improvement over the DSB. The difference between the MMSE and GEV beamformers on this data is not significant, however.

ID	Beamformer	WER
E001	DSB	16.63
E027	MMSE	14.49
E052	GEV	14.47

Table 5.2: Baseline word error rates for 8 channel recordings.

5.3 Optimisation Method

To determine the optimal weight vector the point in the high-dimensional weight space has to be found which leads to maximum negentropy. As done in the original MNB work, the imaginary and real part of the complex weights were treated as separate dimensions when doing so, i.e. for the configuration used in this thesis of eight microphones, there are seven complex weights in the active weight vector and therefore a 14-dimensional optimisation space.

Since optimisation routines are usually implemented to find a minimum, a multi-dimensional minimisation algorithm was used to find the weight vector providing the minimum negative negentropy. As in the original MNB work, all weights were initialised to zero to allow a direct comparison of the new with the original system. In principle the initial parameters could be set to the final vectors of the simpler MNB, which might lead to further improvements at the cost of the time spent calculating these initial weight vectors. In the present work this was however not evaluated because the proposed methods did not provide improved performance over the original MNB.

5.3.1 The Simplex Algorithm

All the experiments in this thesis employed the simplex algorithm based on the work of Nelder and Mead [1965] as implemented by the Gnu Scientific Library (GSL), version 1.9, and accessed through the PyGSL Python interface, version 0.9.1. Even though there are faster optimisation routines available than the simplex algorithm, it is well-known and effective. It simplifies the implementation because in contrast to gradient-based optimisers it does not require the calculation of a derivative.

In geometry, a *simplex* describes an n-dimensional body (strictly speaking, a polytope, which is the generalisation of a polygon to n dimensions). An n-simplex has $n + 1$ vertices in n dimensions; for example, a 2-simplex is a triangle, or a 3-simplex a tetrahedron. To find an optimal point in an n-dimensional space, the simplex algorithm constructs an initial n-simplex and modifies the vertices iteratively, moving them through the optimisation space until a stopping condition is reached.

Let the set $P[k] = \{\boldsymbol{p_0}[k], \ldots, \boldsymbol{p_n}[k]\}$ denote the set of $n + 1$ vertices of the simplex at iteration k. The initial simplex $P[0]$ is constructed from an n-dimensional starting vector $\boldsymbol{x} = \begin{bmatrix} x_0 \\ \vdots \\ x_{n-1} \end{bmatrix}$ and an equally long step size vector

$\boldsymbol{s} = \begin{bmatrix} s_0 \\ \vdots \\ s_{n-1} \end{bmatrix}$ as follows:

$$\begin{aligned} P[0] = \{\boldsymbol{p_0}[0] &= \boldsymbol{x} = (x_0, x_1, ..., x_{n-1}), \\ \boldsymbol{p_1}[0] &= (x_0 + s_0, x_1, ..., x_{n-1}) \\ &\vdots \\ \boldsymbol{p_n}[0] &= (x_0, x_1, ..., x_{n-1} + s_{n-1})\} \,. \end{aligned}$$

Let $f(\boldsymbol{p})$ be the function providing the value of the optimisation criterion (e.g. negative negentropy) for a given point $\boldsymbol{p}$. Associated with each simplex $P[k]$ is the vector of values of f for any of the simplex vertices,

$$\begin{aligned} F[k] = \{\boldsymbol{f_0}[k] &= f(\boldsymbol{p_0}[k]), \\ \boldsymbol{f_1}[k] &= f(\boldsymbol{p_1}[k]), \\ &\vdots \\ \boldsymbol{f_n}[k] &= f(\boldsymbol{p_n}[k])\} \,. \end{aligned}$$

Now let $f_{\text{min}}[k]$ be the element of F with the best (i.e. lowest) value. In each iteration, the algorithm changes the current simplex $P[k]$ by applying a set of simple geometrical transformations to its vertices, such as reflection, expansion, contraction and multiple contraction. Appendix F describes this process in pseudocode.

Due to the nature of the algorithm the current best criterion value f_{min} is not necessarily improved at every iteration, so a lack of improvement of the optimisation criterion cannot be used as a stopping condition. The GSL implementation provides an alternative, where the size of the simplex is computed as the average distance from the geometric centre of the simplex to all of its vertices. The algorithm either stops when the maximum number of iterations is exceeded or the size of the simplex becomes smaller than a user-provided value, which was set to 10^{-10} in the experiments reported here. Based on inspection of typical optimal vectors, the initial step size vector was set to 0.2 in all dimensions. Unless noted otherwise, the maximum number of iterations was set to 2000, which had been observed to be a value that was only exceeded in case of malfunctioning.

5.3.2 One-Dimensional Optimisation: Line Search

Some modules of the system described in the present work require a one-dimensional optimisation, or line search, in which case the GSL implementation of the Brent minimisation algorithm is used [Brent, 1973]. This algorithm is a combination of the well-known golden section search with a parabolic interpolation approach. In broad terms, the algorithm proceeds as follows. At the beginning of each iteration, only three points are stored: a lower limit and an upper limit providing the current interval, and a middle point within that interval which must have a lower criterion value than the interval limits. Given a starting interval and an initial middle point, the algorithm iteratively reduces the interval with the aim of bracketing the minimum of the function. Brent's method constructs an interpolating parabola through the three existing points, and the minimum of the parabola becomes a candidate for a new limit point, which would replace the point with the highest criterion value. In order to do so the parabola's minimum must lie within the current interval though, if that is not the case, a golden section step is taken as a fall-back measure. To do so, the larger of the two intervals between current lower limit and middle point, and middle point and upper limit is chosen, and a new point in this interval is determined by dividing it in a golden section ratio. The criterion value at this new point is evaluated, and the least useful point is then discarded to obtain an updated interval and middle point. The search stops when either the maximum number of iterations is exceeded, or the size of the interval becomes smaller than a user-specified value. In the experiments reported here, the stopping criterion was set to 10^{-5}, and the maximum number of iterations to 100 but never exceeded.

The remainder of the thesis will now describe the modifications made to the MNB and the experimental outcome.

Chapter 6

Investigating Various Aspects of the MN Beamformer

This chapter describes a set of experiments involving four modifications to the original MNB:

1. a simpler and more efficient offline shape parameter estimation method,
2. a change of the generalised Gaussian pdf formula to a theoretically valid pdf for a complex r.v.,
3. the use of the same variance for both pdfs used in the calculation of the negentropy, and
4. the use of the empirical differential entropy rather than the exact version in the calculation of the negentropy.

Due to dependencies between points 2 to 4, these changes and the resulting ASR results will be presented together in Section 6.2, while point 1 is presented on its own in the next section.

6.1 Improving the Offline Estimation Method

We begin by investigating the effect on WER of a simpler and more efficient shape parameter estimation method than the one described in Section 4.4.3. Recall that the original MNB uses an iterative gradient ascent method, whereby after an initialisation step involving the moment estimate, the ML estimate of sigma is repeatedly determined with Eq. (4.22), followed by the calculation of the shape parameter gradient [**MM-ML-GA**].

An initial experiment consisted of leaving out the relatively complex shape parameter initialisation to the moment estimate, and simply setting it to 0.7, a super-Gaussian value. The scale parameter is set to the moment estimate as before. This variant involving ’manual initialisation’ is labelled [**MI-ML-GA**].

Departing from the gradient ascent approach, a new and simpler estimation method can be implemented based on the following observation: if an estimate of $\hat{\sigma}$ is given, substituting it into the likelihood equation (4.21) results in an expression that only depends on f and therefore a one-dimensional optimisation problem which can be solved by a simple line search (LS), as suggested by [Wölfel and McDonough, 2009, ch. 13.5.2]. Since Eq. (4.22) is a closed-from solution for $\hat{\sigma}_{\mathrm{ML}}$, using that for the substitution and performing the line search results in a joint ML estimate for both GG parameters. Performing the substitution of (4.22) with $E = 1, C = C_1$ and $D = D_1$ as defined in (4.3) into Eq. (4.21) yields:

$$\begin{aligned}
L_{GG_r}(\mathcal{Y}; f) &= -K \log\left(2\Gamma(1+1/f)\ C_1 \hat{\sigma}\right) - \frac{\sum_{k=0}^{K-1} |Y[k]|^f}{(C_1\, \hat{\sigma})^f} \\
&= -K \log\left(2\Gamma(1+1/f)\ C_1 \left[\frac{1}{C_1}\left(\frac{f}{K}\sum_{k=0}^{K-1}|Y[k]|^f\right)^{1/f}\right]\right) \\
&\quad - \frac{\sum_{k=0}^{K-1} |Y[k]|^f}{\left(C_1 \left[\frac{1}{C_1}\left(\frac{f}{K}\sum_{k=0}^{K-1}|Y[k]|^f\right)^{1/f}\right]\right)^f} \\
&= -K \log\left[2\Gamma(1+1/f)\left(\frac{f}{K}\sum_{k=0}^{K-1}|Y[k]|^f\right)^{1/f}\right] - \frac{K}{f}\,.
\end{aligned}$$

Using a line search, the optimal shape parameter maximising this expression over the training data is found per frequency bin. We will refer to this method as [**ML-LS-ML**].

ID	PDF Model	Neg-entropy	Estimation methods			WER %	Time (offPE) hh:mm
			Gaussian σ^2 (onPE)	GG $\hat{\sigma}$ (onPE)	GG f (offPE)		
E2016	1	exact	ML=MM	ML	MM-ML-GA	13.50	13:45
E2021	1	exact	ML=MM	ML	MI-ML-GA	13.64	10:38
E2035	1	exact	ML=MM	ML	ML-LS-ML	13.35	04:18

Table 6.1: Comparison of old and new offline shape parameter estimation methods.

Table 6.1 presents the ASR results obtained with the old and new offline parameter estimation methods, keeping all other aspects of the system fixed. The first column shows an experiment identifier, the second column the pdf model (in this case the original model 1 of Sec. 4.1.1 in both cases), and the third column the negentropy type; in all of the above cases the exact negentropy of Eq. (4.5) was used rather than the empirical entropy of Eq. (4.8). The fourth to sixth columns show the parameter estimation methods using the identifiers introduced here and in Sec. 4.4.2, and the seventh column lists the word error rate. The final column gives the amount of CPU time taken for the offline parameter estimation method on a machine with a 3 GHz Intel Xeon 5160 processor and 16 GB RAM machine. We observe that

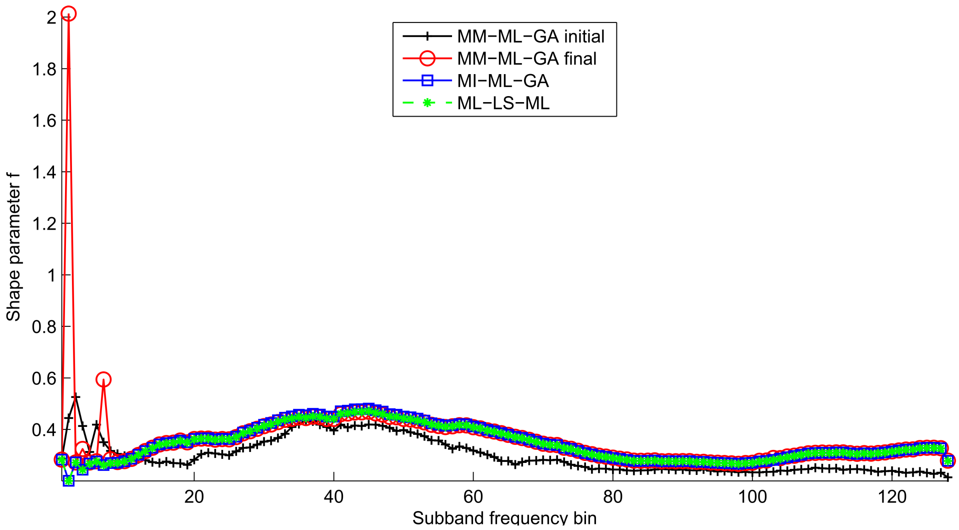


Figure 6.1: The results of various shape parameter estimation methods compared.

1. the initialisation to the fixed shape parameter value rather than the moment method (MI-ML-GA rather than MM-ML-GA) leads to a slightly worse WER but saves 23% of the estimation time;

2. the new offline estimation method ML-LS-ML, which is more than three times faster than the old method MM-ML-GA and still takes only 40% of the time needed by MI-ML-GA, leads to a small improvement in WER.

While we see small these changes in WER, according to the MAPPSWE test none of the result pairs are significantly different. The new estimation methods therefore provide improved efficiency and comparable word error rates.

An interesting difference can be seen when examining the resulting shape parameters. Fig. 6.1 plots the shape parameter for each frequency bin as estimated by the various methods, including the initial moment estimate approximation of MM-ML-GA to clarify the effect of the initialisation. We observe that for the bins above bin 10, all estimates are rather similar, with values in the super-Gaussian range 0.2-0.5. In particular, the three final estimates are almost identical for those bins, with only the initial moment estimate approximations being slightly lower. For the bins below bin 10, some more pronounced differences can be seen, with the main ones concerning the estimates for bin 2 and bin 7 of the final MM-ML-GA method. Bin 2 in particular is the only value above 2.0, i.e. no longer super-Gaussian as expected. Since these outliers neither occur for the fixed initialisation variant MI-ML-GA nor the line search method ML-LS-ML, they cannot be attributed to either the moment estimates or the gradient ascent procedure separately, but rather the combination of both. We could investigate further why this is the case, but since the focus of this section is the validation of the simpler and more efficient method, we will simply conclude by summarising that the new method ML-LS-ML does not suffer from the problems of estimate outliers and is much more efficient than the previously used method while still providing comparable WERs. Based on these findings, all further experiments presented in this thesis will use the offline shape parameter estimation method ML-LS-ML.

6.2 Further modifications

In the following sections changes to three further aspects of the MNB will be explained before the ASR results for the eight possible combinations of the old versus the new setting of each aspect are presented together in Section 6.2.4. The reasons for presenting the results together rather than separately are twofold: on the one hand, the eight experiments provide more evidence for the effect of each modified aspect, and more importantly, this allows us to describe a dependency that exists between the negentropy calculation and the parameter estimation methods.

6.2.1 Correcting the PDF for a Circularly Complex R.V.

The second modification to the original MNB configuration presented in this thesis is a change in the pdf model. Kumatani et al. used Eq. (4.2), the GG pdf for a *real-valued* r.v. with the magnitude of the complex subband output as the random variable. We referred to this pdf as model 1. Using the real-valued version for complex-valued data can be considered theoretically problematic in that the pdf does not integrate to unity over the entire complex plane. We therefore evaluated a modified pdf which is based on the assumption that the subband sample $Y_m(k)$ can be modelled as a *spherically invariant random process* (SIRP) [Brehm and Stammler, 1987], which implies that the probability is independent of the phase of the complex-valued r.v.. While model 1 also depends only on the magnitude $|Y_m(k)|$, it is not a valid SIRP pdf due to the aforementioned integration issue. Correcting this, we can express the pdf for a zero-mean, spherically invariant *complex-valued* r.v. Z with a generalised Gaussian distribution as

$$p_{\text{GG,c}}(Z) = \frac{1}{D_2} \exp\left\{ - \left(\frac{|Z|}{\hat{\sigma}\, C_2} \right)^f \right\} , \tag{6.1}$$

with

$$C_2 = \left[\frac{\Gamma(2/f)}{\Gamma(4/f)} \right]^{1/2} \text{ and } D_2 = \pi\, \Gamma(1 + 2/f)(C_2 \hat{\sigma})^2 \, . \tag{6.2}$$

See Appendix C.2 for a derivation. We will refer to this version of the GG pdf as *model 2*. Figure 6.2 shows both pdf models in the complex plane. We observe that the new model, shown by a dark grey surface, exhibits a spikier centre and flatter tail than the original model, i.e. more probability mass near the origin and less mass in regions far from it.

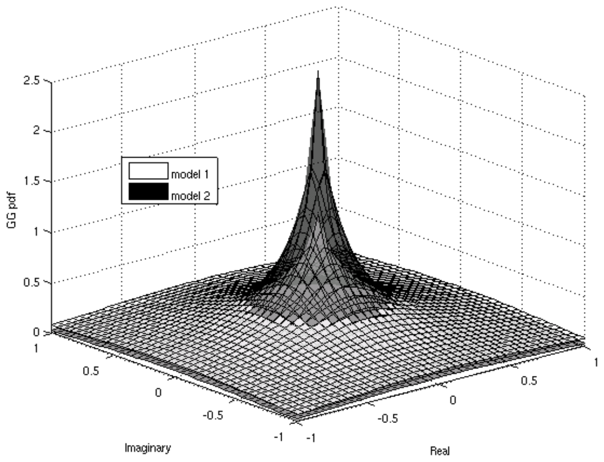


Figure 6.2: The old and new GG pdf model in the complex plane (both with a shape parameter of 0.7 and scale parameter of 1.0)

6.2.1.1 Negentropy and loglikelihood for pdf model 2

We will now give expressions for the negentropy assuming pdf model 2. In Appendix E the differential entropy for a generalised Gaussian r.v. is shown to be

$$H_{\mathrm{d}}(Y_{\mathrm{GG}}) = \log[2\pi\hat{\sigma}^2 C_2^2 \Gamma(2/f)/f] + 2/f\,, \tag{6.3}$$

with the Gaussian case specialising to

$$H_{\mathrm{d}}(Y_{\mathrm{gauss}}) = 1 + \log \pi\sigma^2\,. \tag{6.4}$$

The regularised negentropy optimisation criterion in terms of the complex pdf parameters and the current weight vector's size therefore becomes:

$$\begin{aligned} \mathcal{J}(Y,\alpha) &= H_d(Y_{\mathrm{gauss}}) - H_d(Y_{\mathrm{GG}}) - \alpha\|\boldsymbol{W_a}\|^2 \\ &= 1 + \log\left(\pi\sigma_Y^2\right) - \log\left[2\pi\,\hat{\sigma}_Y^2\, C_2^2\,\Gamma(2/f)/f\right] \\ &\quad -\frac{2}{f} - \alpha\|\boldsymbol{W_a}\|^2\,. \end{aligned} \tag{6.5}$$

The loglikelihood function for the new pdf model (6.1) is derived in Appendix D.2 to have the same format as for model 1 (Eq. (4.21)), this time setting $C = C_2$ and $D = D_2$ as defined in Eq. (6.2). From this, we can obtain the ML estimate of the pdf scale parameter as before. For a given shape parameter f, the optimal $\hat{\sigma}$ in the maximum likelihood sense is

$$\hat{\sigma} = \frac{1}{C_2}\left[\frac{f}{2K}\sum_{k=0}^{K-1}|Y(k)|^f\right]^{1/f}\,. \tag{6.6}$$

The ML estimate therefore has the same format as Eq. (4.22), now setting $C = C_2$ and $E = 2$ for model 2.

Specialising (6.6) for the Gaussian case by setting $f = 2$, we have

$$\hat{\sigma}_{\text{Gauss}} = \left[\frac{\Gamma(1)}{\Gamma(2)}\right]^{-1/2} \left[\frac{1}{K}\sum_{k=0}^{K-1} |Y(k)|^2\right]^{1/2} = 1 \cdot \left[\frac{1}{K}\sum_{k=0}^{K-1} |Y(k)|^2\right]^{1/2} . \quad (6.7)$$

We therefore arrive at the familiar moment method estimate for the variance,

$$\sigma_Y^2 = \frac{1}{K}\sum_{k=0}^{K-1} |Y(k)|^2 , \quad (6.8)$$

which means that for the Gaussian pdf with a complex r.v., the ML estimate of the variance equals the moment method estimate, as before for the real r.v. case.

Whether the theoretically more accurate model leads to changes in WER is discussed further below in Section 6.2.4.

6.2.2 Using the Same Variance For Both Entropies

The next change to the original MNB is, like the change in pdf, theoretically motivated. The definition of negentropy requires p_{Gauss} and p_{GG} to have the same variance, which would imply setting the scale parameter to the square root of the Gaussian variance estimate (estimated by any method). This definition was not followed by the original MNB, since the scale parameter was set to the ML estimate over the frames of the test utterances, while the variance was set to the moment estimate over the same frames.

It will be demonstrated below that following the definition of the negentropy when using the exact negentropy in combination with an offline shape parameter estimate is problematic. However, when using the empirical entropy, which is the final change to be introduced in this chapter, the same problems do not occur. The effect on WER of using this more efficient online estimation variant, referred to as [**SQRT-GAUSS**], will therefore be evaluated in Section 6.2.4.

6.2.3 Empirical vs. Exact Differential Entropy

In Section 4.2.1 the empirical differential entropy (4.8) was introduced as an approximation to the exact differential entropy. Instead of an analytical expression involving an overall parameter estimate, it calculates the entropy over a set of samples, which in our case mean the subband sample values per speech frame. The MNB variants to be introduced in Chapter 7 depend on frame-dependent parameter estimates, so they cannot use the exact empirical entropy, which would require a single parameter estimate. In order to separate the effects of using the empirical rather than exact entropy on the one hand, and the new frame-dependent parameter estimates on the other hand, we investigate the effect of the switch to the empirical entropy version in this chapter. As it turns out, under certain conditions the empirical and exact entropy are actually equivalent, but there are other cases where the use of the empirical entropy can make a difference to the WER.

6.2.4 ASR Results With Modified MNB Beamformers

The ASR results obtained for all possible combinations of the last three changes to the original MNB are shown in Table 6.2. It has the same structure as the previous results table 6.1, omitting the shape parameter estimation time since it is the same for all experiments in the table and in any case not relevant to the issues discussed. We begin the interpretation of the results by examining the effect of the new pdf model by comparing experiments E2035 with E2032, both with exact negentropy, and E2034 with E2036, both with the empirical version. In the first case model 2 achieves a slightly better WER, and in the latter it improves results by approximately half a percent. Both differences are however not statistically significant according to the MAPSSWE test. The fact that the theoretically more accurate model 2 does not lead to significant improvements deserves further discussion. It seems that the difference in the fine structure of the two pdf models can partly be reduced through an adjustment of the shape factors. Figure 6.3 plots models 1 and 2 on a log-scale in 2D by setting the imaginary part of the r.v. to zero, with three different shape parameters for model 2. As in the 3-D figure 6.2, model 2 has a peakier centre and less probability mass in the tail than model 1 for the same shape parameter. More importantly the figure also shows that, within the same model, the former is an effect of reducing f and the latter of increasing f. When comparing the offPE results for model 2 with those obtained for model 1, it turns out that the estimated shape parameters for model 2 are consistently approx. 0.2 lower than those for model 1. This suggests that an approximation between the

		Neg-	Estimation methods			WER
ID	Model	entropy	Gaussian	GG $\hat{\sigma}$	GG f	
			σ^2 (onPE)	(onPE)	(offPE)	%
E2035	1	exact	ML=MM	ML	ML-LS-ML	13.35
E2032	2	exact	ML=MM	ML	ML-LS-ML	13.31
E2034	1	empirical	ML=MM	SQRT-GAUSS	ML-LS-ML	13.65
E2036	2	empirical	ML=MM	SQRT-GAUSS	ML-LS-ML	13.14
E2053	1	exact	any	SQRT-GAUSS	any offPE	(˜DSB)
E2053c	2	exact	any	SQRT-GAUSS	any offPE	(˜DSB)
E2033 (=E2035)	1	empirical	ML=MM	ML	ML-LS-ML	13.35*
E2039 (=E2032)	2	empirical	ML=MM	ML	ML-LS-ML	13.31*

Table 6.2: Combinations of pdf model, negentropy type and parameter estimation methods, with word error rates (WER) where applicable. The WER values marked with asterisks are theoretically equivalent to another experiment.

two models is indeed taking place, and that the more significant part happens in the tail of the distribution. If that is the case, the insignificant change in WER might be explained by the fact that the values in the tail of the distribution influence the entropy sum (4.8) less than the values near the mean. Concluding the discussion of the pdf models, it should be noted that both model 1 and 2 are very similar, and the question arises whether a SIRP model is the most appropriate data representation for the complex data at hand. Both models access the subband values through their magnitude, and it might be worthwhile investigating possible alternatives to the modelling of complex numbers. However, this line of investigation is outside the scope of this thesis, and model 2 was therefore used for all remaining experiments.

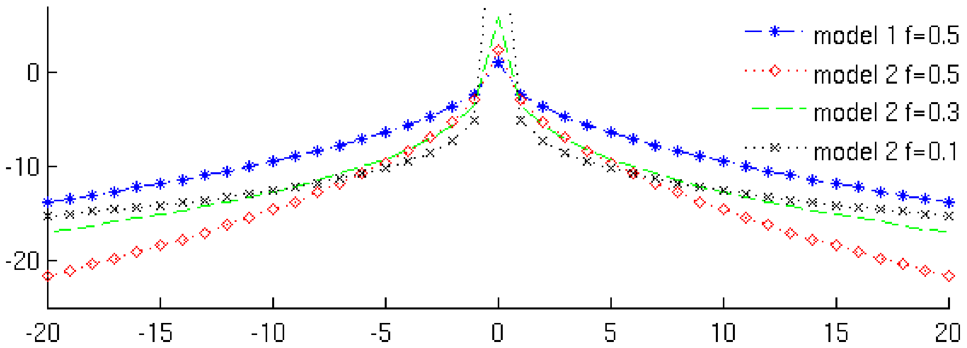


Figure 6.3: Comparison of GG pdf models of various shape parameters.

Analysing Table 6.2 for evidence on the influence of the empirical rather than exact entropy, we note that exactly the same results are obtained for E2035 and 2033, and for E2032 and 2039. An analysis of the equations involved explain why this is the case. Using the ML estimate of the scale parameter in the empirical entropy equation of either GG pdf model turns out to be equivalent to using the differential entropy, assuming we use the standard moment estimate for the Gaussian variance. To confirm this for e.g. model 1, we substitute the ML estimate (4.22) with $E = 1$ and $C = C_1$ into the empirical entropy Eq. (4.9) and simplify to obtain

$$\begin{aligned} H_{\mathrm{e}}(p_{\mathrm{GG}}, \mathcal{Y}) &= \log D_1 + \frac{\sum_{k=0}^{K-1} |Y(k)|^f}{K \hat{\sigma}^f \, C_1^f} \\ &= \log D_1 + \frac{\sum_{k=0}^{K-1} |Y(k)|^f}{K \left[\frac{1}{C_1} \left(\frac{f}{K} \sum_{k=0}^{K-1} |Y[k]|^f \right)^{1/f} \right]^f \, C_1^f} \\ &= \log D_1 + \frac{C_1^f \sum_{k=0}^{K-1} |Y(k)|^f}{K \left(\frac{f}{K} \sum_{k=0}^{K-1} |Y[k]|^f \right) \, C_1^f} \\ &= \log D_1 + 1/f \\ &= \log \left[2\Gamma(1 + 1/f) C_1 \hat{\sigma} \right] + 1/f \, . \end{aligned} \tag{6.9}$$

The final expression is equivalent to the exact differential entropy (4.6); the proof for model 2 can be done in an analogous way. The WER of the experiments E2033 and E2039 are therefore marked with an asterisk, as they result in the same weight vectors as E2035 and E2032 and represent duplicates. We will return to the question of how the empirical entropy affects WER further below.

Turning to the effect of using the SQRT-GAUSS online estimation method for the scale parameter, in case of the empirical entropy we find that while doing so slightly increased the error rate for the old pdf model 1 (E2033 vs. E2034), it slightly decreased it for model 2 (E2039 vs. E2036), resulting in the best WER of the set of experiments (13.14% for E2036). Again these differences are not statistically significant, so we conclude that the simpler estimation SQRT-GAUSS method leads to comparable results when the empirical entropy is used. When using the exact differential entropy however, the use of SQRT-GAUSS is problematic and leads to WER rates close to the DSB baseline. The reason is of a theoretical nature: $\hat{\sigma}$ and σ cancel each other out in the differential negentropy equation, with the resulting negentropy term depending only on the offline shape parameter. To confirm this for e.g. model 1, set $\hat{\sigma} = \sigma$ in Eq. (4.13), yielding

$$\begin{aligned} H_{\mathrm{d}}(p_{\mathrm{GG}}, \mathcal{Y}) &= 1/2 - 1/f + \log \frac{\sqrt{2\pi}\,\sigma}{2\Gamma(1+1/f)\,C_1\sigma} \\ &= 1/2 - 1/f + \log \frac{\sqrt{2\pi}}{2\Gamma(1+1/f)\,C_1}\,. \end{aligned} \tag{6.10}$$

When the shape parameter is fixed, as it always is in our experimental configuration, this implies a constant negentropy value, and when using the regularised negentropy, the only factor affecting the optimisation will be the size of the weight vector, resulting in a zero or near-zero weight vector $\mathbf{w}_{\mathrm{a},m}$. Since zero weight vectors correspond to the DSB solution, the WER is close to the DSB baseline value. This is true for both model 1 and 2, so we did not run the ASR step of experiment E2053 and and E2053c and marked the WER with (˜DSB) instead.

In combination with the duplicates discussed above, this means that we have no regular minimal pair comparisons for the effect of the empirical entropy on WER, neither for model 1 nor model 2. We can however validate the combination of SQRT-GAUSS online estimation with the empirical entropy as a successful configuration for the MNB, with model 2 providing a small WER improvement, which was however not significant for the size of the test set used.

6.3 Conclusions

Let us summarise the findings of the above investigations and the implications for further ASR experiments. Assuming we always use offline shape parameters and the moment estimate for the Gaussian variance, we can conclude the following points:

- The simpler parameter estimation method ML-LS-ML has been validated as being more efficient and providing a WER comparable to previous methods.

- The theoretically motivated pdf model 2 provides comparable WER results to model 1, and an analysis of the pdf parameters suggests that this robustness stems from an adjustment of the offline shape parameters. We note that both model 1 and model 2 use the same approach of modelling the complex data through the real-valued magnitude though, and further investigation into alternative representations might be of interest.

- When using the empirical entropy, following the definition of negentropy closely by setting the online scale parameter estimate to the square root of the variance (SQRT-GAUSS) leads to comparable results as the ML estimate, which requires additional computation. However, in combination with the exact negentropy and offline shape parameters it leads to beamformer weights similar to the DSB weights and in turn a significantly worse WER than obtained with any other MNB method.

- Regarding the use of the empirical or exact differential negentropy, no minimal pair comparison was possible, but a number of combinations have been identified as duplicates or degenerate cases. Best results were obtained with the empirical version in combination with SQRT-Gauss shape parameter estimation and model 2.

All further work will therefore use pdf model 2, the offline shape parameter estimated with the ML-LS-ML method, the empirical negentropy, and apply SQRT-GAUSS by setting the shape parameter to the square root of the variance. How exactly the variance is estimated will vary though, with the differences between the various methods being the content of the following chapter.

Chapter 7

Modelling the Nonstationarity of Speech

In order to calculate the negentropy over the beamformer outputs, the Gaussian and GG pdf parameters have to be estimated first. The original MNB uses a fixed per-utterance estimate for this purpose, i.e. a value that is constant for all frames of an utterance. However, speech is highly non-stationary, so a better model might be obtained by accounting for these short-term variations. The development of time-dependent parameter estimates to reflect this idea is the topic of this chapter.

All of the pdf parameters could be made time-dependent, but in the present work the changes will be restricted to the parameters estimated in an online fashion. Since the shape parameter is estimated off-line in all cases as done in the original MNB process, this means focusing on the GG scale parameter and the Gaussian variance estimate for the moment. It was pointed out in Section 6.2.2 that the definition of negentropy requires the variance of the Gaussian and GG pdf to be the same, and since the GG scale parameter is the square root of the GG variance, implementing this definition means that a scale parameter estimate automatically determines a variance estimate and vice versa. This approach will be used in all of the remaining experiments, and both variance and scale parameter are therefore determined by a single estimation, which we will refer to as variance estimation. This chapter describes how several different methods of deriving a time-dependent online estimate of the variance are implemented and then evaluated through ASR experiments. We will use the same labelling notation for the various estimation methods as in Chapter 4 of introducing each method identifier in bold and square brackets when it is first defined.

7.1 Global Variance Baseline

The baseline for the various variance estimation methods is given by the original MNB work's approach, which used the sample variance σ^2 calculated over all frames of the test utterance, Eq. (4.19). For conciseness we generally leave out the frequency bin subscript m, but all variance estimates should be understood as being utterance and frequency-dependent. We will refer to this baseline as the *global variance* baseline, where the term global refers to the entirety of the test utterance. The global variance method is labelled ML=MM as before, and the corresponding experiment E2036 provides the baseline WER of 13.14% shown in Table 6.2. As mentioned, a potentially problematic aspect of the global variance baseline is that it does not reflect the nonstationarity of speech. The features for phonemes differ substantially in their statistical characteristics such as mean and variance, in fact standard ASR systems depend on this very fact to function (cf. Chap. 2). The validity of a variance estimate based on a global average for all frames is therefore questionable, and in the remainder of this chapter several other variance estimation methods will be proposed with the aim of reflecting the changing nature of speech.

7.2 Moving Average Window of Fixed Length

An improvement over the global variance estimate might be expected from the use of a a local variance estimate based on a moving window of fixed length around the current frame. This would provide a frame-dependent estimate that might reflect the characteristics of the speech frames more closely. To evaluate this approach, we use a central moving average of N frames to either side of the current frame, i.e. the variance is calculated over a window of length $2N + 1$, as described by

$$\sigma^2(k) = \frac{1}{(2N+1)} \sum_{n=k-N}^{k+N} |Y(n)|^2.$$

For the initial and final frames no previous or following frames are available, so we could either ignore those frames or shorten the window on the side where frames are missing. Doing the latter results in the following expression using the lower bound l and upper bound u:

$$\sigma^2(k) = \frac{1}{L} \sum_{n=l}^{u} |Y(n)|^2, \quad L = u - l + 1 \tag{7.1}$$

where

$$l = \begin{cases} k - N & \text{if } k - N >= 0, \\ 0 & \text{otherwise,} \end{cases} \qquad u = \begin{cases} k + N & \text{if } k + N < K, \\ K - 1 & \text{otherwise.} \end{cases} \tag{7.2}$$

As before, K denotes the total number of frames of an utterance, and the frame index k ranges from 0 to $K - 1$. The identifier [**MovAvg-*L***] is introduced to refer to this moving average method, where L is replaced by the length of the window. Note that in the negentropy calculation, frames with a variance estimate of zero, which may occur at the beginning and end of recordings, are ignored, since the division by $\hat{\sigma}$ in the GG pdf (6.1) would lead to an error. This approach is used by any of the frame-dependent variance estimation methods described in this work.

Due to the sampling period of 16 kHz and effective frame shift of 256 samples used in our experiments, a window of N frames corresponds to $N \cdot 256$ samples or $(N \cdot 256)/16$ milliseconds (ms) of speech. In an analysis of the hand-labelled TIMIT data set [Garofolo et al., 1993], Wang et al. [1996] found the average phone length to be 60-70 ms, so a window of 5 frames (2 frame to either side of the current frame), corresponding to a window length of 80 ms, approximates this value. However, since this number of frames is very low, longer windows of 11, 21 and 51 frames were also tested. Table 7.1 shows the WERs obtained with the variance estimates based on the fixed windows of different lengths and compares them to the baseline MM=ML, which used all frames of the utterance (on average 6.7 seconds).

ID	Variance estimation	Window length in ms (frames)	WER %
E2036	ML=MM	6672 on avg. (417 frames on avg.)	13.14
E3020	MovAvg-51	816 (51 frames)	13.45
E3007	MovAvg-21	336 (21 frames)	13.38
E3009	MovAvg-11	176 (11 frames)	13.91
E3012	MovAvg-5	80 (5 frames)	14.90

Table 7.1: WER of global variance baseline and moving average models.

According to the MAPSSWE test, the only significant difference in the table is the increase in WER for the shortest window of 5 frames compared to any of the other window lengths and the baseline. The following points can be concluded: Firstly, shortening the global window for the variance estimate does not lead to a significant performance loss as long as the window is not extremely short. Even though a shorter window requires more computation over all frames than the global variance method, this result is very useful for a real-time implementation of the beamformer which would not have access to all frames, or time to process them. Secondly, although the shortest window of 80 ms comes closest to the average phone length reported by [Wang et al., 1996], it leads to a significantly worse performance. This could be due to a number of reasons:

(I) a small number of frames may not be sufficient to obtain a reliable variance estimate for the central frame, i.e. the problem might be data sparsity. After all, 5 values is a very small number to base a parameter estimate on.

(II) a very short window may be appropriate, but how short the window can be may depend on each phone. In other words, the fixed window length may not reflect the variation in phone lengths sufficiently, and lead to unrepresentative estimates for phones with longer average lengths.

(III) even if the estimate is a correct reflection of the central frame's estimate, a strongly localised variance estimate may not be beneficial in the given experimental context.

The following experiments are devised with the aim of eliminating the possible reasons for the increase of WER and providing alternative variance estimation methods. Of the three possible reasons, the third one is the most difficult to analyse and identify experimentally because it groups together the various assumptions, processing steps and relationships that hold in the time-dependent system but not the baseline system. For example, one aspect of the experimental context that may lead to the unsuitability of strongly localised variance estimates is the fact that the shape parameter is still a fixed offline estimate, not dependent on the utterance frames or any local window over them. In Sec. 6.2.4 we saw evidence that the two parameters can partly compensate for each other, so this influence could also be expected to play a role when changing the scale parameter values from a global to a local estimate. In the moving average experiments reported, the same shape parameter estimates were used as for the baseline case, which were obtained with the joint ML estimate ML-LS-ML over the complete utterance, as described in the previous chapter. As a first investigation, the offline shape parameter estimation was therefore repeated with the use of frame-dependent variance estimates obtained with the moving average window. The MovAvg-21 experiment was repeated with the resulting shape parameters in experiment E3019. The resulting WER of 13.59% does not constitute a significant change when compared to experiment E3007, which used the global shape parameters. However, in this configuration the shape parameters are still fixed per utterance, and in order to fully test the hypothesis that the fixed shape parameters cause the performance drop, the system would have to be changed so that both shape and scale parameter are frame-dependent. This modification was outside the scope of the current thesis, but remains an interesting investigation for the future.

7.2.1 Limitations of a Fixed Window Length

As mentioned above in point (II), the failure to achieve improved performance with a very short window may be due to the fact that the window length may not be appropriate for all phonemes. We could spend more time trying to find an optimal fixed window length for our given data, but apart from the problems associated with overfitting such an estimate to the data, the fact that the duration of a phoneme varies greatly from one phoneme class to another undermines the idea of a fixed window itself. While the aforementioned work of Wang et al. [1996] reported an average phone duration of 60-70 ms, it also found that the duration distributions were strongly influenced by contextual factors such as phone classes specified by long or short vowels, word stress, syllable position within the word and within an utterance, postvocalic consonants, and speaking rate. Given these variations, a fixed window length is certainly a compromise, and it seems more promising to base the estimate on variable-length segments of homogeneous frames, for example contiguous sequences of frames belonging to the same phoneme or part thereof. However, if we knew which frames belonged to which phoneme, we would already have solved the ASR task, so we use an idea from the LiMaBeam algorithm to obtain approximations to such a grouping [Seltzer, 2003]. LiMaBeam used an initial ASR decoding step involving a set of simple auxiliary HMMs to obtain an alignment of frames to phonemes, which it then used to calculate the beamformer's optimisation criterion. Having optimised the beamformer weights, they were applied to the array input data, and the resulting waveform was decoded to obtain the final transcription. Even though the initial decoding contained errors, this approach proved successful in terms of ASR results. The remainder of this chapter will describe ways of applying the same idea of using HMMs to derive information for the beamforming process to the maximum negentropy beamformer context.

7.3 Variable HMM-based Window

There are at least two ways in which hidden Markov models trained on a large corpus of speech data can be exploited in the variance estimation process of the MNB:

1. The results of a first decoding step with the HMMs can be used to obtain an alignment of frames to HMM states, which correspond to sub-phoneme units exhibiting similar statistical characteristics. As explained above, this would provide a homogeneous grouping of frames into segments for which the same variance estimate can be used. Assuming the alignment is sufficiently correct, this system tests hypothesis (II) that the fixed length of the window is problematic.

2. We can make use of the statistics stored by the HMMs themselves to derive a variance estimate. Since each standard HMM state is associated with a cepstral mean, which is a transformed version of the variance of the training data assigned to this state, we can reverse the transformation to obtain a variance estimate for all test utterance frames assigned to this state. Given that an HMM-derived estimate would be based on much more data, this variant indirectly investigates hypothesis (I) that the small number of frames used for the variance estimate is problematic. However, the derivation of test data statistics from training data is only approximative, and it remains to be seen if this has a detrimental effect.

To begin with, two estimation methods were implemented based on the first idea of using the segmentation into variable-lengths windows provided by a first decoding pass to obtain a variance estimate. For the estimation method [**HMM-WIN-STATE**], all the frames assigned to the same state of an HMM were grouped together into a local window and assigned the same variance estimate based on this window. More specifically, the following procedure was carried out:

1. An initial beamforming step combines the different channels into a single one. In the experiments of this study we used the original MNB for this step, but a different beamformer could also be used.

2. We then use our speaker-adapted ASR system to decode the single channel test utterance, and obtain an alignment between each frame k and an HMM state $s(k)$ with the Viterbi algorithm.

3. Let $S_{s(k)}$ be the set of all contiguous frame numbers around the current frame k which have been assigned to the same HMM state $s(k)$. Formally,

   $$S_{s(k)} = \{k\} \cup \{n : (s(n) = s(k)) \wedge ((n-1) \in S_{s(k)} \vee (n+1) \in S_{s(k)})\} .$$

   The variance for frame k is then set to the sample variance over the frames associated with this set:

   $$\sigma^2(k) = \frac{1}{|S_{s(k)}|} \sum_{n \in S_{s(k)}} |Y(n)|^2 . \qquad (7.3)$$

4. For each bin, the iterative optimisation procedure determines the active weights that maximise the negentropy over all frames for a given utterance. In each iteration i, the empirical negentropy is calculated for the beamformer output Y^i based on the current active weights w_a^i, where the negentropy calculation uses the frame-dependent variance estimates of the previous step. The shape parameter estimates were obtained off-line, as before.

Figure 7.1 depicts the procedure for a single utterance, with the iterative optimisation part shown within the area enclosed by a dashed line. Note that the quiescent weight vector and blocking matrix have been omitted in the diagram for clarity, but the relationships are the same as depicted in Fig. 4.5, i.e. the beamforming output depends on both of them. With this in mind, a comparison of the two figures shows that the only difference to the original MNB lies in the variance estimate, which is now based on the variable length windows derived from the alignments obtained with the HMMs. The alignments used to determine the segmentation were initially obtained from the fourth decoding pass of the data beamformed with the original MNB, which we shall refer to as *MNB alignments*. Since these alignments contain some errors, another variant was also tested, involving so-called *oracle alignments* that were obtained by performing a forced alignment on the known true transcriptions of the test data. While these would not be available in a realistic testing scenario, they provide insight into how much of the performance change can be attributed to incorrect segmentations. The first two rows of Table 7.2 show the ASR results for both alignment types. We observe that experiments E3014a and E3014b both show a significant increase in error rate of approx. 2% compared to the baseline and moving average estimation methods, with no significant difference between the use of oracle alignments and those obtained from the realistic decoding step. Further inspection of the length of the segments resulting from the grouping by HMM state revealed an average length of 2.3 frames per segment in both cases. Based on the experience with the moving average window of 5 frames, it is not surprising to observe an even further increase in error for even shorter windows.

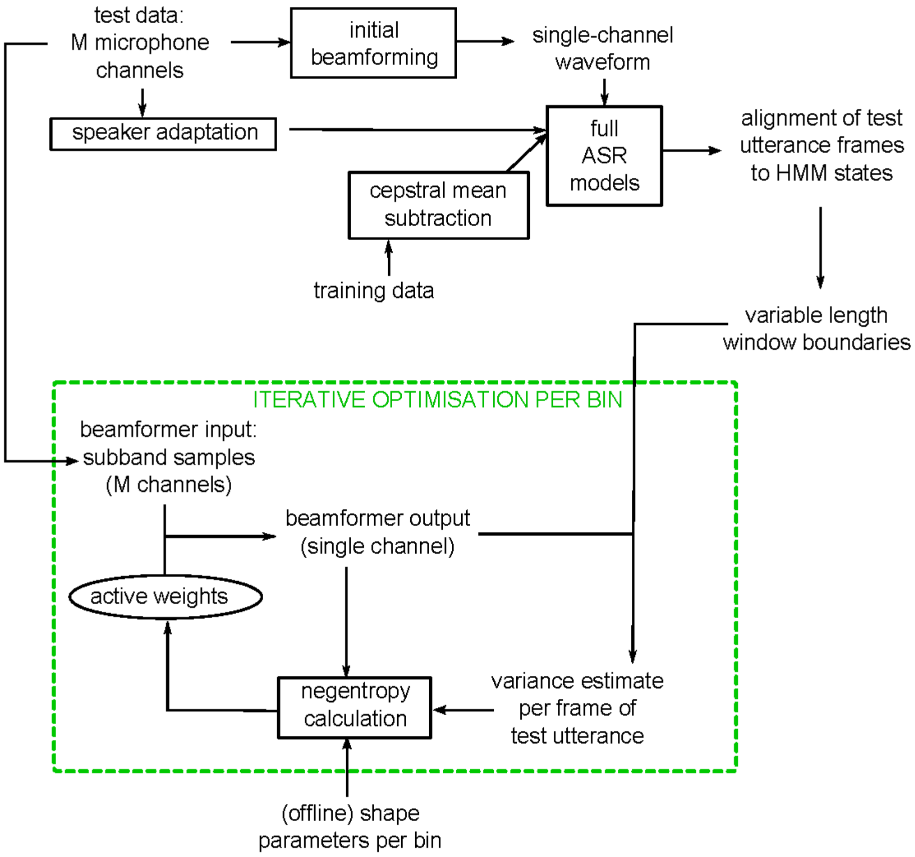


Figure 7.1: The optimisation process with variance estimation HMM-WIN-STATE or HMM-WIN-HMM, for a single utterance and bin.

ID	Variance estimation	Alignments used	Avg. window length	WER %
E3014a	HMM-WIN-STATE	MNB alignments	2.32	15.36
E3014b	HMM-WIN-STATE	oracle alignments	2.34	15.35
E3021	HMM-WIN-HMM	MNB alignments	5.88	14.8

Table 7.2: WER for variance estimate based on variable length HMM-alignment based windows.

A further method was therefore implemented which groups all frames assigned to the same hidden Markov model (i.e. triphone) [**HMM-WIN-HMM**]. Experiment E3021 tested this segmentation method, and resulted in an average segment length of 5.9 frames or 94 ms. This window length is a little higher than the average phone length estimated by Wang et al. [1996], but not substantially. In terms of ASR results, it achieved a significantly better WER of 14.8% than the windows based on HMM states, but still much worse than the baseline WER of 13.14% and not significantly different from the 14.9% of E3012, which used the moving average window of length 5. It can therefore be concluded that a grouping of subband samples by triphones as based on the alignments obtained with HMM decoding does not lead to ASR improvements and gives comparable results to a fixed window of the same length. We therefore discard hypothesis (II) that the fixed length of the moving average window was the cause of the performance deterioration and that windows adapted to the length of each phoneme would be more suitable.

7.4 Hidden Markov Model Maximum Negentropy Beamforming

We now turn to the second idea involving the use of HMMs, the derivation of a frame-dependent variance estimate from the statistical parameters of HMMs, which have been trained on a large corpus of speech and should therefore not suffer from the problem of data sparsity. For consistency with [Rauch et al., 2008], the resulting system will be referred to as the *hidden Markov model maximum negentropy beamformer* (HMM-MNB). The overall procedure proposed to optimise the active weights with HMM-based variance estimates for a given test utterance is depicted in Fig. 7.2. Compared to the variance estimate based on the variable-length windows derived from the HMM alignments, the variance estimation method is changed to set the variance to the PSD value reconstructed from the single cepstral mean associated with an *auxiliary* HMM state aligned with the kth frame. This process, including the multiplication with a ‘cepstral mean addition term’, is now described in more detail.

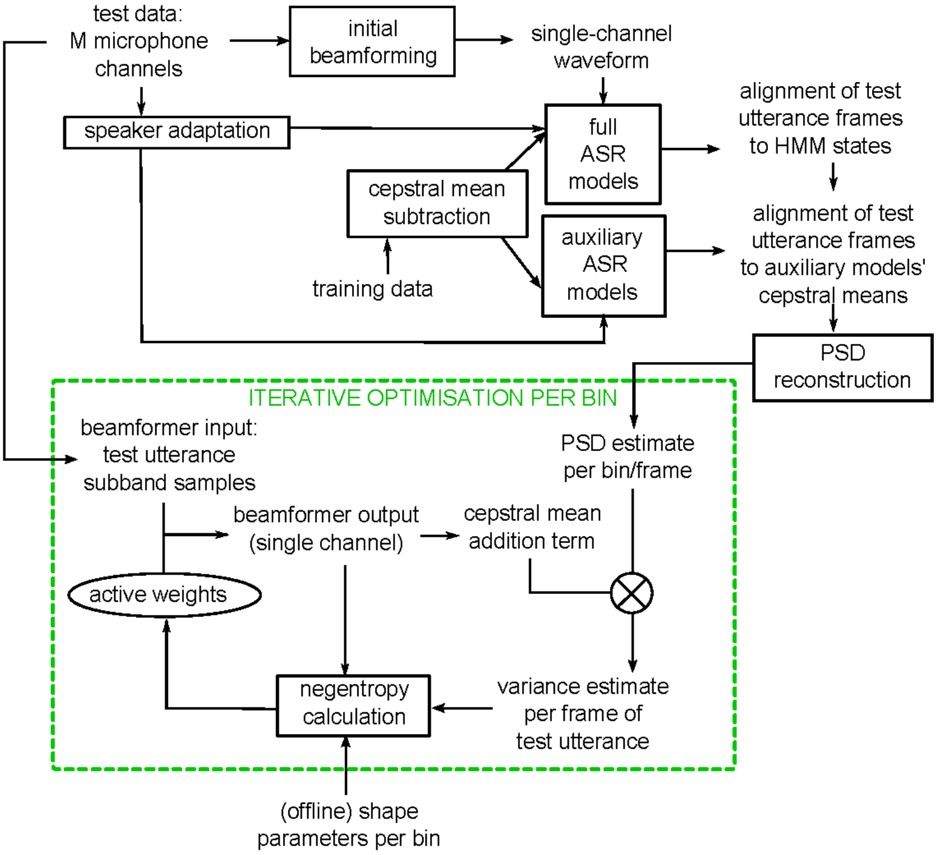


Figure 7.2: The HMM-MNB process for a single utterance and bin.

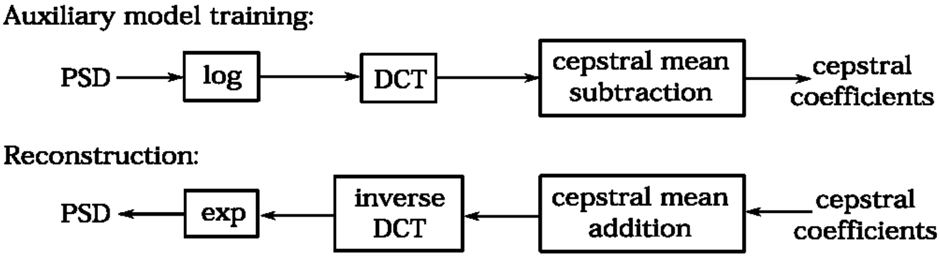


Figure 7.3: The PSD reconstruction process.

7.4.1 Reconstruction of the Power Spectral Density

The calculation of the negentropy requires an estimate of the variance of the beamformer output in the subband domain, $\sigma_m^2(k) = \mathbb{E}\{|Y_m(k)|^2\}$. As remarked in Sec. 2.2.4, the power of a zero-mean signal provides an estimate of its variance, and we can apply this idea to the subband samples. Our aim is therefore to obtain an estimate of the PSD per frame, and this section will describe how such an estimate can be derived from a set of auxiliary hidden Markov models.

The idea underlying the HMM-MNB's PSD reconstruction is to train a set of auxiliary models with the same front-end as the beamforming data, and then reverse the processing chain by applying the inverse transformations to the means of the Gaussian pdfs associated with the auxiliary HMM states that are aligned with a test utterance frame, thereby obtaining a variance estimate for the frame. The actual alignment of frames to states is done with the full HMMs used for ASR in order to get good accuracy, but the alignments can be transferred from the full to the auxiliary HMMs because both are based on the same context-dependency information. While it would in principle be possible to train the auxiliary models on the subband samples directly, these features would not be an ideal representation for the training process and lead to suboptimal phoneme statistics. Cepstral features are a successful representation for ASR which allow the use of cepstral mean subtraction, and they are based on transformations that can be inverted, so the auxiliary models are trained on cepstral features.

The upper part of Figure 7.3 shows the transformations from PSD to cepstral coefficients in the front-end of the auxiliary HMMs, and the lower part contains the corresponding inverse transformations that are needed for the reconstruction of the PSD from the HMM means. To describe the reconstruction process formally we begin with the construction of the cepstral vectors. Let $\mathbf{Y}_C(k)$ and $\mathbf{c}_C(k)$ denote the k-th vectors of subband samples and cepstral coefficients of an utterance from the corpus C, respectively, where C is either the test or training corpus. Their relationship can be expressed as

$$\mathbf{c}_C(k) = \mathbf{T}_\nu \log |\mathbf{Y}_C(k)|^2 \,, \tag{7.4}$$

where $\mathbf{T}_\nu$ is the Type 2 *discrete cosine transform* (DCT) matrix which has been truncated to ν *rows*, where ν is typically in the order of 13. This step leads to the final feature vector of length ν per frame. The components of the Type 2 DCT matrix can be expressed as

$$[\mathbf{T}]_{k,l} \triangleq \cos\left[\frac{\pi}{N}\left(l+\frac{1}{2}\right)k\right] \;\; \forall\; k,l = 0,\ldots,N-1.$$

In Eq. (7.4), the square magnitude and logarithm are calculated individually for each component $Y_{C,m}(k)$ of $\mathbf{Y}_C(k)$. Like the DFT output, the DCT output represents the strength of the components of the input at various frequencies, and this spectrum of the log spectrum is referred to as the cepstrum of the original signal, hence the name cepstral features or coefficients. Ignoring the higher cepstral coefficients by truncation means ignoring the higher frequencies present in the log spectrum. The final vector $\mathbf{c}(k)$ therefore only models the spectral envelope due to the resonances of the vocal tract and not the finer ripples caused by the fundamental frequency, thereby separating ASR-relevant from irrelevant information. After the logarithm and the DCT, the third transformation is the subtraction of the mean intended to normalise for short term channel effects. The calculation of the cepstral mean $\bar{\boldsymbol{\mu}}_C$ over the K frames of an utterance is described by

$$\bar{\boldsymbol{\mu}}_C \triangleq \frac{1}{K}\sum_{k=0}^{K-1}\mathbf{c}_C(k) = \frac{1}{K}\mathbf{T}_\nu\sum_{k=0}^{K-1}\log\left|\mathbf{Y}_C(k)\right|^2 . \tag{7.5}$$

The cepstral coefficients of a test utterance frame can therefore be expressed as

$$\bar{\mathbf{c}}_{\text{test}}(k) = \mathbf{c}_{\text{test}}(k) - \bar{\boldsymbol{\mu}}_{\text{test}} , \tag{7.6}$$

and the cepstral vectors $\bar{\mathbf{c}}_{\text{train}}(k)$ of an utterance used to train the auxiliary models as

$$\bar{\mathbf{c}}_{\text{train}}(k) = \mathbf{c}_{\text{train}}(k) - \bar{\boldsymbol{\mu}}_{\text{train}} . \tag{7.7}$$

The result of the training with these latter frames are the cepstral means associated with each auxiliary HMM state s, denoted by $\boldsymbol{\mu}_{\text{HMM:s}}$. Now let $\hat{\boldsymbol{\mu}}_{\text{HMM}}(k)$ be the speaker-adapted mean of the auxiliary HMM state aligned to the k-th frame of the test utterance by the Viterbi algorithm, which will be the starting point for the HMM-MNB's PSD reconstruction. Turning to the formal description of this reconstruction, let us for the moment assume that we are starting the process with the cepstral frame obtained from a test utterance, $\bar{\mathbf{c}}_{\text{test}}(k)$. The first step of the reconstruction process is to account for the CMS by adding back its cepstral mean, $\bar{\boldsymbol{\mu}}_{\text{test}}$. The second step is the inversion of the DCT matrix multiplication, which can be achieved by a multiplication with the inverse DCT matrix. The inverse $\mathbf{T}^{-1}$ of the Type 2 DCT matrix is equivalent to a Type 3 DCT matrix whose components have all been scaled by a factor of $2/M$ where M is the number of subbands used for beamforming. As in the construction of the cepstral frames, the matrix must be truncated to ν *columns*, and the truncated inverse matrix is denoted by $\mathbf{T}_\nu^{-1}$.

The final step is the inversion of the logarithm by use of the exponential function. Taken together and resubstituting Eq. (7.6) and (7.4), we have the transformation

$$\begin{aligned}\boldsymbol{\sigma^2}_{\text{PSD-TEST}}(k) &= \exp\left[\mathbf{T}_\nu^{-1}(\bar{\mathbf{c}}_{\text{test}}(k) + \bar{\boldsymbol{\mu}}_{\text{test}})\right] \\ &= \exp\left[\mathbf{T}_\nu^{-1}\mathbf{c}_{\text{test}}(k)\right] \\ &= \exp\left[\mathbf{T}_\nu^{-1}\mathbf{T}_\nu \log|\mathbf{Y}_{\text{test}}(k)|^2\right] \qquad (7.8) \\ &\approx |\mathbf{Y}_{\text{test}}(k)|^2 \,, \qquad (7.9)\end{aligned}$$

where $\boldsymbol{\sigma^2}_{\text{PSD-TEST}}(k)$ is the vector of variance estimates $\sigma_m^2(k)$ for the k-th frame produced by this method, which will be labelled [**PSD-TEST**]. As with the square magnitude and logarithm, the exponential operation in Eq. (7.8) is applied component by component. For further reference, we introduce the term *combined reconstruction matrix* for the matrix product $\mathbf{T}_\nu^{-1}\mathbf{T}_\nu$. The reconstruction of the PSD (7.8) is approximative only due to the truncation of the cepstral vectors and DCT matrices in this term. If we used the full matrices, the reconstruction would be exact since the combined reconstruction matrix would be the identity matrix, $\mathbf{T}^{-1}\mathbf{T} = \boldsymbol{I}$. The actually reconstructed value of Eq. (7.8) however is a smoothed version of the original PSD (an example of both unsmoothed and smoothed PSD will be shown later in Fig. 7.4).

The above reconstruction is based on the cepstral coefficients of the test utterance. The HMM-MNB however does not use these values, but the auxiliary models' means $\hat{\boldsymbol{\mu}}_{\text{HMM}}(k)$ instead. The reconstruction process is therefore described by the expression

$$\begin{aligned}\boldsymbol{\sigma^2}_{\text{PSD-HMM}}(k) &= \exp\left[\mathbf{T}_\nu^{-1}(\hat{\boldsymbol{\mu}}_{\text{HMM}}(k) + \bar{\boldsymbol{\mu}}_{\text{test}})\right] \\ &= \exp\left[\mathbf{T}_\nu^{-1}\hat{\boldsymbol{\mu}}_{\text{HMM}}(k) + \mathbf{T}_\nu^{-1}\bar{\boldsymbol{\mu}}_{\text{test}}\right] \\ &= \exp\left[\mathbf{T}_\nu^{-1}\hat{\boldsymbol{\mu}}_{\text{HMM}}(k)\right] \cdot \exp\left[\mathbf{T}_\nu^{-1}\bar{\boldsymbol{\mu}}_{\text{test}}\right] \,, \qquad (7.10)\end{aligned}$$

where $\boldsymbol{\sigma^2}_{\text{PSD-HMM}}(k)$ denotes the vector of the variance estimates produced by this method, for which we introduce the identifier [**PSD-HMM**].

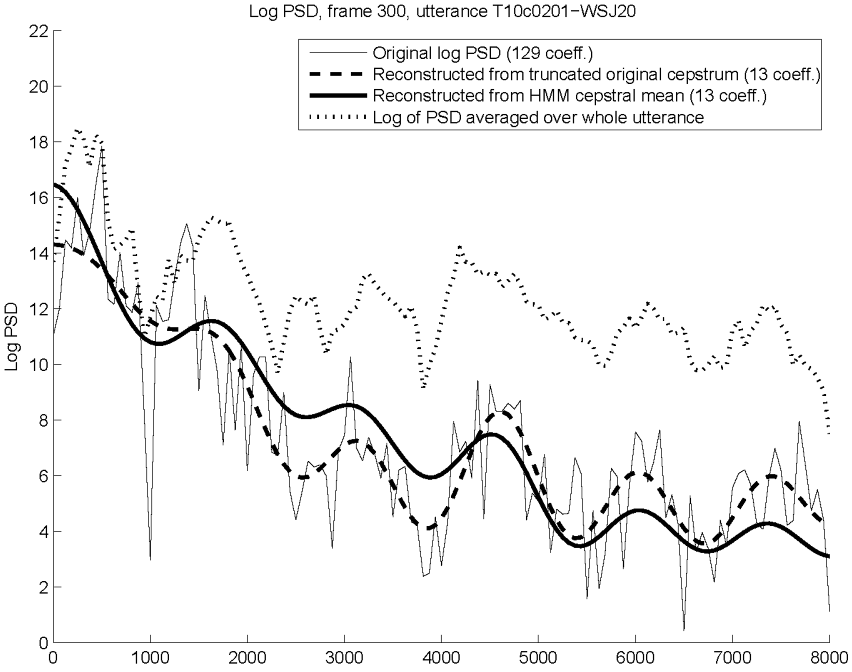
Note that the only term in (7.10) dependent on the changing beamformer output of the test utterance is the second exponent

$$\bar{\boldsymbol{\eta}} \triangleq \mathbf{T}_\nu^{-1}\bar{\boldsymbol{\mu}}_{\text{test}} = \frac{1}{K}\mathbf{T}_\nu^{-1}\mathbf{T}_\nu \sum_{k'=0}^{K-1} \log|\mathbf{Y}_{\text{test}}(k')|^2 \,. \qquad (7.11)$$

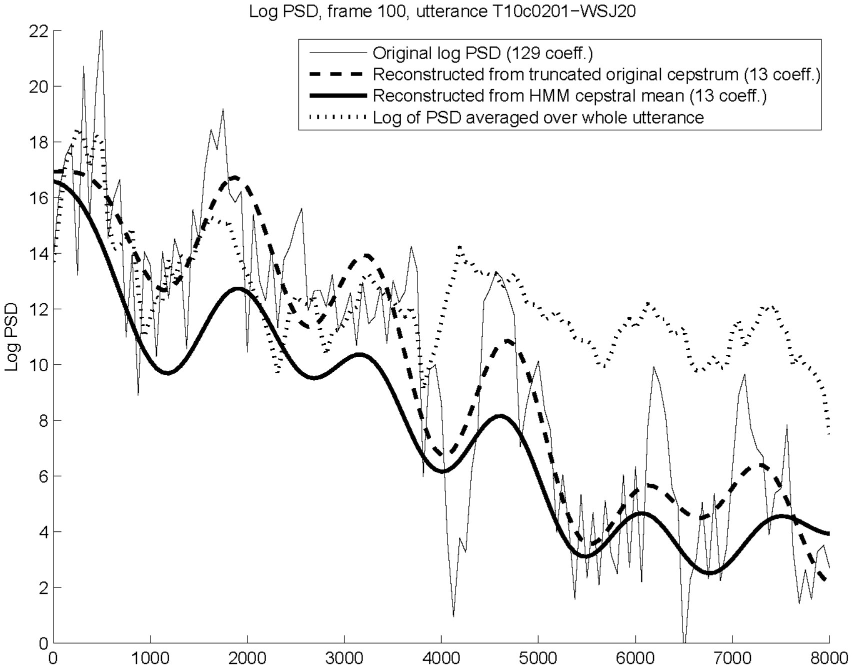
The PSD reconstruction process can therefore be split into two parts, one of which is independent of the optimisation (the first exponential term in Eq. (7.10)), and one which is dependent on each optimisation iteration: the calculation of the 'cepstral mean addition term'

$$\boldsymbol{\eta} = \exp\left[\mathbf{T}_\nu^{-1}\bar{\boldsymbol{\mu}}_{\text{test}}\right] \,. \qquad (7.12)$$

Both terms can then be multiplied together to obtain the final variance estimate to be used in the negentropy calculation, as illustrated in Fig. 7.2.

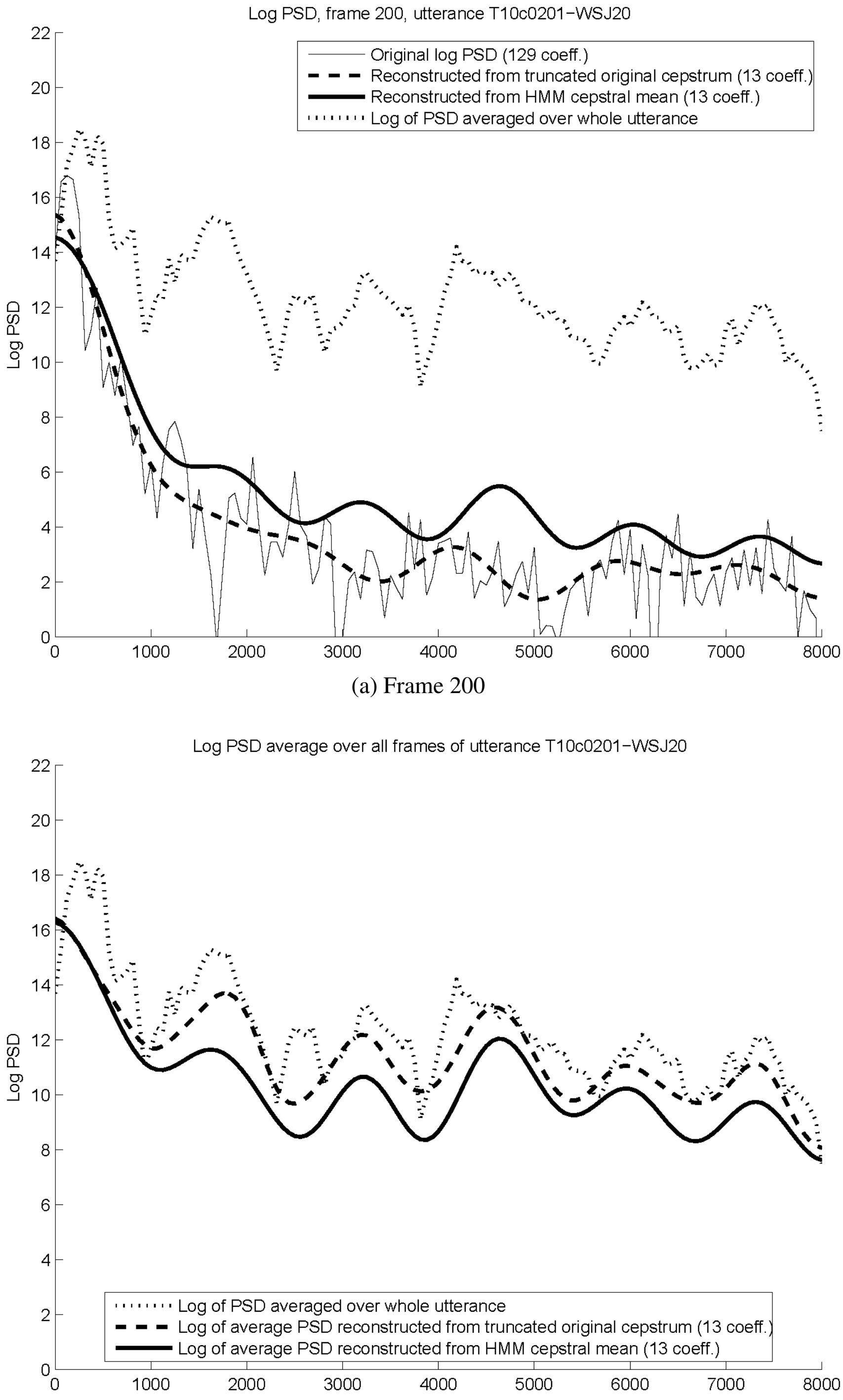


(a) Frame 300



(b) Frame 100

Figure 7.4: Original and reconstructed log PSD for single frames of a test utterance.



(a) Frame 200

(b) Averages over all frames of the test utterance.

Figure 7.5: Original and reconstructed log PSD of a test utterance.

Figures 7.4(a) and (b) and 7.5(a) show examples of the log PSD for one frame of a test utterance beamformed with the MNB, as obtained with the different estimation methods presented so far. The original PSD values are shown with a thin solid line, and the average PSD over all utterance frames (corresponding to the global variance baseline ML=MM) with a thin dotted line. These are compared to two smoothed versions obtained by reconstructing the PSD from 13-dimensional cepstral coefficients: the bold dashed line shows the PSD reconstructed from the utterance's cepstral coefficients for the frame (PSD-TEST), while the solid bold line plots the PSD values reconstructed from the cepstral mean in the auxiliary HMM state aligned with this frame (PSD-HMM). We observe that in the log PSD domain the HMM-based reconstruction, while sometimes lower or higher in value than the original PSD, approximates the spectral envelope of the frame relatively well. The estimate obtained by averaging the PSD over the entire utterance, on the other hand, models only the long-term spectral tilt, and thus does not capture the non-stationarity of human speech. Figure 7.5(b) plots the logarithm of the averaged estimates over all frames of the utterances, i.e. additionally to the average test utterance PSD that is also shown in the previous three plots, the averages of the estimates based on the reconstructed PSDs is also shown. Again we observe that the reconstructed estimates provide a smoothed version of the original PSD average, with the HMM-based reconstruction leading to slightly lower values than the reconstruction based on the test utterance cepstrum.

7.5 Problem Of Large Negentropy And Non-Convergence

The HMM-MNB procedure was implemented and executed, but due to a problem which occurred during beamforming no meaningful ASR evaluation of the system as proposed above was possible. This section will examine the problem in more detail and evaluate several possible solutions. To do so we will report the results for the first test utterance, but the behaviour has been confirmed for other utterances.

In all of the previous MNB experiments the simplex optimiser converges within the maximum number of iterations (set to 2000) for all bins. In the new HMM-MNB experiments however, not all of the bins converge. For example, for the first test utterance (exp. 3010a) only 11 of the 128 bins converge within this limit. Increasing the maximum number of iterations to 30,000 increased the number of converged bins to 95 (exp. 3026a); however, 33 bins still did not converge within this very large limit. Inspection showed that the weights were growing ever larger for 22 of the non-converging bins, a problem which increasing the maximum number of iterations cannot solve. The magnitude of the MNB's final weights generally lies in the interval $[0, 5]$; for example, for the global variance baseline exp. 2036, the minimum value of all 14 weights over all utterances and all bins is -2.8302, the maximum 2.203 and the mean magnitude (excluding the all-zero weights of bin 0) 0.152. Considering the final HMM-MNB weights of exp. 3010a for the first test utterance, here the minimum lies at approx. $-4.6 \cdot 10^{20}$, the maximum at $2.4 \cdot 10^{20}$

and the mean magnitude at $1.2 \cdot 10^{18}$. These values are clearly degenerate and not the result of an optimisation process terminated prematurely due to the number of iterations. Before we examine the reasons for the enormous weight growth further, we introduce a modification intended to speed up future experiments by detecting unusually large weights before the maximum number of iterations is exceeded. A weight size limit was implemented which leads to the termination of the optimisation as soon as one of the weight vector components passes the allowed maximum value. Table 7.3 shows the effect of such a limit on the convergence behaviour of the HMM-MNB, again using the first test utterance as an example. The experiment identifier is given in the first column, followed by the maximum number of iterations, the maximum weight limit and the number of bins that converged for this configuration, out of the total of 128 bins processed. The number of bins which did not converge is listed in the fifth column, with the total number given in bold and the numbers in parentheses showing first the cases where non-convergence was due to the limit on the optimisation iterations (the number before the slash) and those where the new weight size limit led to the termination (the number after the slash). The sixth column lists the number of bins that did not converge whose final weight vectors had large components with an absolute value over 100. The final column shows the average number of iterations until convergence or termination for each bin, rounded to the next integer. The behaviour of the baseline experiment E2036 was included for comparison. Note that the converged bins had low weights (with an absolute value below 5) in all cases. The following observations can be made:

ID	Max. iterations	Weight size limit	Converged bins	Non-converged bins: **total** (due to max. iterations / weight size)	Non-converged bins with large weights	Avg. iterations
E2036	2000	none	128	**0** (0 / 0)	0	838
E3010a	2000	none	11	**117** (117 / 0)	23	1922
E3010b	2000	100,000	13	**115** (74 / 41)	41	1377
E3010c	2000	100	8	**120** (55 / 65)	65	1011
E3026a	30000	none	95	**33** (33 / 0)	22	11402
E3026b	30000	100,000	79	**49** (8 / 41)	41	5446
E3026c	30000	100	60	**68** (3 / 65)	65	2993

Table 7.3: The effect of changing the maximum number of iterations and imposing a weight size limit on the convergence behaviour of the HMM-MNB (first test utterance only). All experiments except E2036 used the variance estimation method PSD-HMM.

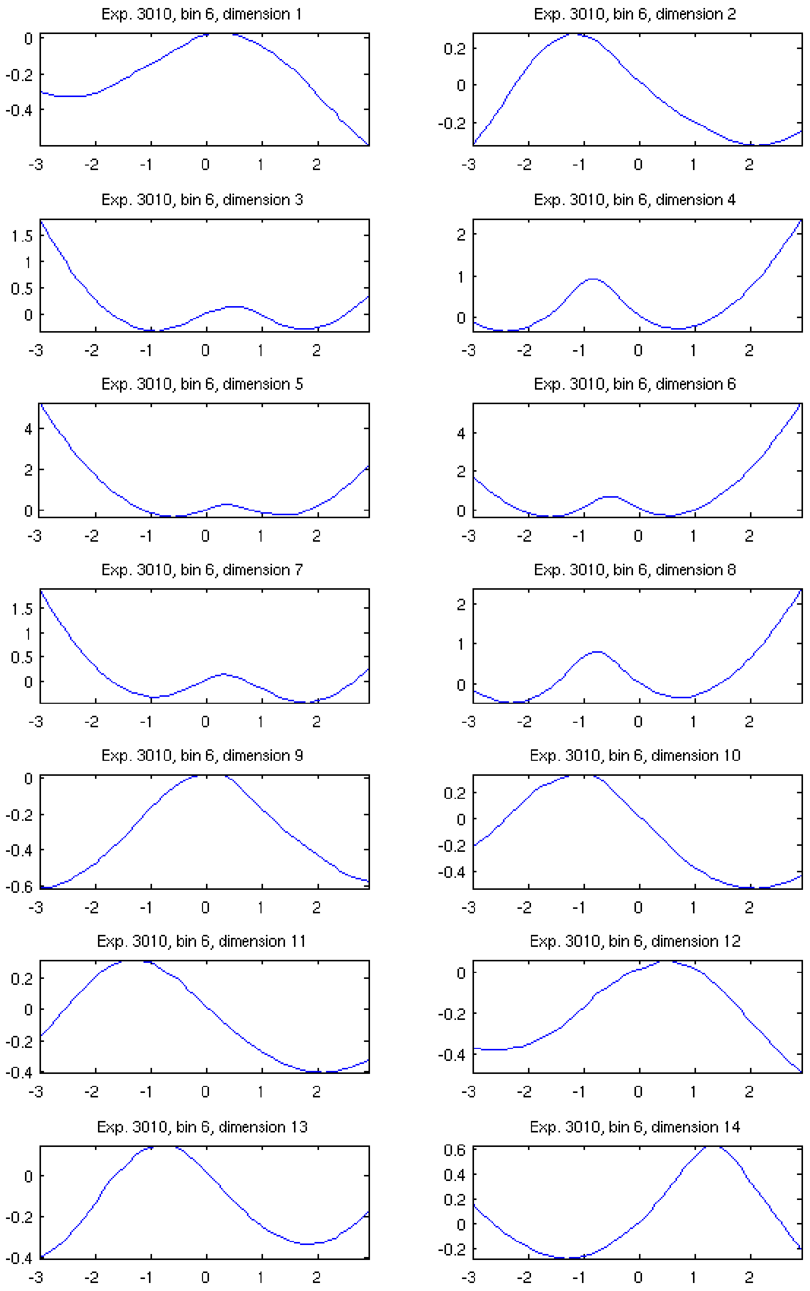
- Increasing the maximum number of iterations to 30,000 consistently increases the number of converged bins, but many bins take more than 2000 iterations to converge. Imposing a weight limit to terminate the non-converging bins can reduce the number of iterations again significantly.

- The numbers indicate that the assumption of independence of bins does not seem to hold. If the final weight vectors of the different bins did not influence each other, imposing a lower weight size limit should not increase the number of converged bins for the same limit on the iterations, as can be observed for E3010a and E3010b. In fact, given that all converged bins had low weights, the number of converged bins should not decrease either, and the number of bins for which large weights occur should not change. All of these cases be seen in the table, and the interdependence of bins can also be confirmed by comparing the final weight vectors of the converged bins, which do not stay the same. This issue will be looked into in Section 7.6.

- Since the number of bins with large weights is not affected significantly by the number of iterations, further experiments investigating this issue can be conducted with 2000 iterations at most, so that the experiments can be conducted in less time.

We can gain some insight into why some of the weight vectors grow so large by visualising the optimisation surface, i.e. by plotting the negentropy value against the weight vectors. Figures 7.6 and 7.7 show the negentropy optimisation surface of exp. 3010c for two different bins. Since a 14-dimensional space cannot be visualised easily, this is done by varying the value of a single weight in the range $[-3, 3]$ while keeping all other 13 dimensions fixed. This must be done around a point in the 14-dimensional space, so the initial zero weight vector was chosen for this purpose. In Fig. 7.6 we see the surface for bin 6, one of the bins that converged, and Fig. 7.7 plots the values for bin 10, a typical surface for one of the non-converging bins. The plots show clearly why the first bin converged, while the weights ended up growing larger and larger for the latter: the surfaces for bin 6 have a local maximum within the plotted range that the optimiser can find, while the surfaces for bin 10 exhibit a global minimum near the centre of the range, with the negentropy growing steadily the further away the weight moves from the minimum. This convex ‘cup shape’ is representative of the non-converging bins, and explains why the negentropy maximising optimiser chooses ever larger weights.

The experiment was repeated for two microphones instead of eight in order to verify that the ‘cup shape’ occurs over the whole optimisation surface and not just around a single point in the space. Since two microphones correspond to a single complex weight, we can plot the optimisation surface in 3 dimensions, with the real component plotted along the first axis and the imaginary component along the second axis. Fig. 7.8 shows such plots for bins 13-24 of the 2-sensor experiment 3011. In this experiment, 30 of the 128 bins converged, including the bins number 14, 15, 18 and 22 from the displayed range. These plots confirm that the increase of the negentropy with the size of the weight vector components is not restricted to



Exp. 3010, bin 6, dimension 1

Exp. 3010, bin 6, dimension 2

Exp. 3010, bin 6, dimension 3

Exp. 3010, bin 6, dimension 4

Exp. 3010, bin 6, dimension 5

Exp. 3010, bin 6, dimension 6

Exp. 3010, bin 6, dimension 7

Exp. 3010, bin 6, dimension 8

Exp. 3010, bin 6, dimension 9

Exp. 3010, bin 6, dimension 10

Exp. 3010, bin 6, dimension 11

Exp. 3010, bin 6, dimension 12

Exp. 3010, bin 6, dimension 13

Exp. 3010, bin 6, dimension 14

Figure 7.6: The optimisation surface in 1-dimensional plots along one dimension each: exp. 3010, bin 6.

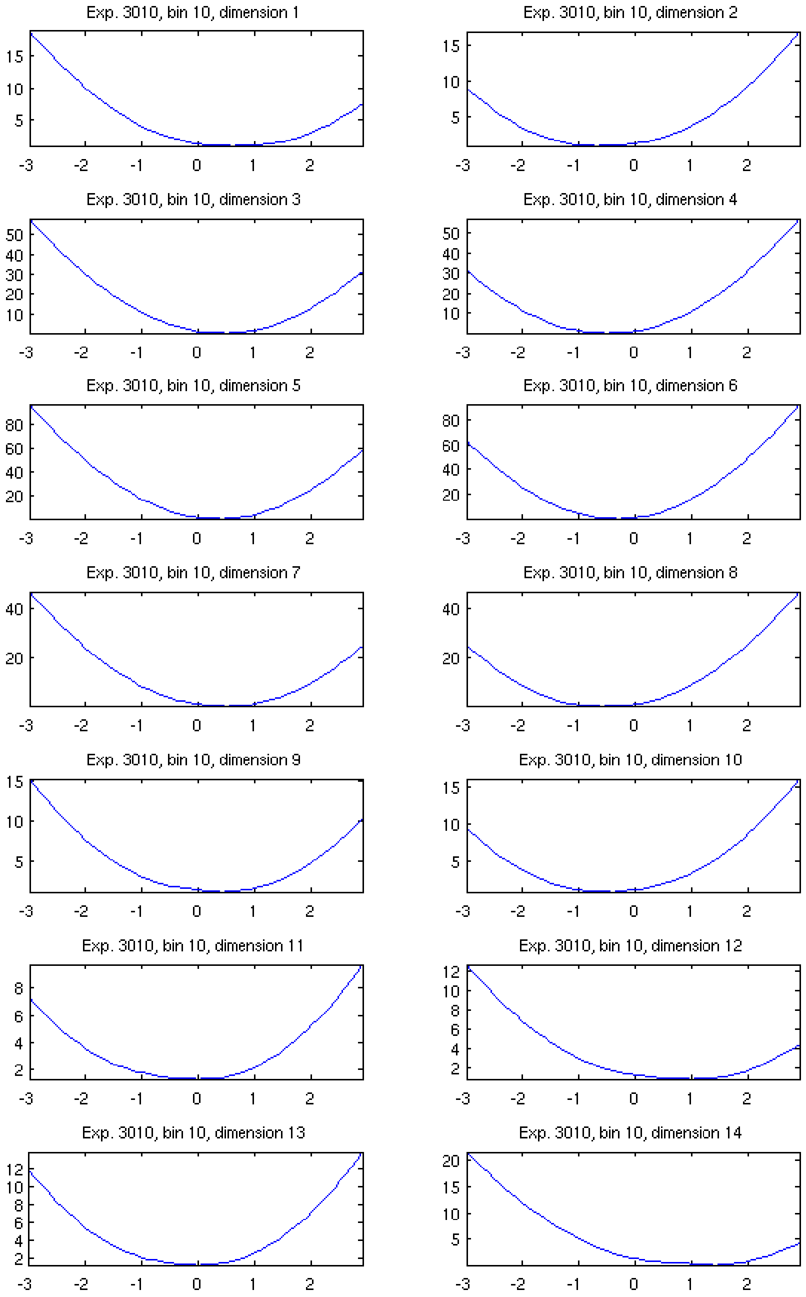


Figure 7.7: The optimisation surface in 1-dimensional plots along one dimension each: exp. 3010, bin 10.

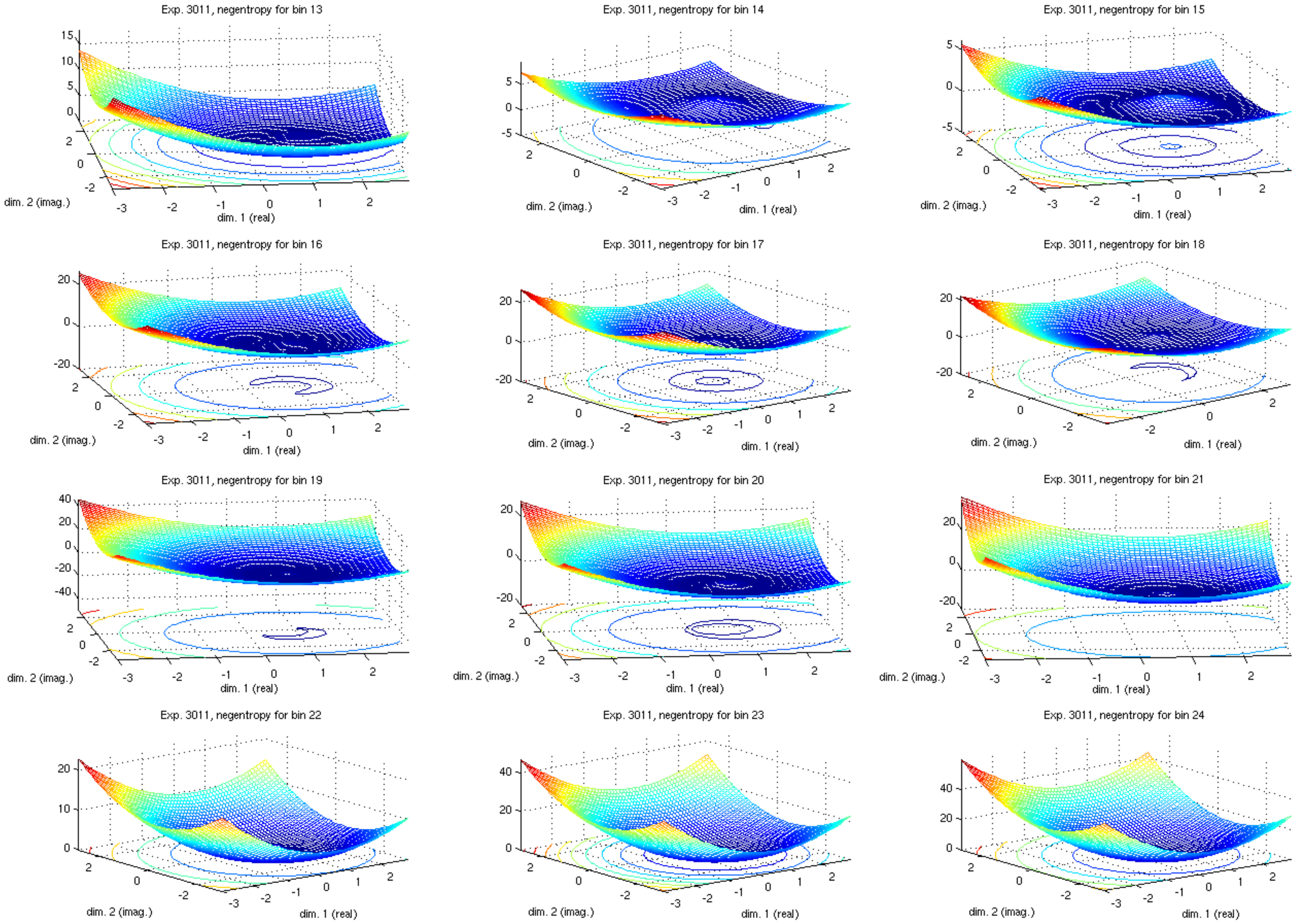


Figure 7.8: The optimisation surface for bins 13-24 in 3-dimensional plots: exp. 3011. Of these bins number 14, 15, 18 and 22 converged.

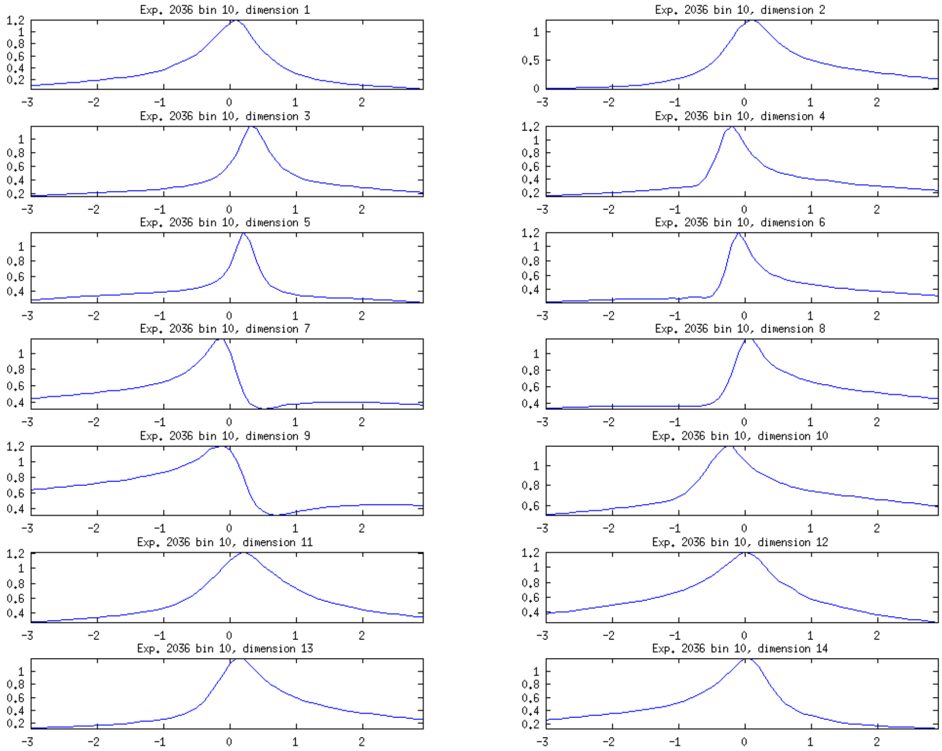


Figure 7.9: The optimisation surface for the baseline system: exp. 2036, bin 10.

a small area of the plane only, but a global phenomenon. For the converging bins, we can observe a small local optimum in the centre of the ‘cup’, similarly to the 14-D case shown in Fig. 7.6. To illustrate that the optimisation surface looks very different for the baseline system, Fig. 7.9 shows the plots for bin 10 as constructed during experiment 2036. As expected, the convex shape does not occur in this case.

The optimisation surfaces explain why the optimiser behaves the way it does, however they do not shed light on the reason for the convex shape itself. This can be done by inspecting the size of the components involved in the negentropy calculation. Recall that the negentropy is the difference between the Gaussian entropy and the GG entropy (cf. Sec. 4.2.2). Examining the details of the negentropy computation during the HMM-MNB optimisation, it can be seen that the Gaussian entropy component is responsible for the large negentropy values rather than the subtracted GG entropy. Through further inspection we can identify the term in the calculation of this entropy which grows excessively. Let us derive an expression for the GG (empirical) entropy and based on that, for the Gaussian entropy, to pinpoint this term and explain the problem. Substituting the GG pdf model 2 Eq. (6.1) with $Z = Y(k)$ and a frame-dependent scale parameter $\hat{\sigma}_k$ into the empirical entropy definition (4.8) yields

$$\begin{aligned}
H_{\mathrm{e}}(Y_{\mathrm{GG}}) &= -\frac{1}{K}\sum_{k=0}^{K-1}\log p_{GG,c}(Y(k)) \\
&= -\frac{1}{K}\sum_{k=0}^{K-1}\log\left[\frac{1}{D_2}\,\exp\left\{-\left(\frac{|Y(k)|}{\hat{\sigma}_k\,C_2}\right)^f\right\}\right] \\
&= -\frac{1}{K}\sum_{k=0}^{K-1}\left[\log\frac{1}{D_2}-\left(\frac{|Y(k)|}{\hat{\sigma}_k\,C_2}\right)^f\right] \\
&= -\frac{1}{K}\sum_{k=0}^{K-1}\left[\log\frac{1}{\pi\,\Gamma(1+2/f)(C_2)^2}-\log\hat{\sigma}_k^2-\left(\frac{|Y(k)|}{\hat{\sigma}_k\,C_2}\right)^f\right] \\
&= -\,\log\frac{1}{\pi\,\Gamma(1+2/f)(C_2)^2}+\frac{1}{K}\sum_{k=0}^{K-1}\left[2\,\log\hat{\sigma}_k+\left(\frac{|Y(k)|}{\hat{\sigma}_k\,C_2}\right)^f\right]\,.
\end{aligned}$$

By setting the shape parameter $f = 2.0$ we obtain the Gaussian entropy[1]:

$$H_{\mathrm{e}}(Y_{\mathrm{Gauss}}) = \log\pi+\frac{1}{K}\sum_{k=0}^{K-1}\left[2\,\log\hat{\sigma}_k+\left(\frac{|Y(k)|}{\hat{\sigma}_k}\right)^2\right]\,.$$

The term causing the excessively large negentropy values turns out to be the squared quotient term of the sum, $(|Y(k)|/\hat{\sigma})^2$. This term grows very large when the numerator is large and the denominator small. The log files of exp. 3010 confirm that the variance estimates of the HMM-MNB can be significantly smaller than those of the global variance case, and in combination with frames that have a large magnitude $|Y(k)|$, a very large negentropy value is the result. The larger the weights, the larger the beamformer output magnitudes, which in turn lead to large negentropy values, and the result is the observed 'cup shape' of the optimisation surface. Graphically, the effect of a large negentropy contribution of a frame for a small variance value and large magnitude can also be seen as computing the negentropy for a point in the outer regions of the plot in Figure 4.4(a).

The above results demonstrate the sensitivity of the negentropy calculation to variance estimates which are too small. Two questions arise: firstly, why the HMM-derived variance estimates are considerably smaller than the global variance estimates, and secondly, whether the effect can be corrected or avoided so that the HMM-derived information can still be used successfully. The smaller estimates could be due to a wrong assumption or to an inherent property of the process, such as the smoothing operation or the mismatch between the training and the test data. In the following sections we will test a number of different variance estimation methods to answer the two questions, beginning with the reexamination of an assumption that the previous results have already shown not to hold: the independence of the optimisation runs for each bin.

[1] Note that C_2 evaluates to 1 for f=2 and $\Gamma(2) = 1$.

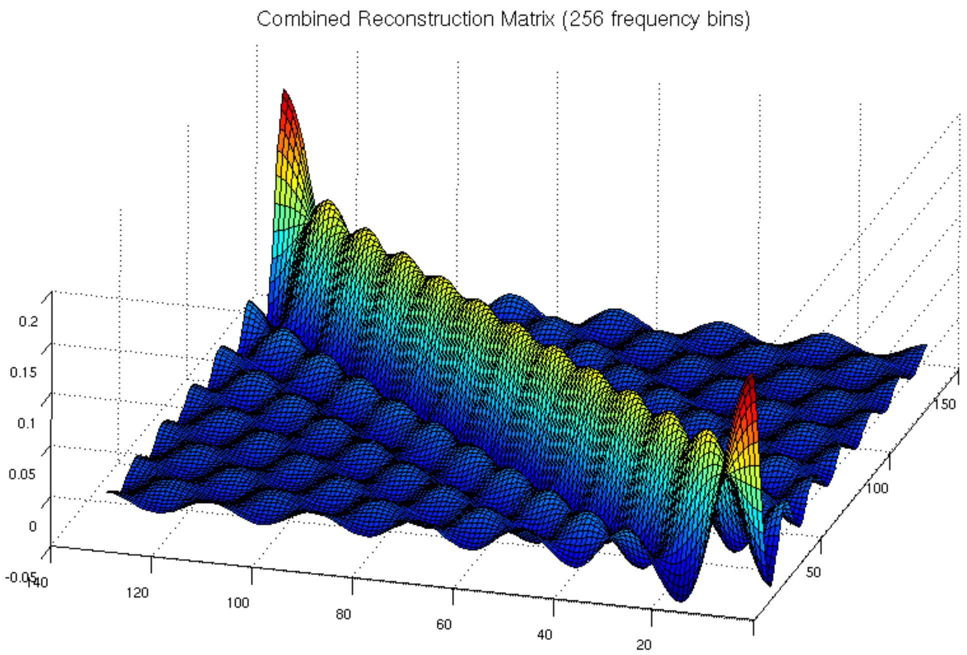


Figure 7.10: The combined reconstruction matrix $\mathbf{T}_{13}^{-1}\mathbf{T}_{13}$, assuming truncation to 13 cepstral coefficients.

7.6 Interdependence of Bins

As pointed out earlier, the results in Table 7.3 indicate an interdependence of the bin optimisations, contrary to the assumption on which the bin-by-bin process is based. Let us therefore reexamine the theoretical bases of the PSD reconstruction, specifically the combined reconstruction matrix $\mathbf{T}_{\nu}^{-1}\mathbf{T}_{\nu}$. While the smoothing effect seen earlier is an indication of the influence of adjacent bins on each other, we initially assumed that the matrix is diagonally dominant[2], with the implication that the off-diagonal terms are relatively small and their influence of one bin on the other can be ignored. However, as demonstrated by Fig. 7.10, which plots the values of this matrix for the 129 bins of an FFT length of 256 and the truncation of the DCT matrices to 13 coefficients, the matrix is in fact not diagonally dominant. In fact, approximately 9 entries to the left and right of the diagonal have relatively large values when compared to the remainder of the matrix. The plot shows this broader diagonal ‘ridge’ quite clearly. The implication of this matrix structure is that the reconstructed variances of the different bins are not approximately independent of each other, so a final weight vector of one bin affects the negentropy calculation of another. However, the influence of the adjacent bins gets weaker with distance from the diagonal, and it remains to be seen in how far the

[2] A matrix is called diagonally dominant if the magnitude of the diagonal entry in every row is larger than the sum of the magnitudes of all the other (off-diagonal) entries in that row.

dependence influences the results of the bin-wise optimisation with respect to ASR results, and the convergence problem in particular. We could account for the interdependence of bins by introducing a new HMM-MNB variant which optimises the weight vectors for all bins jointly. However, this approach is computationally very demanding. In our experiments we optimised 128 bins and a 14-dimensional weight vector, and this would result in a 1729-dimensional optimisation problem, which was not solvable by the system within a reasonable amount of time. Instead, two variants were implemented. The first one does the optimisation bin-by-bin as before, but computes the average negentropy over all 128 bins when optimising each bin. This method, labelled **[PSD-HMM-AVGBIN-ALL]**, restricts the search space by only varying the active weight vector of the current bin, but considers its effect on all other bins. While faster to compute than the joint optimisation over all bins, it is still computationally very expensive. An additional variant was therefore implemented which only averages the negentropy over the nine adjacent bins to either side of the current bin, reflecting the higher values along the diagonal of the reconstruction matrix. Including the current bin, 19 bins are averaged, and the method is therefore labelled **[PSD-HMM-AVGBIN-19]**. To express both methods formally, let us denote the regularised negentropy for bin m by $\mathcal{J}^m(Y, \alpha)$ as defined in Eq. (4.18), where as before Y is the beamformer output and α the regularisation constant. For the method PSD-HMM-AVGBIN-ALL, the optimisation criterion becomes the average over all processed bins:

$$\mathcal{J}^m_{\text{AVG-ALL}}(Y, \alpha) = \frac{1}{M'} \sum_{m=1}^{M'} \mathcal{J}^m(Y, \alpha), \quad M' = M/2\,, \tag{7.13}$$

where M is the number of all bins (in our experiments 256). For the method PSD-HMM-AVGBIN-19 we use a window of 9 adjacent bins around the current bin:

$$\mathcal{J}^m_{\text{AVG-19}}(Y, \alpha) = \frac{1}{L} \sum_{i=l}^{u} \mathcal{J}^i(Y, \alpha), \quad L = u - l + 1 \tag{7.14}$$

where the lower limit l and upper limit u are given by

$$l = \begin{cases} m-9 & \text{if } m-9 >= 1, \\ 1 & \text{otherwise,} \end{cases} \qquad u = \begin{cases} m+9 & \text{if } m+9 < M', \\ M' & \text{otherwise.} \end{cases}$$

Table 7.4 shows the convergence behaviour for the two new estimation methods, with the results of the PSD-HMM method with the same configuration repeated in the lower part for ease of comparison. We can observe the following from the table:

- The convergence behaviour for method PSD-HMM-AVGBIN-ALL has improved significantly. Most importantly, the number of bins with large weights is reduced substantially to approx. 1-5 bins, depending on the weight size limit.

ID	Max. iterations	Weight size limit	Converged bins	Non-converged bins: **total** (due to max. iterations / weight size)	Non-converged bins with large weights	Avg. iterations
PSD-HMM-AVGBIN-ALL: averaging negentropy over all 128 bins						
E3028a	2000	none	33	**95** (95 / 0)	1	1777
E3028b	2000	100000	36	**92** (87 / 5)	5	1690
E3028c	2000	100	123	**5** (0 / 5)	5	980
PSD-HMM-AVGBIN-19: averaging negentropy over 9 adjacent bins to either side						
E3029a	2000	none	36	**92** (92 / 0)	10	1703
E3029b	2000	100000	37	**91** (75 / 16)	16	1505
E3029c	2000	100	109	**19** (0 / 19)	19	866
PSD-HMM: no averaging, results from Table 7.3						
E3010a	2000	none	11	**117** (117 / 0)	23	1922
E3010b	2000	100,000	13	**115** (74 / 41)	41	1377
E3010c	2000	100	8	**120** (55 / 65)	65	1011

Table 7.4: Convergence behaviour for the variance estimation method PSD-HMM-AVGBIN (first test utterance only).

- The problem of large weights occurs more frequently for method PSD-HMM-AVGBIN-19, but still much less frequently than for PSD-HMM, for 10-19 bins.

Both methods are so computationally expensive that even with a more efficient optimisation routine and time spent improving the efficiency of the code, they do not seem suitable for a practical implementation in the near future. In any event, even though they reduce the convergence problem substantially, they do not solve it completely. We will therefore attempt to find alternative solutions, ideally involving the optimisation for a single bin to avoid time-intensive computations.

7.7 Increasing the Regularisation Constant

In all of the experiments reported so far, the regularisation constant was set to 0.01, as described in Sec. 4.3.2. Since the purpose of the regularisation constant is to prevent weights that are too large, increasing the regularisation constant might address the non-convergence problem. The three methods involving the reconstruction of the PSD described so far were therefore repeated with varying regularisation constants, ranging from 0 to 1.0. Table 7.5 shows that the convergence behaviour is not affected significantly by changing the optimisation constant. It appears that the convex shape of the optimisation surface is so strong that the given values of

ID	Max. iterations	Weight size limit	Regularisation constant	Converged bins	Non-converged bins: **total** (due to max. iterations / weight size)	Avg. number of iterations
PSD-HMM-AVGBIN-ALL						
E3040a	2000	100	0.0	124	**4** (1 / 3)	1015
E3028c	2000	100	0.01	123	**5** (0 / 5)	980
E3040b	2000	100	0.1	122	**6** (1 / 5)	972
E3040c	2000	100	0.5	125	**3** (0 / 3)	978
E3040d	2000	100	1.0	121	**7** (2 / 5)	961
PSD-HMM-AVGBIN-19						
E3032a	2000	100	0.0	107	**21** (0 / 21)	849
E3029c	2000	100	0.01	109	**19** (0 / 19)	866
E3032b	2000	100	0.1	108	**20** (0 / 20)	863
E3032c	2000	100	0.5	109	**19** (0 / 19)	875
E3032d	2000	100	1.0	109	**19** (0 / 19)	866
PSD-HMM						
E3033a	2000	100	0.0	6	**122** (57 / 65)	1021
E3010c	2000	100	0.01	8	**120** (55 / 65)	1011
E3033b	2000	100	0.1	7	**121** (55 / 66)	1003
E3033c	2000	100	0.5	8	**120** (57 / 63)	1040
E3033d	2000	100	1.0	9	**119** (56 / 63)	1030

Table 7.5: Effect of varying the regularisation constant on convergence behaviour (first test utterance only).

the regularisation constant, which have the opposite effect of imposing a concave tendency on the surface, cannot modify it significantly. Increasing the regularisation constant therefore does not solve the non-convergence problem, and since it constitutes an ad-hoc procedure, a more principled analysis of the problem with the aim of identifying and eliminating the cause of the problem will be conducted in the following section.

7.8 Different Approaches to the Convergence Problem

The non-converging bins occur because of variance estimates that are too small for some frames, and identifying and eliminating the cause for this should lead to the solution of the convergence problem. The plots of the reconstructed PSDs shown earlier suggest three possible reasons why the PSD-HMM estimate can be smaller than the baseline variance:

1. **Non-stationarity:** because the bin's PSD for a given frame differs from the average, it can be significantly lower than it. This can be seen in Fig. 7.5(a), where the PSD-HMM estimate is lower than the average for all bins.

2. **Smoothing:** due to the truncated DCT, the HMM-PSD estimate smoothes the rapid variations over the bins. Due to this smoothing, some bins will receive much lower variance estimates than they would in a non-smoothed estimate, as seen in a comparison of the non-smoothed baseline variance and the smoothed estimate based on the test utterance cepstrum in Fig. 7.5(b).

3. **Mismatch between test and training data:** Fig. 7.5(b) also shows that the PSD-HMM estimates are on average lower than the baseline estimates. This may be due to the fact that the training data is clean headset speech, whereas the test data is noisy, and clean speech could be expected to have a lower variance than noisy speech in general.

To analyse how these factors affect the convergence behaviour, Table 7.6 describes the baseline and the HMM-PSD estimation method along the following three dimensions: the kind of averaging performed for the variance estimate; whether test or training data is the basis for the estimate; and whether the subband samples were used directly, or whether a smoothed version was derived from cepstral coefficients. Row A describes the baseline system, which does not suffer from the non-convergence problem, and row B the HMM-MNB, which does. The remainder of the table lists all theoretically possible combinations of the settings of these two systems for the three dimensions, providing minimal pairs to investigate which of the three differences leads to the problem. The estimation method identifiers in the second column are marked in bold within square brackets only if they were not introduced before. Regarding the averaging operation, while an HMM-based variance estimation method does not do any averaging over the frames of the test utterance, the derived variance estimate is based on an average obtained during training; the table entries in rows B and H have therefore been marked with a comment indicating this. We begin the investigation by modifying a single setting in the table of the baseline row A, starting with the transformation the data undergoes before the estimate is done, i.e. the last column. Changing it to a cepstrum-derived estimate leads us to row C, whose variance estimate has been named AVG-PSD-TEST and will be defined formally below.

	Variance estimation method	Averaging	Utterances used	Type of data used
A	ML=MM	over all frames of utterance	test utterance	subband samples
B	PSD-HMM	none (but avg. during training)	training data avg. (HMM)	smoothed subb. samples derived from cepstrum
C	[**AVG-PSD-TEST**]	over all frames of utterance	test utterance	smoothed subb. samples derived from cepstrum
D	PSD-TEST	none	test utterance	smoothed subb. samples derived from cepstrum
E	[**INST-POW**]	none	test utterance	subband samples
F	[**AVG-PSD-HMM**]	over all frames of utterance	training data avg. (HMM)	smoothed subb. samples derived from cepstrum
G	[**ML=MM-TRAIN**]	over all frames of utterance	training data avg. (HMM)	subband samples - this would mean training the HMMs on subband data
H	[**INST-POW-TRAIN**]	none (but avg. during training)	training data avg. (HMM)	subband samples - this would mean training the HMMs on subband data

Table 7.6: Different variance estimation methods.

The estimation method PSD-TEST of Eq. (7.9) removes the HMM and averaging aspects from the estimation method PSD-HMM, since the truncated cepstrum is computed for each test utterance frame and the PSD is then reconstructed from it. The method AVG-PSD-TEST uses the same reconstruction process, but averages the PSD over all frames, in the same way as the global variance baseline averages over all frames. Compared to the baseline, the effect of the reconstruction from the truncated cepstrum is a smoothing of the average power spectrum over adjacent bins, as discussed before. An example of such a smoothed estimate is given by the dashed line in Fig. 7.5(b). Formally, the variance estimate vector of length M for all subbands of AVG-PSD-TEST is given by

$$\begin{aligned}\boldsymbol{\sigma^2}_{\text{AVG-PSD-TEST}} &= \frac{1}{K}\sum_{k=0}^{K-1}\boldsymbol{\sigma^2}_{\text{PSD-TEST}}(k)\\ &= \frac{1}{K}\sum_{k=0}^{K-1}\exp\left[\mathbf{T}_\nu^{-1}\mathbf{c}_{\text{test}}(k)\right]\\ &= \frac{1}{K}\sum_{k=0}^{K-1}\exp\left[\mathbf{T}_\nu^{-1}\mathbf{T}_\nu\log|\mathbf{Y}_{\text{test}}(k)|^2\right]\ . \qquad (7.15)\end{aligned}$$

For a single frequency bin m we can express its variance estimate σ_m^2 by

$$\sigma^2_{\text{AVG-PSD-TEST}}(m) = \frac{1}{K}\sum_{k=0}^{K-1}\exp\mathbf{t}_m^T\log|\mathbf{Y}_{\text{test}}(k)|^2, \qquad (7.16)$$

where $\mathbf{t}_m^T$ denotes the m-th row of the combined reconstruction matrix $\mathbf{T}_\nu^{-1}\mathbf{T}_\nu$. As before, when setting ν to the length of the full cepstral vector (in our experiments 129) instead of truncating it, the combined reconstruction matrix becomes the identity matrix, and the method becomes equivalent to the global variance baseline. A first AVG-PSD-TEST experiment was run with $\nu = 13$, which is the same value as used for the auxiliary methods of the HMM-MNB. The number of cepstral coefficients used for the reconstruction was then increased to 64 and 120 in two further experiments. Table 7.7 shows the convergence behaviour in the three cases. Its format mostly follows Table 7.3 with an added column for the number of cepstral coefficients ν replacing the average number of iterations. The table also shows the results for the methods PSD-TEST (E3038) and AVG-PSD-HMM (E3039), so that the reconstruction from the PSD both for the test utterance cepstrum and the HMM cepstrum is evaluated with or without averaging. We can make the following observations for the method AVG-PSD-TEST:

- The non-convergence problem also occurs in this case. For the originally used number of cepstral coefficients, approximately the same number of bins are affected by it as in the PSD-HMM case (compare E3022 with E3010b).

- The convergence speed does however seem faster for the converging bins than for the PSD-HMM cases (the number of converged bins in E3022 is much larger than for E3010b).

- The lower the number of cepstral coefficients, the more often bins have weights with large components. This relationship is not linear: setting the number of cepstral coefficients to 64 leads to an only slightly lower number of 40 problematic bins when compared to the 44 non-converging bins obtained with 13 coefficients. However, the speed at which the weights grow is slower in this case, which can be seen from the fact that although the 40

ID	Max. iterations	Weight size limit	Cepstral coefficients	Converged bins	Non-converged bins: **total** (due to max. iterations / weight size)	Avg. number of iterations
AVG-PSD-TEST						
E3034	2000	100	13	56	**72** (1 / 71)	549
E3035	2000	100	64	65	**63** (0 / 63)	555
E3036	2000	100	100	98	**30** (9 / 21)	939
E3037	2000	100	120	127	**1** (1 / 0)	946
PSD-TEST						
E3038	2000	100	13	66	**62** (0 / 62)	595
AVG-PSD-HMM						
E3039	2000	100	13	8	**120** (62 / 58)	1102

Table 7.7: Convergence behaviour for the variance estimation method AVG-PSD-TEST with different numbers of cepstral coefficients (first test utterance only).

non-converged bins all had large components (with an absolute value above 100), they did not reach the limit of 100,000 within 2000 iterations, as most bins did for in E3022.

- When using 120 cepstral coefficients for the reconstruction, none of the bins suffer from large weights. However, one bin does not converge within 2000 iterations.

For the PSD-TEST and AVG-PSD-HMM experiments the results are as follows:

- Using the estimation method PSD-TEST leads to 62 bins which do not converge due to their weight size. This is slightly less than E3034, the AVG-PSD-TEST experiment with the same configuration, so removing the averaging aspect improves the convergence to a small degree.

- The AVG-PSD-HMM exp. E3039 results in 120 non-converged bins, the same number of non-converged bins as for PSD-HMM experiment E3010c. However, there is a slight improvement in the number of bins with large weights, from 65 for E3010c without averaging to 58 bins with the averaging, so in this case removing the averaging leads to slightly worse convergence.

These results provide important insight into the non-convergence issue. Firstly, the averaging aspect does not seem to influence the convergence behaviour substantially, so this can be discarded as a possible reason for the non-converging

ID	Max. iterations	Weight size limit	Cepstral coefficients	Converged bins	Non-converged bins	Avg. number of iterations	WER
E3041	2000	100	129	128	**0**	803	32.35

Table 7.8: Convergence and WER of the variance estimation method PSD-HMM-ALLCEPS. The average number of iterations refers to the first test utterance only.

bins. Secondly, the smoothing over bins due to the truncation of the cepstral features introduces the problem into a system that did not suffer from it before. If we can show that the removal of the smoothing operation also removes the non-convergence issue from the HMM-MNB, then this would identify the truncation as the factor responsible for the non-convergence problem. One way to remove the smoothing would be to leave out the DCT operation and train directly on sub-band samples, as in method INST-POW-TRAIN identified in Table 7.6. However, in that case we could not apply CMS any more, so using all 129 cepstral coefficients when training the auxiliary HMMs seems a more promising way to remove the smoothing. Experiment E3041 implemented this approach, which we will label [**PSD-HMM-ALLCEPS**]. Table 7.8 shows the convergence results and WER, with the average number of iterations applying to the first utterance for comparison with the previous tables, but all other value to the set of all utterances. As we can see, all bins now converge, so we can conclude that the smoothing over bins introduced by truncating the cepstral coefficients is the cause of the non-convergence problem. However, unfortunately it does not lead to an improvement in WER. Experiment E3041 achieves a WER of 32.35%, which is considerably worse than any of the results seen with the baseline system or moving average windows. Possible reasons for this will be discussed in the next chapter.

Chapter 8

Summary and Future Work

The objective of this thesis was the development and evaluation of frame-dependent variance estimates in the negentropy calculation of the maximum negentropy beamformer, motivated by the idea that reflecting the non-stationary nature of speech in the parameters might lead to better speech modelling and thereby increased ASR accuracy. This objective has been partially met, and this chapter will summarise the results and discuss the remaining open questions.

Chapter 6 provided the background for the experimental evaluation. Comparative experiments confirmed that the following changes to the original MNB can be made without loss of accuracy:

- the shape parameter can be estimated with a simpler and more efficient joint maximum likelihood estimation method;
- the empirical negentropy, required by frame-dependent parameter estimates, can be used instead of the differential negentropy;
- the scale parameter can be set to the square root of the variance, as required by the definition of negentropy.

Furthermore, it was found that the generalised Gaussian pdf can be changed to account for a complex r.v. by using an expression which integrates to unity over the complex plane. However, in both cases the data was modelled through the magnitude of the complex subband sample value, which is itself a real number, and further research into alternative representations of complex data might be of interest.

In Chapter 7 frame-dependent variance estimates were introduced, beginning with a simple moving average estimate over a fixed length window around the current frame. This type of variance estimate proved robust in terms of WER for short windows up to 11 frames, resulting in no significant loss of accuracy when compared to the baseline. While this method cannot be used to improve the WER, it is however very relevant for real-time implementations of the maximum negentropy beamformer, since it shows that a result comparable to the WER of the global variance baseline (13.5-13.9%) can be achieved with only 5 frames (80 ms) of speech in advance of the current frame. However, shortening the window even further to 5 frames (2 frames in advance) negatively affected recognition results, increasing the WER to 14.9%. Three hypotheses were identified as to the cause of this increase:

- the inability of a fixed-length window to account for the varying length of the phonemes and the implication that a short window may lead to an inaccurate model for phonemes of longer duration;
- data sparsity, i.e. the insufficiency of such a small number of frames as a reliable sample basis for a variance estimate of the central frame;
- the unsuitability of a strongly localised variance estimate in the given experimental configuration.

The first hypothesis was contradicted by the results obtained with a variable-length window which grouped together all the frames assigned to the same phoneme by an initial HMM decoding pass, since this resulted in a performance drop of the same magnitude as the one observed for the equally short fixed-length windows. Motivated by the data sparsity hypothesis another way of incorporating HMM information was implemented: the variance estimate of each frame was derived from the statistical mean associated with the HMM state assigned to this frame by a first decoding pass. Since HMMs are trained on sufficient data, this method should not suffer from data sparsity. In order to be able to reconstruct the variance from the state mean, the same front-end as the beamforming module (rather than the ASR module) had to be used, so the means of a set of auxiliary HMMs were employed for this process. To ensure sufficiently accurate phoneme representations in these HMMs, standard truncated cepstral coefficients rather than subband sample values were used as feature vectors, which also allowed for cepstral mean subtraction.

When evaluating this HMM-MNB strategy, a problem occurred during beamforming which prevented any meaningful ASR evaluation. The optimisation routine applied to find the optimal active weight vectors for beamforming could not converge for some frequency bins because the negentropy values were increasing with each iteration. Furthermore, it was found that the assumption of independent bins did not hold, contradicting the bin-by-bin optimisation approach. Detailed analysis revealed that the growth occurred in the Gaussian entropy component of the negentropy for frames with large magnitude and a variance value that was too small (as compared to the baseline value). Three possible reasons for the insufficiently large variance values were identified, and after a series of experiments it could be shown that both the interdependence of bins and the non-convergence problem were due to the truncation of the cepstral coefficients, which corresponds to a smoothing operation of the variance over adjacent bins. As predicted, a final experiment without truncation of the cepstrum, i.e. using all 129 cepstral coefficients in the auxiliary HMMs, did not suffer from the non-convergence problem. All bins could be optimised, and the resulting waveforms could be evaluated through ASR experiments.

Unfortunately, although the non-convergence problem was solved by this method, the WER was much higher (32.35%) than the baseline or any of the fixed-length or variable-length windows evaluated before. This might be due to one of the following reasons:

1. The errors in the alignment between frames and HMM phones might lead to the use of unsuitable cepstral means. This seems unlikely since in earlier experiments the use of the oracle alignments instead of the MNB alignments made no difference to the results obtained with variable length windows derived from the alignments.

2. The mismatch between the training and test data might be problematic. The auxiliary models are trained on clean single-channel speech, while the test utterance is beamformed noisy speech. Even though speaker adaptation is applied to the models, a certain mismatch is still to be expected, which might lead to inaccurate modelling. To test this hypothesis, the auxiliary models could be trained on the test data beamformed with a basic set of weights, e.g. the DSB or global variance weights. The test set only consists of 352 utterances though, and this seems an insufficient amount of data for training triphones; changing the configuration to context-independent monophones might be required in this case.

3. The changing active weight vector only influences the cepstral mean addition term, but not the HMM-derived part of the PSD. If the optimisation routine, which evaluates the negentropy for many possible active weight vectors, is not passed a criterion value that sufficiently depends on the changing input, then the decision as to which weight vector is optimal can in itself be suboptimal. It is not obvious how the hypothesis that this factor is at the cause of the error increase can be tested, as the abstraction from the actual beamformer output frame value to the cepstral mean of the aligned HMM state is inherent in the method. In the implemented version, the alignment is only done once, and it would in principle be possible to redo it for each active weight vector, but this would be extremely expensive computationally, and the abstraction would still occur.

4. The increased error rate might be due to the fact that the auxiliary models trained with all cepstral coefficients provide a less suitable representation for speech recognition (or training, in this context), since they contain much irrelevant information; after all, the reason for discarding the higher coefficients was to achieve more suitable feature vectors for ASR. The consequence of using all cepstral coefficients would be a suboptimal representation of the phones by the cepstral means of the HMMs. One way to test this hypothesis could be to modify the training software so that it uses 13 coefficients when calculating the probabilities needed for the training, but stores

the full cepstral vector for the variance reconstruction. Alternatively, recognising the test data with the auxiliary models instead of the full models, and comparing the result to the WER obtained when decoding with the previous auxiliary models using truncated cepstral coefficients, would give an indication of how strongly the truncation affects the phoneme representations.

Note that even if further experiments showed that one of the above reasons causes the increase in word error rate, there is no obvious practical implementation that avoids the problem for any of them. In practice, the oracle alignments would not be available, training on the test data would not be an option, the third option is intrinsic to the proposed method, and truncating the cepstrum would lead to the non-convergence and interdependence problems described in the last chapter.

We therefore return to the list of hypotheses regarding the smaller increase in WER that resulted from the use of the moving average window. Assuming that the HMM-MNB method cannot be used to test the data sparsity hypothesis, the question arises whether there is any other way to determine if the lack of sample values is the cause of the original slight deterioration in WER or the short estimation window. One experiment that might shed light on this question would involve the use of every frame, not every second frame as done by the original MNB, for beamforming.

If data sparsity is not the root of the slight increase in WER, the third hypothesis remains, which suggested the unsuitability of strongly localised variance estimates for the given experimental context as the cause of the increase in WER to almost 15%. As mentioned before, this option is not specific enough to be tested as such, since it groups together all remaining unverified assumptions and processing steps that are made in the short window beamforming version but not in the global variance baseline. The following important points belong to this group:

- The shape parameters remain non-stationary throughout optimisation of a bin's weights. This point was already mentioned in Section 7.2. The global variance baseline also employs shape parameters independent of the beamformer output, with the scale parameters dependent on it, but both estimates are based on the entirety of an utterance, and this is not the case for the non-stationary estimates. An interesting line of investigation for the future therefore consists of the incorporation of time-dependent shape parameters in addition to the time-dependent variance estimates.

- Another assumption the experiments are based on is the super-Gaussian distribution of the clean speech samples as opposed to the more Gaussian distribution of the corrupted speech. This assumption might only hold for speech in general, but not for each phoneme separately, in which case a strongly localised variance estimate would not be suitable. Doing a forced alignment of the training data and compiling histograms of the power spectral values for each phoneme would provide this information, and this investigation is also planned for the future. If the distribution of single phonemes were found to be very different from that of speech overall, it might be more promising to attempt to approximate the actual distribution of each phoneme as observed on a training corpus of clean speech, instead of the super-Gaussian distribution observed for speech overall. However, this would require a sufficient number of frames to estimate the phoneme distribution with the beamformer output for each phone.

Having outlined the remaining open questions and possible lines of future investigations, we conclude that replacing the utterance-dependent variance estimates with time-dependent local estimates as presented in this thesis does not lead to an improvement in ASR performance, but that medium-length estimation windows can provide practical benefits for real-time implementations. In addition, the experiments of Chapter 6 led to the development of a more efficient MNB implementation.

Appendix A

ULA Frequency Response Function

To show the equivalence of the Equations (3.15) and (3.16) we need to show that

$$\sum_{n=0}^{N-1} e^{j\omega n d(\cos(\phi_t)-\cos(\phi))/c} = e^{-j(N-1)\alpha} \cdot \frac{\sin(N\alpha)}{\sin(\alpha)}\,, \tag{A.1}$$

where

$$\alpha = \omega d(\cos(\phi_t) - \cos(\phi))/2c\,.$$

In the proof, we will make use of the following well-known equivalence:

$$\sum_{n=0}^{N-1} x^n = \frac{1-x^N}{1-x}\,. \tag{A.2}$$

We will also use the following expression for $\sin(x)$ derived from Euler's formula $e^{-jx} = \cos(x) - j\sin(x)$:

$$\sin(x) = \frac{e^{jx} - e^{-jx}}{2j}\,. \tag{A.3}$$

Applying Equivalence (A.2) to the left hand side of Equation (A.1) with the substitution $x = e^{-j\omega d(\cos(\phi_t)-\cos(\phi)/c)}$, we obtain

$$
\begin{aligned}
& \sum_{n=0}^{N-1} e^{j\omega n d(\cos(\phi_t)-\cos(\phi))/c} \\
= \quad & \frac{1-\left(e^{-j\omega d(\cos(\phi_t)-\cos(\phi))/c}\right)^N}{1-e^{-j\omega d(\cos(\phi_t)-\cos(\phi))/c}} \\
= \quad & \frac{1-e^{-jN\omega d(\cos(\phi_t)-\cos(\phi))/c}}{1-e^{-j\omega d(\cos(\phi_t)-\cos(\phi))/c}} \\
= \quad & \frac{e^{-jN\omega d(\cos(\phi_t)-\cos(\phi))/2c}}{e^{-j\omega d(\cos(\phi_t)-\cos(\phi))/2c}} \cdot \frac{e^{jN\omega d(\cos(\phi_t)-\cos(\phi))/2c}-e^{-jN\omega d(\cos(\phi_t)-\cos(\phi))/2c}}{e^{j\omega d(\cos(\phi_t)-\cos(\phi))/2c}-e^{-j\omega d(\cos(\phi_t)-\cos(\phi))/2c}} \\
= \quad & e^{-j(N-1)\omega d(\cos(\phi_t)-\cos(\phi))/2c} \cdot \frac{e^{jN\omega d(\cos(\phi_t)-\cos(\phi))/2c}-e^{-jN\omega d(\cos(\phi_t)-\cos(\phi))/2c}}{e^{j\omega d(\cos(\phi_t)-\cos(\phi))/2c}-e^{-j\omega d(\cos(\phi_t)-\cos(\phi))/2c}} \\
\text{(A.3)} \quad = \quad & e^{-j(N-1)\omega d(\cos(\phi_t)-\cos(\phi))/2c} \cdot \frac{2j\sin(N\omega d(\cos(\phi_t)-\cos(\phi))/2c)}{2j\sin(\omega d(\cos(\phi_t)-\cos(\phi))/2c)} \\
= \quad & e^{-j(N-1)\omega d(\cos(\phi_t)-\cos(\phi))/2c} \cdot \frac{\sin(N\omega d(\cos(\phi_t)-\cos(\phi))/2c)}{\sin(\omega d(\cos(\phi_t)-\cos(\phi))/2c)} \\
= \quad & e^{-j(N-1)\alpha} \cdot \frac{\sin(N\alpha)}{\sin(\alpha)}\,.
\end{aligned}
$$

Appendix B

MMSE Beamformer

With our noise and signal model as given by Eq. (3.44) and the plane wave assumption (Eq. (3.47)), we can determine the MMSE beamformer's weight vector as given by Eq. (3.79) further. The first part of the weight vector becomes

$$\begin{aligned} \boldsymbol{\sigma^2}_{XD}^H &= E\{\boldsymbol{X}^H D\} \\ &= E\{(\boldsymbol{d}(\omega, \phi_t) S(\omega) + \boldsymbol{X_N}(\omega))^H D\} \\ &= E\{S(\omega)^* \boldsymbol{d}(\omega, \phi_t)^H D\} + E\{\boldsymbol{X_N}(\omega)^H D\} \,. \end{aligned} \tag{B.1}$$

If the desired signal is the source signal, i.e. $D(\omega) = S(\omega)$, assuming uncorrelated noise and source signals, we have

$$\begin{aligned} \boldsymbol{\sigma^2}_{XD}^H &= E\{S(\omega)^* \boldsymbol{d}(\omega, \phi_t)^H S(\omega)\} + E\{\boldsymbol{X_N}(\omega)^H S(\omega)\} \\ &= E\{S(\omega) S(\omega)^*\} \boldsymbol{d}(\omega, \phi_t)^H \\ &= \sigma_S^2 \boldsymbol{d}(\omega, \phi_t)^H \,. \end{aligned} \tag{B.2}$$

The second part of the weight vector Eq. (3.79) can also be specified further. The PSD matrix of the subband microphone signals under our assumptions becomes

$$\begin{aligned} \boldsymbol{P}_X &= E\{\boldsymbol{X}\boldsymbol{X}^H\} \\ &= E\{(\boldsymbol{d}(\omega, \phi_t) S + \boldsymbol{X_N})(S^* \boldsymbol{d}(\omega, \phi_t)^H + \boldsymbol{X_N}^H)\} \\ &= E\{\boldsymbol{d}(\omega, \phi_t) S S^* \boldsymbol{d}(\omega, \phi_t)^H + \boldsymbol{d}(\omega, \phi_t) S \boldsymbol{X_N}^H \qquad \text{(B.3)} \\ &\quad + \boldsymbol{X_N} S^* \boldsymbol{d}(\omega, \phi_t)^H + \boldsymbol{X_N}\boldsymbol{X_N}^H\} \\ &= E\{S S^*\} \, \boldsymbol{d}(\omega, \phi_t) \, \boldsymbol{d}(\omega, \phi_t)^H + E\{\boldsymbol{X_N}\boldsymbol{X_N}^H\} \\ &= \sigma_S^2 \, \boldsymbol{d}(\omega, \phi_t) \, \boldsymbol{d}(\omega, \phi_t)^H + \boldsymbol{P}_{X_N} \,. \end{aligned} \tag{B.4}$$

Using the well-known matrix inversion lemma, the inverse of the PSD matrix can be derived [Van Trees, 2002, §6.2.2.1], giving

$$\boldsymbol{P}_X^{-1} = \boldsymbol{P}_{X_N}^{-1} - \boldsymbol{P}_{X_N}^{-1} \, \sigma_S^2 \, \boldsymbol{d} \, [1 + \boldsymbol{d}^H \boldsymbol{P}_{X_N}^{-1} \, \sigma_S^2 \, \boldsymbol{d}]^- 1 \, \boldsymbol{d}^H \boldsymbol{P}_{X_N}^{-1} \,. \tag{B.5}$$

Finally, substituting our specialised equations (B.2) and (B.5) into (3.79), we obtain the optimal MMSE weight vector for the plane wave assumption with uncorrelated, additive noise:

$$\begin{aligned}\boldsymbol{W}^H_{\text{MMSE}} &= \sigma_S^2\,\boldsymbol{d}^H\,\boldsymbol{P}^{-1}_{X_N} - \boldsymbol{P}^{-1}_{X_N}\,\sigma_S^2\,\boldsymbol{d}\,[1+\boldsymbol{d}^H\boldsymbol{P}^{-1}_{X_N}\,\sigma_S^2\,\boldsymbol{d}]^{-1}\,\boldsymbol{d}^H\boldsymbol{P}^{-1}_{X_N} \\ &= \frac{\sigma_S^2}{\sigma_S^2+\Lambda(\omega)}\Lambda(\omega)\cdot\boldsymbol{d}(\omega,\phi_t)\boldsymbol{P}^{-1}_{X_N}\,, \qquad \text{(B.6)}\end{aligned}$$

where $\Lambda(\omega)$ is defined as in Eq. (3.81).

Appendix C

Generalised Gaussian PDF Derivations

C.1 The rth Moment of the Real-Valued GG pdf

In this section we derive an expression for the r-th moment of the real-valued generalised Gaussian pdf and show that the scaling parameter $\hat{\sigma}$ is the square root of the variance of the GG pdf.

The rth moment of the GG pdf can be expressed as

$$\mathcal{E}\left\{y^r\right\} = \frac{1}{2\Gamma(1+1/f)C_1\,\hat{\sigma}} \int_{-\infty}^{\infty} y^r \exp\left[-\left(\frac{y}{C_1\,\hat{\sigma}}\right)^f\right] dy, \quad \text{for integer } r > 0\,, \tag{C.1}$$

with C_1 as defined in Eq. 4.3. Since the GG pdf is an even function about the mean, we can rewrite (C.1) as

$$\mathcal{E}\left\{y^r\right\} = \frac{1}{\Gamma(1+1/f)C_1\,\hat{\sigma}} \int_{0}^{\infty} y^r \exp\left[-\frac{y^f}{C_1^f\,\hat{\sigma}^f}\right] dy. \tag{C.2}$$

Defining

$$v = \frac{y^f}{C_1^f\,\hat{\sigma}^f}$$

implies

$$\frac{dv}{dy} = \frac{f y^{f-1}}{C_1^f\,\hat{\sigma}^f}\,,$$

so that Equation (C.2) can be solved as

$$\begin{aligned}\mathcal{E}\left\{y^r\right\} &= \frac{C_1^r\,\hat{\sigma}^r}{f\,\Gamma(1+1/f)} \int_0^{\infty} v^{\frac{r+1}{f}-1}\,e^{-v}\,dv \\ &= \frac{C_1^r\,\hat{\sigma}^r}{f\,\Gamma(1+1/f)}\,\Gamma\left(\frac{r+1}{f}\right).\end{aligned} \tag{C.3}$$

Now the gamma function satisfies the following functional equation:

$$\Gamma(1+z) = z\Gamma(z), \quad \text{for complex z except 0, -1, -2, -3, ...,} \tag{C.4}$$

and therefore

$$f\,\Gamma(1+1/f) = \Gamma(1/f)\,. \tag{C.5}$$

Equation (C.3) can then be simplified as follows:

$$\mathcal{E}\left\{y^r\right\} = \frac{C_1^r\,\hat{\sigma}^r}{\Gamma(1/f)}\,\Gamma\left(\frac{r+1}{f}\right)\,. \tag{C.6}$$

The 2nd order moment (r=2) is then:

$$\mathcal{E}\left\{y^2\right\} = \frac{C_1^2\,\hat{\sigma}^2}{\Gamma(1/f)}\,\Gamma\left(3/f\right) = \frac{\left[\frac{\hat{\sigma}^2\Gamma(1/f)}{\Gamma(3/f)}\right]}{\Gamma(1/f)}\,\Gamma\left(3/f\right) = \hat{\sigma}^2\,. \tag{C.7}$$

This demonstrates that the GG pdf scaling parameter $\hat{\sigma}$ is the square root of the variance.

C.2 Generalisation of the GG pdf to Complex Random Variables

This section provides a derivation of the generalised Gaussian pdf for complex random variables, Equation (6.1).

Assuming that $z = \rho\, e^{\phi}$ is generated by a circular complex random process implies that the pdf of $\mathbf{z}$ then assumes the functional form

$$p(z) = p(\rho,\phi) = \frac{1}{D_2}\exp - \left(\frac{\rho}{C_2\hat{\sigma}}\right)^f$$

where f is the *shape parameter*, $\hat{\sigma}$ is the *scale parameter*, and D_2 is the normalisation constant required to ensure that $p(\mathbf{x})$ is a valid pdf. We write $C_2 \cdot \hat{\sigma}$ instead of just $\hat{\sigma}$ in the denominator of the exponent so that we can ensure that the scale parameter equals the variance, as is the case for the real-valued GG pdf.

We begin with the calculation of D_2. In polar coordinates a differential element of area ΔA can be expressed as

$$\Delta A = \rho\, d\rho\, d\phi.$$

Hence, the normalisation constant must satisfy

$$D_2 = \int_{-\pi}^{\pi}\int_0^{\infty} \rho \exp - \left(\frac{\rho}{C_2\hat{\sigma}}\right)^f\, d\rho\, d\phi \tag{C.8}$$

$$= 2\pi\int_0^{\infty} \rho \exp - \left(\frac{\rho}{C_2\hat{\sigma}}\right)^f\, d\rho\,. \tag{C.9}$$

Under the change of variables

$$v = \frac{\rho}{\hat{\sigma}\, C_2}, \tag{C.10}$$

Eq. (C.9) can be rewritten as

$$D_2 = 2\pi C_2^2 \hat{\sigma}^2 \int_0^\infty v \exp -v^f \, dv\,. \tag{C.11}$$

Next we calculate the variance of $z = \rho e^{j\phi}$, which is by definition

$$\begin{aligned}\sigma_z^2 = E\{|z|^2\} &= \int_{-\pi}^{\pi} \int_0^\infty \rho\, e^{j\phi} \cdot \rho\, e^{-j\phi} \cdot p(\rho, \phi) \cdot \rho\, d\rho\, d\phi \\ &= \frac{2\pi}{D_2} \int_0^\infty \rho^3 \exp - \left(\frac{\rho}{C_2 \hat{\sigma}}\right)^f d\rho\,.\end{aligned}$$

Once more introducing the change of variables Eq. (C.10) provides

$$\sigma_z^2 = \frac{2\pi \hat{\sigma}^4\, C_2^4(z)}{D_2} \int_0^\infty v^3 \exp -v^f \, dv\,. \tag{C.12}$$

The calculation of both c and σ_z^2 involve integrals of the form

$$\begin{aligned}I_n(f) &= \int_0^\infty v^n \, \exp -v^f \, dv \\ &= \frac{1}{f} \cdot \Gamma\left(\frac{n+1}{f}\right) \; \forall n = 1, 3.\end{aligned} \tag{C.13}$$

Substituting (C.13) (with n=1) into (C.11) provides

$$D_2 = 2\pi C_2^2 \hat{\sigma}^2 \frac{1}{f} \cdot \Gamma\left(2/f\right) \tag{C.14}$$

$$= \pi\, \Gamma(1 + 2/f)\, {C_2}^2\, \hat{\sigma}^2\,. \tag{C.15}$$

Then substituting (C.13) (with n=3) and (C.15) into (C.12), we arrive at

$$\sigma_z^2 = \frac{2\pi A_c^4 \hat{\sigma}^4 \cdot \frac{1}{f}\Gamma(4/f)}{2\pi C_2^2 \hat{\sigma}^2 \frac{1}{f}\Gamma(2/f)} = C_2^2 \hat{\sigma}^2 \cdot \frac{\Gamma\left(4/f\right)}{\Gamma\left(2/f\right)}. \tag{C.16}$$

Since the factor C_2 is intended to ensure that $\hat{\sigma}$ equals σ_z, we can now determine C_2 by setting $\hat{\sigma} = \sigma_z$, leading to:

$$C_2 = \left[\frac{\Gamma\left(2/f\right)}{\Gamma\left(4/f\right)}\right]^{1/2} \tag{C.17}$$

The generalised Gaussian pdf of a circular complex random variable z, with shape parameter f and scale parameter $\hat{\sigma}$ equal to the variance of z is therefore

$$p_{\mathrm{gg,c}}(Z) = \frac{1}{\pi\, \Gamma(1 + 2/f)\, {C_2}^2\, \hat{\sigma}^2} \exp\left[-\left(\frac{|Z|}{C_2\, \hat{\sigma}}\right)^f\right]\,. \tag{C.18}$$

Note that for the Gaussian case, $f = 2$ and the pdf reduces to

$$p_{\text{Gauss}}(Z) = \frac{1}{\pi\hat{\sigma}^2} \exp\left\{ - \left| \frac{Z}{\hat{\sigma}} \right|^2 \right\}, \tag{C.19}$$

which is the correct form for complex data [Neeser and Massey, 1993]. Similarly, for Laplacian random variables $f = 1$, and the pdf can be expressed as

$$p_{\text{Laplace}}(Z) = \frac{3}{\pi\hat{\sigma}^2} \exp\left\{ - \left| \frac{\sqrt{6}\, Z}{\hat{\sigma}} \right| \right\}. \tag{C.20}$$

Based on Eq. (C.18), the log-likelihood of the GG pdf for a complex r.v. can be expressed as

$$\log p(Z; f, \hat{\sigma}) = -\log \left\{ 2\pi \, \frac{1}{f} \, \Gamma(2/f) C_2^2 \, \hat{\sigma}^2 \right\} - \frac{|Z|^f}{C_2^f \, \hat{\sigma}^f}. \tag{C.21}$$

Then the derivative of $\log p(Z; f, \hat{\sigma})$ with respect to $\hat{\sigma}$ is given by

$$\frac{\partial \log p(Z; f, \hat{\sigma})}{\partial \hat{\sigma}} = \frac{f \, |Z|^f}{C_2^f \, \hat{\sigma}^{f+1}} - \frac{2}{\hat{\sigma}}. \tag{C.22}$$

Appendix D

Loglikelihood and ML Parameter Estimates

D.1 Derivation for PDF Model 1

For a set of training samples $\mathcal{Y} = \{Y_k\}_{k=0}^{K-1}$, the loglikelihood function under the real-valued GG pdf model 1 (4.2) can be expressed as:

$$\begin{aligned}
\log p_{\mathrm{GG}}(\mathcal{Y}, \hat{\sigma}, f) &= \log \prod_{k=0}^{K-1} p_{\mathrm{GG}}(Y_k) \\
&= \sum_{k=0}^{K-1} \log \left[\frac{1}{2\Gamma(1+1/f)\, C_1\, \hat{\sigma}} \exp\left(- \left| \frac{Y_k}{C_1\, \hat{\sigma}} \right|^f \right) \right] \\
&= \sum_{k=0}^{K-1} \left[- \log(2\Gamma(1+1/f)\, C_1\, \hat{\sigma}) - \left| \frac{Y_k}{C_1\, \hat{\sigma}} \right|^f \right] \\
&= -K \log\left(2\Gamma(1+1/f) C_1\, \hat{\sigma}\right) - \frac{\sum_{k=0}^{K-1} |Y_k|^f}{C_1^f\, \hat{\sigma}^f}\,, \qquad \text{(D.1)}
\end{aligned}$$

with

$$C_1 = \left[\frac{\Gamma(1/f)}{\Gamma(3/f)} \right]^{1/2} .$$

To find the MLE parameter estimate of $\hat{\sigma}$ or f which maximises the loglikelihood (D.1), the partial derivative of $\log p_{\mathrm{GG}}(\mathcal{Y}, \hat{\sigma}, f)$ with respect to the parameter is set to zero. For $\hat{\sigma}$, this is

$$\frac{\partial \log p_{\mathrm{GG}}(\mathcal{Y}; \hat{\sigma}, f)}{\partial \hat{\sigma}} = -\frac{K}{\hat{\sigma}} + \frac{f}{\hat{\sigma}^{f+1}} \left[\frac{\Gamma(1/f)}{\Gamma(3/f)} \right]^{-\frac{f}{2}} \sum_{k=0}^{K-1} |Y_k|^f . \qquad \text{(D.2)}$$

By setting Eq. (D.2) to zero and solving for $\hat{\sigma}$, we obtain

$$\hat{\sigma}_{\mathrm{MLE}} = \frac{1}{C_1}\left[\frac{f}{K}\sum_{k=0}^{K-1}|Y_k|^f\right]^{1/f} . \tag{D.3}$$

For f, we have

$$\frac{\partial l(\mathcal{Y};\hat{\sigma},f)}{\partial f} = Ka(f) - \sum_{k=0}^{K-1}\left(\left(\frac{|Y_k|}{C_1\,\hat{\sigma}}\right)^f \cdot \left[\log\left\{\frac{|y_n|}{C_1\,\hat{\sigma}}\right\} + b(f)\right]\right) \tag{D.4}$$

where

$$\begin{aligned} a(f) &= (f^{-2}/2)[2\Psi(1+1/f) + \Psi(1/f) - 3\Psi(3/f)], \\ b(f) &= (f^{-1}/2)[\Psi(1/f) - 3\Psi(3/f)], \end{aligned}$$

and $\Psi(.)$ is the digamma function. Unfortunately, it is not possible to obtain the ML estimate of f by setting (D.5) equal to zero and solving for f, due to the presence of the special function Ψ [Kumatani et al., 2009]. The equation has a unique solution though, and Varanasi and Aazhang [1989] describe a method to obtain it with gradient descent techniques.

D.2 Derivation for PDF model 2 and Generic Form

The derivation for model 2 is analogous to the derivation for model 1. For a set of training samples $\mathcal{Y} = \{Y_k\}_{k=0}^{K-1}$, the loglikelihood function under the GG pdf model 2 (6.1) can be expressed as:

$$\begin{aligned} \log p_{\mathrm{GG}}(\mathcal{Y},\hat{\sigma},f) &= \log\prod_{k=0}^{K-1} p_{\mathrm{GG}}(Y_k) \\ &= \sum_{k=0}^{K-1}\log\left[\frac{f}{2\pi\,\hat{\sigma}^2\,C_2^2\,\Gamma(2/f)}\,\exp\left(-\left|\frac{Y_k}{\hat{\sigma}\,C_2}\right|^f\right)\right] \\ &= K\log\left[\frac{f}{2\pi\,\hat{\sigma}^2\,C_2^2\,\Gamma(2/f)}\right] - \frac{\sum_{k=0}^{K-1}|Y_k|^f}{C_2^f\,\hat{\sigma}^f} . \qquad (D.5) \end{aligned}$$

with

$$C_2 = \left[\frac{\Gamma(2/f)}{\Gamma(4/f)}\right]^{1/2} .$$

The partial derivative of $\log p_{\mathrm{GG}}(\mathcal{Y},\hat{\sigma},f)$ with respect to $\hat{\sigma}$ is

$$\frac{\partial \log p_{\mathrm{GG}}(\mathcal{Y};\hat{\sigma},f)}{\partial\hat{\sigma}} = -\frac{2K}{\hat{\sigma}} + \frac{f}{C_2^f\,\hat{\sigma}^{f+1}}\sum_{k=0}^{K-1}|Y_k|^f . \tag{D.6}$$

By setting (D.6) to zero and solving for $\hat{\sigma}$, we obtain the MLE parameter estimate

$$\hat{\sigma}_{\text{MLE}} = \frac{1}{C_2}\left[\frac{f}{2K}\sum_{k=0}^{K-1}|Y_k|^f\right]^{1/f} . \tag{D.7}$$

Comparing Equations Eq. (D.7) and Eq. (D.3), it is obvious that the ML estimate can be expressed in the common format

$$\hat{\sigma}_{\text{ML}} = \frac{1}{C}\left(\frac{f}{EK}\sum_{k=0}^{K-1}|Y[k]|^f\right)^{1/f} , \tag{D.8}$$

where for model 1, $C = C_1$ and $E = 1$, and for model 2, $C = C_2$ and $E = 2$.

Appendix E

Negentropy Derivation for PDF Model 2

Based on Eq. (6.2), we can calculate the differential entropy of the complete random variable Y according to

$$H_d(Y) = -\int p_{\text{GG}}(y)\log p_{\text{GG}}(y)dy.$$

Substituting $y = \rho e^{j\theta}$, we find

$$\begin{aligned} H_d(Y_{\text{GG}}) = \int_0^\infty \int_{-\pi}^{\pi} K_1(f,\hat{\sigma})\exp\left\{-\left|\frac{\rho e^{j\theta}}{\hat{\sigma}C_2}\right|^f\right\}\cdot \\ \left[-\log K_1(f,\hat{\sigma}) + \left|\frac{\rho e^{j\theta}}{\hat{\sigma}C_2}\right|^f\right] d\theta\,\rho\,d\rho \end{aligned} \tag{E.1}$$

where we have defined

$$K_1(f,\hat{\sigma}) \triangleq \frac{f}{2\pi\hat{\sigma}^2 C_2^2 \Gamma(2/f)}.$$

Upon defining

$$r \triangleq \frac{\rho}{\hat{\sigma}C_2},$$

Eq. (E.1) can be rewritten as

$$\begin{aligned} H_d(Y_{\text{GG}}) &= -\log K_1(f,\hat{\sigma}) + 2\pi K_1(f,\hat{\sigma})\int_0^\infty r^f e^{-r^f}\sigma C_2 r\cdot\sigma C_2 dr, \\ &= -\log K_1(f,\hat{\sigma}) + 2\pi\sigma^2 C_2^2 K_1(f,\sigma)\int_0^\infty r^f e^{-r^f}\,r\,dr, \\ &= -\log K_1(f,\hat{\sigma}) + \frac{f}{\Gamma(2/f)}\int_0^\infty r^{f+1}e^{-r^f}\,dr. \end{aligned} \tag{E.2}$$

Specialising Eq. (C.13) by setting $\mu = f + 1$, we obtain the integral in Eq. (E.2), which implies

$$\int_0^\infty r^{f+1} e^{-r^f} dr = \frac{1}{f} \cdot \Gamma\left(\frac{f+2}{f}\right) = \frac{2}{f^2}\Gamma(2/f), \tag{E.3}$$

where the final equation follows from $\Gamma(1 + z) = z\Gamma(z)$. Substituting Eq. (E.3) into Eq. (E.2), we find

$$H_\text{d}(Y_\text{GG}) = -\log K_1(f, \hat{\sigma}) + \frac{2}{f} \tag{E.4}$$

$$= \log[2\pi\hat{\sigma}^2 C_2^2 \Gamma(2/f)/f] + 2/f. \tag{E.5}$$

For the Gaussian case, $f = 2$, such that

$$C_2 = \left[\frac{\Gamma(1)}{\Gamma(2)}\right]^{1/2} = 1,$$

$$K_1(f = 2, \sigma) = \frac{1}{\pi\sigma^2}\,.$$

Substituting these values into Eq. (E.5) then yields

$$H_\text{d}(Y_\text{gauss}) = 1 + \log \pi\sigma^2\,.$$

Appendix F

The Simplex Algorithm

The following pseudo-code describes the operation of the simplex minimisation algorithm as implemented by the GSL version 1.9.

Algorithm 1 Pseudocode for nmsimplex_iterate() function of the GSL simplex algorithm implementation (module multimin):

```
1) Determine the highest, second highest and lowest point $p_\text{high}, p_\text{2ndHigh}, p_\text{low}$ with corresponding values $f_\text{high}, f_\text{2ndHigh}, f_\text{low}$ for the function $f$ to minimise.
2) Reflect (mirror at the middle point of the remaining vertices) the highest point and evaluate the objective function at that point:
$f_\text{refl} = f(p_\text{refl}) = f(\text{mirror}(p_\text{high}))$
if $f_\text{refl} < f_\text{low}$ then
    3a) The new point is better than the best point. Now try expansion (reflect again and scale by 2.0) instead: $f_\text{exp} = f(p_\text{exp}) = f(\text{expand}(p_\text{high}))$
    if $f_\text{exp} < f_\text{low}$ then
        the expanded point $p_\text{exp}$ is better than the lowest point, so replace the highest point by it.
    else
        replace the highest point by $p_\text{refl}$.
    end if
else if $f_\text{refl} > f_\text{2ndHigh}$ then
    3b) $p_\text{refl}$ is worse than both the lowest and second highest point, so check if it is at least better than the highest point (or equally good):
    if $f_\text{refl} <= f_\text{high}$ then
        Replace the highest point by $p_\text{refl}$.
    end if
    Next try a one-dimensional contraction (scaling by 0.5) of the highest point:
    if $f_\text{contr} = f(p_\text{contr}) = f(\text{contract}(p_\text{high}))$ then
        The resulting point is better than the now highest point, so replace the highest point by it.
    else
        The contracted point is not better. Now contract the whole simplex (scale each point by 0.5) except the best point.
    end if
else
    3c) $p_\text{refl}$ is not better than the lowest point, but better than the second highest point (or equally good), so replace the highest point by it.
end if
Update $F[k]$ to reflect the new set of points and return the lowest point and value.
```

Bibliography

Alejandro Acero. *Acoustical and environmental robustness in automatic speech recognition*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 1991.

John Aldrich. R.A. Fisher and the making of maximum likelihood 1912-1922. *Statistical Science*, 12(3):162–176, 1997.

Tasos Anastasakos, John McDonough, Richard Schwartz, and John Makhoul. A compact model for speaker–adaptive training. In *Proceedings of the Intern. Conference on Spoken Language Processing*, volume 2, pages 1137–1140, 1996.

A. Andreou, T. Kamm, and J. Cohen. Experiments in vocal tract normalization. In *Proceedings of the CAIP Workshop: Frontiers in Speech Recognition II*, 1994.

F. Asano, S. Ikeda, M. Ogawa, H. Asoh, and N. Kitawaki. A combined approach of array processing and independent component analysis for blind separation of acoustic signals. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2729–2732, Washington, DC, USA, 2001. IEEE Computer Society.

B.S. Atal. Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. *Journal of the Acoustical Society of America*, 55:1304–1312, 1974.

M. Basseville. Distance measures for signal processing and pattern recognition. *Signal Processing*, 18(4):349–369, December 1989.

L. E. Baum. An inequality and associated maximization technique in statistical estimation for probalistic functions of markov processes. In *Inequalities III: Proceedings of the Third Symposium on Inequalities*, pages 1–8, University of California, Los Angeles, 1972. Academic Press.

Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

F. Beritelli, S. Casale, and G. Ruggeri. Performance evaluation and comparison of ITU–T/ETSI voice activity detectors. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2001.

Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, USA, 1995.

S. F. Boll. Suppression of acoustic noise in speech using spectral subtraction. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, volume 27, pages 113–117, April 1979.

M. Brandstein and D. Ward, editors. *Microphone Arrays*. Springer, Berlin, Heidelberg, New York, 2001.

Helmut Brehm and Walter Stammler. Description and generation of spherically invariant speech–model signals. *Signal Processing*, 12:119–141, 1987.

R.P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall, Englewood Cliffs, NJ, 1973.

H. Buchner, R. Aichner, and W. Kellermann. Blind source separation for convolutive mixtures: A unified treatment. In Y. Huang and J. Benesty, editors, *Audio Signal Processing for Next-Generation Multimedia Communication Systems*, pages 255–293. Kluwer Academic Publishers, Boston/Dordrecht/London, February 2004.

Ingvar Claesson and Sven Nordholm. A spatial filtering approach to robust adaptive beaming. *IEEE Transactions on Antennas and Propagation*, 19:1093–1096, 1992.

Israel Cohen, Sharon Gannot, and Baruch Berdugo. An integrated real-time beamforming and postfiltering system for nonstationary noise environments. *EURASIP Journal on Applied Signal Processing*, (11):1064–1073, October 2003.

J. Cohen. Application of an auditory model to speech recognition. *Journal of the Acoustical Society of America*, 85(6), June 1989.

M. Cooke, P. D. Green, and M. D. Crawford. Handling missing data in speech recognition. In *Proceedings of the Intern. Conference on Spoken Language Processing*, pages 1555–1558, Yokohama, Japan, 1994.

Jan Mark de Haan. *Filter Bank Design for Subband Adaptive Filtering*. PhD thesis, Karlskrona. Blekinge Institute of Technology, 2001.

Jan Mark De Haan, Nedelko Grbic, Ingvar Claesson, and Sven Erik Nordholm. Filter bank design for subband adaptive microphone arrays. *IEEE Transactions on Speech and Audio Processing*, 11(1):14–23, Jan. 2003.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

Simon Doclo. *Multi-microphone noise reduction and dereverberation techniques for speech applications*. PhD thesis, Katholieke Universiteit Leuven, Leuven, Belgium, Faculty of Engineering, 2003.

Simon Doclo, Ann Spriet, Jan Wouters, and Marc Moonen. Frequency-domain criterion for the speech distortion weighted multichannel wiener filter for robust noise reduction. *Speech Communication, special issue on Speech Enhancement*, 49:636–656, 2007.

J. Armando Domínguez-Molina, Graciela González-Farías, and Ramón M. Rodríguez-Dagnino. A practical procedure to estimate the shape parameter in the generalized gaussian distribution, January 2008. URL `http://en.wikipedia.org/wiki/Generalized_Gaussian_distribution`.

J. Droppo, A. Acero, and L. Deng. Uncertainty decoding with splice for noise robust speech recognition. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Orlando, Florida, USA, 2002.

N. W. D. Evans, J. S. D. Mason, W. M. Liu, and B. Fauve. An assessment on the fundamental limitations of spectral subtraction. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 145–148, 2006.

Jonathan G. Fiscus, Jerome Ajot, and John S. Garofolo. The rich transcription 2007 meeting recognition evaluation. In *Multimodal Technologies for Perception of Humans: International Evaluation Workshops CLEAR 2007 and RT 2007, Baltimore, MD, USA, May 8-11, 2007, Revised Selected Papers*, pages 373–389, Berlin, Heidelberg, 2008. Springer-Verlag.

R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, New York, second edition, 1987.

Jr. Forney, G.D. The Viterbi algorithm. In *Proceedings of the IEEE*, volume 61, pages 268 – 278, March 1973.

Jeroen Fransen, Dave Pye, Tony Robinson, Phil Woodland, and Steve Young. WSJCAM0 corpus and recording description. Technical report, Cambridge University Engineering Department, 1994.

Otis Lamont Frost. Am algorithm for linearly constrained adaptive array processing. *Proceedings of the IEEE*, 60(8):926–935, August 1972.

S. Furui. Speaker–independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on ASSP*, 34:52–59, 1986.

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

M. J. F. Gales and S. J. Young. Cepstral parameter compensation for hmm recognition in noise. *Speech Communication*, 12(3):231–239, 1993.

Mark Gales and Steve Young. The application of hidden Markov models in speech recognition. *Foundations and Trends in Signal Processing*, 1(3):195–304, 2008. ISSN 1932-8346.

M.J.F. Gales. Semi-tied covariance matrices for hidden markov models. *IEEE Transactions on Speech and Audio Processing*, 7:272–281, 1999.

Robert G. Gallager. *Information Theory and Reliable Communication*. John Wiley & Sons, New York, 1968.

Sharon Gannot and Israel Cohen. Speech enhancement based on the general transfer function GSC and postfiltering. *IEEE Transactions Speech and Audio Processing*, 12:561–571, 2004.

Sharon Gannot, David, Burshtein, and Ehud Weinstein. Signal enhancement using beamforming and nonstationarity with applications to speech. *IEEE Transactions on Signal Processing*, 49:1614–1626, 2001.

John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallett, Nancy L. Dahlgren, and Victor Zue. *TIMIT Acoustic-Phonetic Continuous Speech Corpus*. Linguistic Data Consortium, Philadelphia, 1993.

Tobias Gehrig, U. Klee, John McDonough, S. Ikbal, Matthias Wölfel, and C. Fügen. Tracking and beamforming for multiple simultaneous speakers with probabilistic data association filters. In *Proceedings of Interspeech*, pages 2594–2597, 2006.

L. Gillick and S. Cox. Some statistical issues in the comparison of speech recognition algorithms. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 532–535, 1989.

Nedelko Grbić. *Optimal and Adaptive Subband Beamforming*. PhD thesis, Blekinge Institute of Technology, 2001.

Lloyd J. Griffiths and Charles W. Jim. An alternative approach to linearly constrained adaptive beamforming. *IEEE Transactions Antennas and Propagation*, 30(1):27–34, January 1982.

R. Haeb-Umbach and H. Ney. Linear Discriminant Analysis For Improved Large Vocabulary Continuous Speech Recognition. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 13–16, 1992.

Thomas Hain, Phil Woodland, Thomas Niesler, and Edward Whittacker. The 1998 HTK system for transcription of conversational telephone speech. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 57–60, 1998.

Wolfgang Herbordt and Walter Kellermann. Frequency-domain integration of acoustic echo cancellation and a generalized sidelobe canceller with improved robustness. *European Transactions on Telecommunications (ETT)*, 13:123–132, 2002.

Wolfgang Herbordt and Walter Kellermann. Adaptive beamforming for audio signal acquisition. In J. Benesty and Y. Huang, editors, *Adaptive Signal Processing: Applications to Real-World Problems*, pages 155–194. Springer, Berlin, Germany, 2003.

Wolfgang Herbordt, Herbert Buchner, Satoshi Nakamura, and Walter Kellermann. Multichannel bin-wise robust frequency-domain adaptive filtering and its application to adaptive beamforming. *IEEE Transactions on Audio, Speech and Language Processing*, 15:1340–1351, 2007.

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

H. Hermansky and N. Morgan. RASTA processing of speech. *IEEE Transactions on Speech and Audio Processing*, 2(4):587–589, October 1994.

H. Günter Hirsch. Estimation of noise spectrum and its application to SNR-estimation and speech enhancement. TR-93-012. Technical report, International Computer Science Institute, 1993.

Osamu Hoshuyama, Akihiko Sugiyama, and Akihiro Hirano. A robust adaptive beamformer for microphone arrays with a blocking matrix using constrained adaptive filters. *IEEE Transactions on Signal Processing*, 47:2677–2684, 1999.

Aapo Hyvärinen. Survey on independent component analysis. *Neural Computing Surveys*, 2:94–128, 1999.

Aapo Hyvärinen and Erkki Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13:411–430, 2000.

Frederick Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA, 1997.

Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice Hall, 2nd edition, 2000.

Kenichi Kumatani, Tobias Gehrig, Uwe Mayer, Emilian Stoimenov, John McDonough, and Matthias Wölfel. Adaptive beamforming with a minimum mutual information criterion. *IEEE Transactions on Audio, Speech and Language Processing*, 15:2527–2541, 2007.

Kenichi Kumatani, John McDonough, Stefan Schacht, Dietrich Klakow, Philip N. Garner, and Weifeng Li. Filter bank design based on minimization of individual aliasing terms for minimum mutual information subband adaptive beamforming. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1609–1612, Las Vegas, Nevada, USA, 2008.

Kenichi Kumatani, John McDonough, Barbara Rauch, Dietrich Klakow, Philip N. Garner, and Weifeng Li. Beamforming with a maximum negentropy criterion. *IEEE Transactions on Audio, Speech and Language Processing*, 17:994–1008, July 2009.

H. Liao and M. J. F. Gales. Joint uncertainty decoding for noise robust speech recognition. In *Proceedings of Interspeech*, pages 3129–3132, Lisbon, Portugal, 2005.

M. Lincoln, I. McCowan, I. Vepa, and H. K. Maganti. The multi-channel Wall Street Journal audio visual corpus (MC-WSJ-AV): Specification and initial experiments. In *Proceedings of the IEEE workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 357–362, 2005.

Claude Marro, Yannick Mahieux, and K. Uwe Simmer. Analysis of noise reduction and dereverberation techniques based on microphone arrays with postfiltering. *IEEE Transactions on Speech and Audio Processing*, 6:240–259, 1998.

J.W. McDonough. *Speaker compensation with all–pass transforms*. Doctoral thesis, Johns Hopkins University, Baltimore, Maryland, USA, 2000.

Paul Mermelstein. Distance measures for speech recognition: Psychological and instrumental. In C. H. Chen, editor, *Pattern Recognition and Artificial Intelligence*, pages 374–388. Academic Press, New York, 1976.

P. J. Moreno, B. Raj, and R. M. Stern. A vector Taylor series approach for environment-independent speech recognition. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 733–736, Washington, DC, USA, 1996. IEEE Computer Society.

Fredy D. Neeser and James L. Massey. Proper complex random processes with applications to information theory. *IEEE Transactions on Information Theory*, 39(4):1293–1302, July 1993.

J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, January 1965.

Sven Nordebo, Ingvar Claesson, and Sven Nordholm. Adaptive beamforming: spatial filter designed blocking matrix. *IEEE Journal of Oceanic Engineering*, 19:583–590, 1994.

Sven Nordholm, Ingvar Claesson, and Bengt Bengtsson. Adaptive array noise suppression of handsfree speaker input in cars. *IEEE Transactions on Vehicular Technology*, 42:514–518, 1993.

Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing*. Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

Douglas B. Paul and Janet M. Baker. The design for the wall street journal-based CSR corpus. In *HLT '91: Proceedings of the workshop on Speech and Natural Language*, pages 357–362, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

David Pearce and Hans-Günter Hirsch. The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *in ISCA Workshop on Automatic Speech Recognition: Challenges for the Next Millenium*, pages 29–32, 2000.

Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

Barbara Rauch, Kenichi Kumatani, Friedrich Faubel, John McDonough, and Dietrich Klakow. On Hidden Markov Model Maximum Negentropy Beamforming. In *Proceedings of International Workshop on Acoustic Echo and Noise Control*, Seattle, WA, USA, September 2008.

John A. Rice. *Mathematical Statistics and Data Analysis*. Duxbury Press, April 2001.

Hiroshi Saruwatari, Toshiya Kawamura, Tsuyoki Nishikawa, Akinobu Lee, and Kiyohiro Shikano. Blind source separation based on a fast-convergence algorithm combining ica and beamforming. *IEEE Transactions on Audio, Speech and Language Processing*, 14(2):666–678, 2006.

Michael Seltzer. *Microphone Array Processing for Robust Speech Recognition*. PhD thesis, Carnegie Mellon University, 2003.

K.U. Simmer and C. Marro J. Bitzer. Post-filtering techniques. In M. Brandstein and D. Ward, editors, *Microphone Arrays*, pages 39–57. Springer, Berlin, Heidelberg, New York, May 2001.

P. P. Vaidyanathan. *Multirate Systems and Filter Banks*. Prentice Hall, Englewood Cliffs, 1993.

D. van Compernolle. Noise adaptation in hidden markov model speech recognition systems. *Computer Speech and Language*, 3(2):151–168, 1989.

H. L. Van Trees. *Optimum Array Processing*. Wiley-Interscience, New York, 2002.

Barry D. Van Veen and Kevin M. Buckley. Beamforming: A versatile approach to spatial filtering. *IEEE Signal Processing Magazine*, 5(2):4–24, April 1988.

Mahesh K. Varanasi and Behnaam Aazhang. Parametric generalized gaussian density estimation. *Journal of the Acoustical Society of America*, 86:1404–1415, 1989.

A. P. Varga and R. K. Moore. Hidden Markov model decomposition of speech and noise. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 845–848, 1990.

Xue Wang, Louis C. W. Pols, and Louis F. M. Ten Bosch. Analysis of context-dependent segmental duration for automatic speech recognition. In *Proceedings of the Intern. Conference on Spoken Language Processing*, pages 3–6, 1996.

Ernst Warsitz and Reinhold Haeb-Umbach. Blind acoustic beamforming based on generalized eigenvalue decomposition. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(5):1529–1539, July 2007.

Ernst Warsitz, Alexander Krueger, and Reinhold Haeb-Umbach. Speech enhancement with a new generalized eigenvector blocking matrix for application in a generalized sidelobe canceller. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, USA, April 2008.

M. Wölfel and J.W. McDonough. Minimum variance distortionless response spectral estimation, review and refinements. *IEEE Signal Processing Magazine*, 22 (5):117–126, Sept. 2005.

Matthias Wölfel and John McDonough. *Distant Speech Recognition*. John Wiley & Sons, New York, 2009.

Matthias Wölfel, John Mcdonough, and Alex Waibel. Minimum variance distortionless response spectral estimation, review and refinements. In *IEEE Signal Processing Magazine*, pages 117–126, 2005.

C. Yang, F.K. Soong, and T. Lee. Static and dynamic spectral features: their noise robustness and optimal weights for ASR. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005.

R. Zelinski. A microphone array with adaptive post-filtering for noise reduction in reverberant rooms. In *Proceedings of the Intern. Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, New York, NY, USA, April 1988.