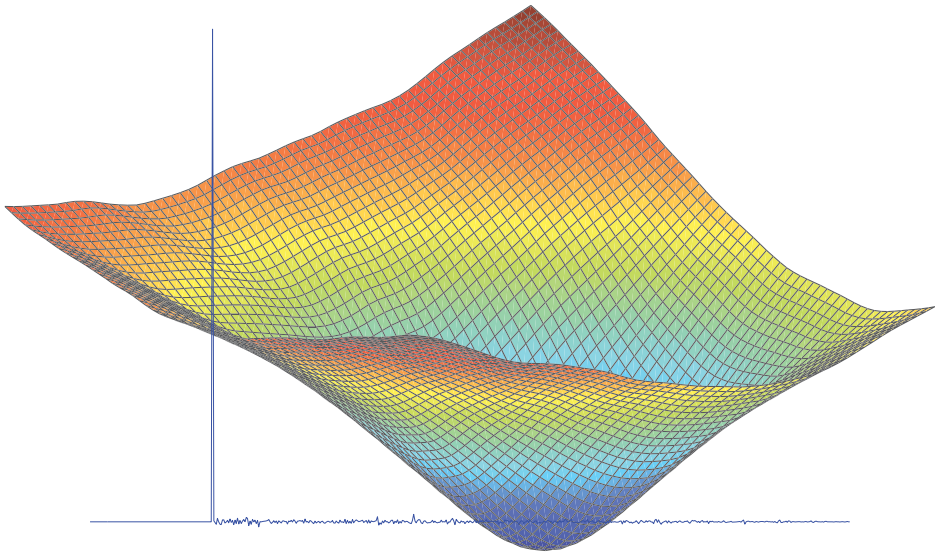
Information and Kurtosis Approaches Applied to Adaptive Blind Speech Dereverberation



Group 1040, 10th semester

Prepared from the 2nd of February to the 3rd of June 2009

Applied Signal Processing and Implementation
Department for Electronic Systems
Aalborg University

AALBORG UNIVERSITET

Aalborg Universitet
Elektronik og Elektroteknik
Fredrik Bajers Vej 7
DK-9200 Aalborg
Tlf. (+45) 9940 8600
http://www.es.aau.dk

Title:
Information and Kurtosis Approaches Applied to Adaptive Blind Speech Dereverberation

Theme:
Master thesis

Project period:
E10, Spring term 2009

Project group:
09gr1040

Participants:
Mads Lauridsen
Niels Lovmand Pedersen

Supervisor:
Kjeld Hermansen

Copies: 4

Number of pages: 140

Attachment: CD

Printed: June 3, 2009

Abstract:

Reverberation in speech, caused by room reflections, is problematic especially for hearing impaired people, and therefore blind speech dereverberation is an important research area. The task is to remove reverberation from the output of a room, where the room as well as the speech input are unknown. In this project different approaches, based on higher-order statistics (HOS) and information theory, to the blind dereverberation problem, are analysed. The speech is assumed to consist of independent and identically distributed (iid) non-Gaussian samples, whereas the room output is assumed to consist of non iid Gaussian distributed samples.

The method based on HOS maximizes the fourth order cumulant, referred to as kurtosis, because it is known that all Gaussian distributed signals have higher-order cumulants equal to zero, hence the output is then made as non-Gaussian as possible. For the information approach two methods are analysed: maximum and minimum entropy. The maximum entropy method maximizes the entropy through a nonlinear transformation, such that statistical dependency between samples is minimized. The entropy of the room output is minimized by the latter method, because Gaussian distributed signals lead to maximum entropy.

The functionality of the three methods is analysed, tested, and compared on basis of the reduction of early and late reflections, convergence ability, and a real life application. Furthermore the complexity with respect to the computations needed by each method is calculated. Based on tests the project group presents adjustments to the methods which improve the performance on all fields for all methods. Furthermore the project group presents a novel algorithm based on the information theory methods for future work.

The conclusion is that all three methods are able to significantly reduce the reverberation. The methods based on information theory seem to be superior to the method based on HOS.

Mads Lauridsen Niels Lovmand Pedersen

Preface

This report is written during the 10th semester specialisation in Applied Signal Processing and Implementation at the department of Electronic Systems, Aalborg University.
According to the study guidelines the student must be able to [1, p. 54]:

- Account for the relevance of the chosen problem to the specialisation including specifically account for the core of the problem and the technical connections in which it appears
- Account for possible methods to solve the problem statements of the project, describe and assess the applicability of the chosen method including account for the chosen delimitation and the way these will influence on the results of the product
- Analyse and describe the chosen problem applying relevant theories, methods and experimental data
- Describe the relevant theories and methods in a way that highlights the special characteristics and hereby document knowledge of the applied theories, methods, possibilities and delimitations within the relevant problem area
- Analyse and assess experimental data, including the effect the assessment method has on the validity of the results.
- Form synthesis between the technical problem and the theoretical and experimental results and perform a critical assessment of the formed synthesis and the outcome of the project.

This report will cover analysis, implementation and test of algorithms used to deconvolve room impulse responses. The implementation of the algorithms is made in Matlab 2008b. The objective of this report is to compare the algorithms, which are based on higher-order statistics and information theory, with regards to their ability to deconvolve, convergence speed, and the number of required multiplications.
In the report the natural logarithm is denoted with log. If both time and vector indexes are used the variable $\mathbf{h}_j(k)$ is the j'th coefficient at time index k, where the vector is denoted with bold letters. Literature references are represented with numbers, e.g. [number], and the full list of references is found in the Bibliography section.

The enclosed CD contains the following: the report in PDF format `09gr1040.pdf`, the source code for the algorithms in Matlab format, and the recordings made in the Section of Acoustics at Aalborg University.
The project group would like to thank electronics engineer Claus Vestergaard Skipper who is the technician in the Section of Acoustics for his help with the measurements of room impulse responses.

Contents

1 Introduction **1**

1.1 Previous Work 2

1.2 Problem Statement 3

2 Project Definitions **5**

2.1 Room 5

2.2 Speech Signal 7

2.3 Performance Metrics 8

3 Inversion of a Non-Minimum Phase Filter **11**

3.1 Inversion using the FFT 15

4 Deconvolution using Statistical Techniques **19**

4.1 Higher-Order Statistics 20

4.2 Information theory 20

4.3 Higher-Order Statistics Theory 21

5 Kurtosis Approach **27**

5.1 Unconstrained Maximization 28

5.2 Constrained Maximization 41

5.3 Summary 42

6 Information Approach **43**

6.1 Maximum Entropy 44

6.2 Summary 58

6.3 Minimum Entropy 59

6.4 Summary 74

7 Tests and Comparison of the Approaches **77**

7.1 Impulse-to-Noise Ratio 79

7.2 Convergence speed 80

7.3 Reduction of Late Reverberation 81

7.4 Tests on measured RIR 83

7.5 Summary 91

8 Complexity of the Algorithms **93**

8.1 Multiplication Count for Kurtosis Maximization 94

8.2 Multiplication Count for Maximum Entropy 95

8.3 Multiplication Count for Minimum Entropy 96

8.4 Comparison with a Digital Signal Processor 97

9 Conclusion **101**

10 Future Work **105**

Bibliography **107**

A Autocovariance Calculations **111**

B Accept/Reject Sampling **113**

C Nonparametric Density Estimation **115**

D Pdf Estimation Test **119**

E Matlab Code for the Algorithms **121**

F Measurement of a RIR **125**

1 Introduction

When a person is speaking in a regular room the listeners will not only perceive the direct signal, but also multipath signals created by reflections on the room walls. The multipath signals are delayed and possibly attenuated as compared to the direct path signal. This phenomenon is known as reverberation. A single delayed signal, which is attenuated and part of the feedback existing in the room, is called an echo, but when multiple signals are delayed, attenuated, and part of the feedback it is called reverberation.
The reverberation effect can be divided into two separate effects: early and late reflections. Each of the two types degrades the speech intelligibility and the overall quality of the signal. The scope of this project is to reduce the reverberation effects on speech. According to [2] the problem is most severe for hearing-impaired people, and the suggested application for this project is therefore a hearing aid. Furthermore the project will focus on single microphone systems since this provides a more challenging problem, and because a single microphone setup is more practical.

The room's impact on the speech is modeled as a convolution in time of the speech signal and the room impulse response (RIR) [3]. Furthermore the system can also be modeled as containing additive noise [4]. Because of the degradation of the speech intelligibility it is of interest to minimize the effect of the room. One possibility is to change the layout of the room or the reflecting material on the walls. This method is however not always applicable. The second possibility, which is investigated in this report, is to use signal processing on the received reverberant speech signal.
Since the model is a convolution in time the inverse RIR can be applied to the signal. The idea is that if the designed inverse RIR is determined and applied, ideally only the delayed version of the direct signal will remain after the signal processing. There is several problems associated with this method. The RIR is in general assumed to be non-minimum phase. A non-minimum phase filter is a mixture of a minimum phase filter, where all zeros lie inside the unit circle, and a maximum phase filter, where all the zeros lies outside of the unit circle. The inversion of the a non-minimum phase filter will therefore result in a filter, which has poles outside the unit circle, which is unstable or non-causal. [5]

Blind dereverberation is addressed in this project where neither the input nor the room is known. This is the case in many applications, e.g. a speaker in a room. Many articles have been published within this area and in the next section some of the interesting methods are presented.

1.1 Previous Work

From the introduction of the reverberation problem it emerges that the RIR must in general be assumed to be a non-minimum phase impulse response. Two widespread approaches for obtaining the inverse RIR are based on either the direct estimation of the inverse RIR or the estimation of the RIR followed by an inversion of this. The former approach has the advantage that no RIR has to be estimated. The scope of this project is on the direct estimation of the inverse RIR, and this brief survey of previous work will focus on this approach. The previous work described is categorized further into two main categories: higher-order statistics (HOS) and information theory.
The key assumption for both methods is that the reverberant speech, which is delayed and weighted versions of the assumed independent and identically distributed (iid) speech samples, approximates a Gaussian distribution due to the central limit theorem.

Due to the assumption that the RIR is non-minimum phase, 2nd order statistics, that is the autocovariance of the signal and the following power spectrum, cannot be used to reconstruct the direct signal. Higher-order statistics is therefore needed, where it is known that for a Gaussian distributed signal these are all zero.
Gillespie et al. propose in [6] an algorithm that maximizes the fourth order statistic, referred to as kurtosis, of the LP residuals which entails a non-Gaussian distribution of the residuals. The algorithms using LP residuals are based on the observation that the residuals contain the impulses from reverberant speech, that is the impulses from clean speech and the impulses caused by room reflections. Furthermore it is assumed that the LP coefficients obtained from clean speech are equivalent to the LP coefficients for reverberant speech. This implies that an estimate of the clean speech can be obtained by synthesizing the modified LP residuals and thereby obtaining the improved speech [7].
In [8] Tonelli et al. emphasise stability problems in certain unpredictable conditions of the approach by Gillespie et al. A maximum-likelihood (ML) approach is therefore suggested where a probability density function (pdf) with a high kurtosis is maximized. However the method does not significantly improve upon the late reverberations and the authors therefore suggest further research with focus on longer equalization filters. A two-stage algorithm is proposed by Wu et al. in [2] for removing the late reverberation impulses. The first stage consists of the algorithm by [6] and the second stage is a shifted spectral subtraction of the power spectrum of the late impulses and the power spectrum of the filtered speech in the first stage. The comparison of the performance of this two-stage approach for attenuating the late reverberation impulses with the approaches in [6] and [8] is however difficult because different performance metrics are used in the experimental results.

From an information theoretic point of view, the reverberated speech, which consists of delayed and summed samples of itself, can be viewed as redundant samples. The reverberation problem becomes a redundancy reduction problem, where statistical dependency between samples is minimized. Bell et al. proposes in [9] a method for minimizing the mutual information between samples, which corresponds to minimizing the statistical dependency. In the kurtosis

approach, statistics of higher order than four are ignored. It is assumed that if the kurtosis of the direct signal matches the kurtosis of the processed signal, a perfect deconvolution have been obtained. Bell et al. suggest that all HOS of the signal should be considered to obtain minimum statistical dependency between samples. The HOS of the reverberated signal, are accessed through a nonlinear transformation of the signal. The HOS are then weighted by maximizing the entropy on the output of the nonlinear transformation. The transformation bounds the values of the output samples, and it is known that maximizing entropy corresponds to minimizing mutual information for bounded samples.
The main disadvantage of the proposed method by Bell et al. is that the chosen nonlinear transformation should be chosen according to the direct signal, which is not known. This entails that this choice relies heavily on the a priori knowledge of the direct signal. Erdogmus et al. presents in [10] a method that does not rely on this a priori knowledge. Knowing that the reverberated samples are assumed Gaussian distributed and their values are unbounded, then as shown by Shannon in [11] the entropy are maximum. The method by Erdogmus et al. therefore minimizes the entropy of the reverberated signal.

The previous work do not solve the dereverberation problem and it therefore remains a very challenging research area.

1.2 Problem Statement

The scope of this project is to analyse and simulate methods for blind deconvolution of reverberant speech using a single microphone setup. The brief survey of previous work showed that a variety of different methods have been proposed for this matter. This project will take its starting point in the work by Gillespie et al., [6], Bell et al., [9], and Erdogmus et al., [10]. The use of higher-order statistics and information theory for speech dereverberation will be investigated. The proposed methods will be analysed and tested under known conditions and adjustments will be made. In addition to this a real life RIR will be measured and the methods will be tested on this RIR.

The outline of this report is as follows. In chapter 2 the prerequisites for the reverberation problem are described, including the performance metrics for the analysed, implemented, and tested methods. If the non-minimum phase filter was known it is possible to invert it, and this is shown in chapter 3. As already stated the objective of this project is the direct estimation of the inverse RIR for speech dereverberation. The principles presented in chapter 3 are therefore not used in the rest of the report and they are only presented to provide problem insight. The basic theory of higher-order statistics, hereunder that they are a necessary requirement to determine the phase-response of a non-minimum phase filter, is described in chapter 4. The chapter also includes basic information theory. In chapter 5 and 6 the HOS and information approaches are analysed, implemented, and tested, respectively. The methods are compared according to the chosen performance metrics in chapter 7 and 8. Finally, conclusions are made in chapter 9 and ideas for future work is presented in chapter 10, where a novel algorithm is proposed. The appendix contains background theory, the Matlab code for the algorithms, and simulation and measurement data.

2

Project Definitions

In this chapter project definitions are presented. The definitions are useful for the reader since they provide a brief overview of widely used terms in the report. The definitions and their values are set by the project group, but based on previous work.
The chapter is divided into three separate sections in which the definitions are explained. First the definitions regarding the room is presented. Next the source signal, that is the speech, is described and finally the performance metrics used to evaluate the presented algorithms are provided.

2.1 Room

In this section the room and its properties are described. The room is described by the filter, **g**, between the source and sink i.e. speaker and microphone. The output of the speaker is the speech, **s**, and the output of the sink is the reverberated speech **x**. This is illustrated in figure 2.1.

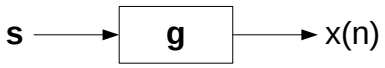


*Figure 2.1: The block diagram of the speech, **s**, corrupted by the room impulse response, **g**, producing the output **x**.*

Throughout the report the whole sequence of a signal will be denoted by a vector written with bold letters, i.e. $\mathbf{x}$. Each sample, i.e. $x(n)$, will correspond to an element in that sequence.
The effect of the room is modeled as a convolution between the room impulse response (RIR), $g(n)$, and the input signal that is the speech, $x(n)$ [8, Eq. 1]

$$y(n) = g(n) * x(n) \tag{2.1}$$

where:	n is the time index	[-]
	$y(n)$ is the output that is the signal received by the listener/microphone	[-]

In this project different room models are used. Simplified room filters will be defined by the project group when testing and comparing the algorithms. In this way it is possible to test the algorithms under known conditions. A real life RIR will also be measured in the Section of Acoustics at Aalborg University (see appendix F). This RIR will only be used as a proof of concept for the algorithms in a real life application. This gives an insight in how the algorithms perform under complex conditions.

The goal is to equalize the RIR with another filter, **h**, so that the output of the latter filter is ideally the direct speech signal, which is delayed. That is the convolution of the RIR with the equalizer filter will be a unit impulse [8, Eq. 2]

$$g(n) * h(n) = \delta(n - d) \tag{2.2}$$

where:	d is the delay	[-]
	$\delta(n)$ is the Kronecker delta, which is equal to 1 for $n = 0$ and 0 otherwise	[-]

In the frequency domain the RIR $g(n)$ can be written as [3, Eq. 1]

$$\mathcal{F}\{g(n)\} = G(\omega) = |G(\omega)| \exp\{j\phi(\omega)\} \tag{2.3}$$

where:	$\mathcal{F}\{g(n)\}$ is the Fourier transform of $g(n)$	[-]
	$\lvert G(\omega)\rvert$ is the amplitude	[-]
	$\phi(\omega)$ is the phase	[-]

The phase can be expressed as a sum of the minimum phase part $\phi_{\mathrm{m}}(\omega)$ and the difference from the minimum phase $\phi_{\mathrm{d}}(\omega)$. Rewriting equation 2.3 and knowing that any filter can be decomposed into a minimum phase and an all-pass filter yields [3, Eq. 3]

$$\begin{aligned} G(\omega) &= |G(\omega)| \exp\{j\left(\phi_{\mathrm{m}}(\omega) + \phi_{\mathrm{d}}(\omega)\right)\} && (2.4)\\ &= |G(\omega)| \exp\{j\phi_{\mathrm{m}}(\omega)\} \cdot \exp\{j\phi_{\mathrm{d}}(\omega)\} && (2.5)\\ &= M(\omega)A(\omega) && (2.6) \end{aligned}$$

where:	$M(\omega)$ is the minimum phase part $\lvert G(\omega)\rvert \exp\{j\phi_{\mathrm{m}}(\omega)\}$	[-]
	$A(\omega)$ is the all-pass part $\exp\{j\phi_{\mathrm{d}}(\omega)\}$	[-]

If $\phi_{\mathrm{d}}(\omega) = 0$ the all-pass part of $G(\omega)$ is $A(\omega) = 1$ and the system will be minimum phase. When $A(\omega) \neq 1$ the system is non-minimum phase. The RIR is in general regarded as a non-minimum phase system [12]. This means the response cannot be inverted to a stable filter unless the filter is allowed to be non-causal [3]. The inversion of a non-minimum phase filter will be further examined in chapter 3.

As explained in the introduction in chapter 1 a room will cause reverberation in a received signal. In the time domain the effects are divided into early and late reflections. According to [8] early reflections can be defined as the first 100 ms of the reflection. The project group therefore defines the late reverberations as reflections that occur 100 ms or more after the direct signal. Figure 2.2 illustrates the direct, early and late signals.

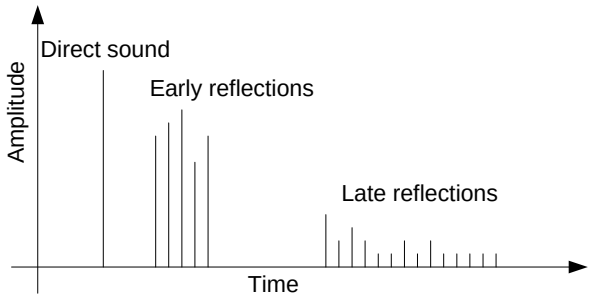


Figure 2.2: The direct, early and late reflections in the time domain. Based on [13, Fig. 1].

The early reflections are the coloration of the speech whereas the late reflections are referred to as echoes.

2.2 Speech Signal

The algorithms must be developed and tested under known and controlled conditions. A key assumption for the input to the room is that it is independent (also denoted a white process) and non-Gaussian distributed. This assumption will be used throughout the report and in the algorithms presented in the following chapters. The algorithms have been tested using iid samples drawn from a probability density function (pdf) based on the function $\cosh(\cdot)$. The pdf will be described in detail and substantiated in chapter 6. Its property is that it is very easy to adjust to fit the a priori knowledge of the source, which in this project is speech. The project group suggests the pdf to be a suitable match for the speech and it is used when testing the algorithms on the room filters mentioned in the previous section.

A real speech signal is also used for testing the algorithms. This however will only be used when testing on the real life RIR. Speech sounds are made when air passes through the vocal folds. The vocal folds can be moved together or apart, where the distance between them are called the glottis. When they are close together and air passes through them, they start to vibrate, and the sounds made are referred to as voiced sounds such as [b], [g] and all the English vowels. When they are far apart the air cannot make the vocal folds vibrate and unvoiced sounds are made such as [k] and [f]. [14]
In figure 2.3 a simplified model for speech generation is given.

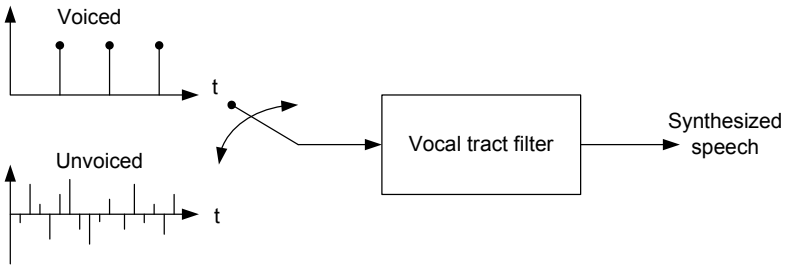


Figure 2.3: Block diagram of speech generation.

The vocal tract can be modelled as an autoregressive (AR) process. The voiced sounds are generated when the input is given as a pulse train also referred to as the glottal pulses. Unvoiced sounds are made when the input is white noise. Linear predictive coding (LPC) relies

on the model in figure 2.3 and the LP residuals consist mainly of the glottal pulses [8]. LP residuals are a whitened version of the speech.

As already mentioned, a key assumption of the input to the room is that it is iid and non-Gaussian distributed. Two scenarios can be described for this assumption. One, if the residual LPC filter is a minimum phase filter and applying the synthesized speech signal the output will be the glottal pulses. The glottal pulses are then independent and non-Gaussian distributed, this is the ideal scenario. Two, the minimum phase residual LPC filter is a linear filter which because of the Central Limit Theorem (which says that the sum of independent and identically distributed random variables will approximately be Gaussian distributed) will make the output more Gaussian. In this report it is assumed that the input to the room will be close to being non-Gaussian distributed and therefore the first scenario is assumed to be valid.

2.3 Performance Metrics

The developed algorithms are to be evaluated on many different levels. In this section the used metrics are defined and explained. The selected metrics are algorithm complexity, measured in the number of used multiplications, and the impulse-to-noise ratio. The impulse-to-noise ratio is used when testing the algorithms ability to dereverberate early and late reflections and when testing the convergence speed of the algorithms. The algorithm complexity is used when comparing the algorithms with a digital signal processor, with respect to a real time implementation.

An important metric for the hearing aid application is the complexity of the developed algorithm. A hearing aid has limited computational powers and therefore it is important that the algorithms are as effective as possible. In this project computational complexity is determined to be proportional to the number of multiplications. Other important metrics are execution time (measured in seconds or clock cycles) and the amount of required memory for program and data storage. The execution time and memory requirements will however not be analysed in this project, but an estimate of how far the algorithms are from a real time application will be considered.

During the development of the algorithm it is convenient to have a measure of how well the filter equalizes the room. According to equation 2.2 the perfect equalizer filter convolved with the RIR will result in a delayed unit impulse (a Kronecker delta). When the approximation of the inverse filter is imperfect the unit impulse will be corrupted by noise, see figure 2.4.

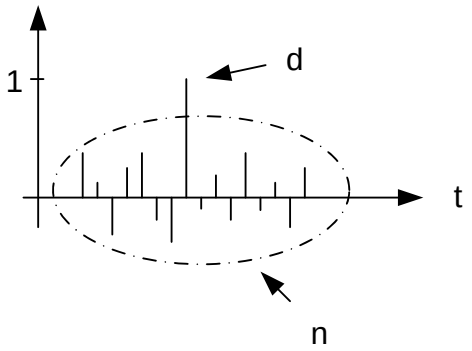


Figure 2.4: Unit impulse response d corrupted by noise n where the entire convolution result is $d + n$.

The noise n in figure 2.4 can be seen as additive noise in the convolution result of $g(n) * h(n)$. The processed speech will then be corrupted by convolution noise. This is illustrated in figure 2.5.

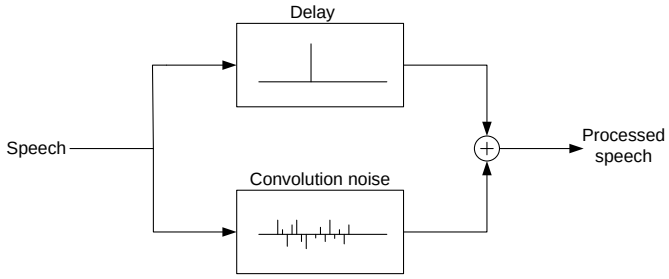


Figure 2.5: The convolution result.

The metric will be useful for comparing different algorithms since the convolution between the RIR and the equalized filter can always be performed, provided that the RIR is known. In this project the metric will be called the impulse-to-noise ratio (INR). The idea is to compare the energy in the unit impulse with the energy in the noisy signal. Since the unit impulse is always scaled to one in the implemented algorithms the INR is defined as:

$$\text{INR} \triangleq \frac{d^2}{s^2 - d^2} = \frac{1}{s^2 - 1} \qquad [-] \tag{2.7}$$

where: d is the magnitude of the unit impulse [-]
s is the magnitude of the convolved filters in equation 2.2 [-]

The properties of the INR is that when the value is below one it means the noise contains more energy than the unit impulse. When the value is exactly one the unit impulse and the noise have equally sized energy. As the noise energy decreases towards zero the INR will increase towards infinity. The INR is adopted from information theory, where it is known as signal-to-interference ratio (SIR), see for example [10].

In this chapter project definitions were made. In the next chapter it is shown how a non-minimum phase can be inverted. The used methods are however not applicable in a blind dereverberation approach and they are purely presented to provide problem insight.

3

Inversion of a Non-Minimum Phase Filter

In this section the inversion of a known filter impulse response is described. In the project the room impulse response is unknown, but the basic inverse filter theory will provide a mathematical basis which is useful and therefore the inversion of the known RIR is presented.

A desirable property of the inverted filter is that it is stable and causal. These two properties will be used throughtout the section and therefore they are defined here [15]

- If the region of convergence (ROC) contains the unit circle the filter is stable. Stability means that the output of the filter is bounded when its input is bounded.
- If the ROC extends outwards from the outermost pole (away from the center of the unit circle) the filter is causal. Causality means that the current output of the filter only depends on the current and previous inputs.

Given the RIR it is important to determine whether the filter is minimum phase or not. If the filter is minimum phase it means the direct inverse will be stable and causal [15]. As described in chapter 1 the zeros of the minimum phase RIR are located inside the unit circle. When the filter is inverted the locations of the poles and zeros will swap hence the zeros will become poles and vice versa. The following example illustrates the swapping of pole and zero locations:

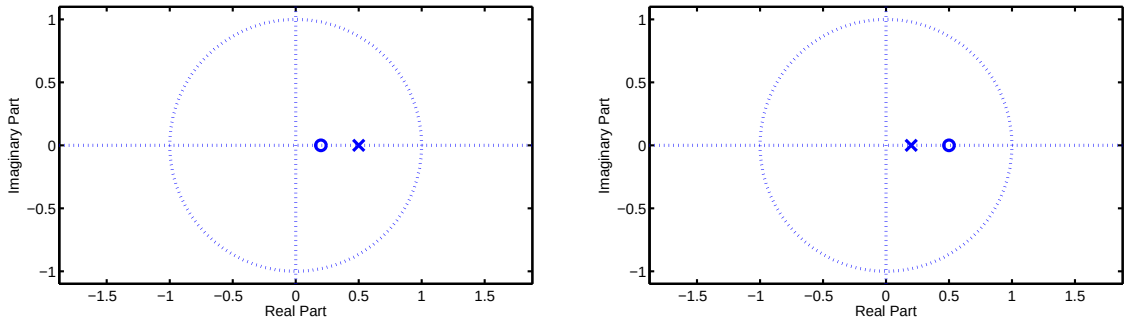
$$H(z) = \frac{1 - 0.2z^{-1}}{1 - 0.5z^{-1}} \tag{3.1}$$

$$\Downarrow$$

$$H_{\text{inv}}(z) = \frac{1}{H(z)} = \frac{1 - 0.5z^{-1}}{1 - 0.2z^{-1}} \tag{3.2}$$

where: $H(z)$ is the RIR [-]
$H_{\text{inv}}(z)$ is the inverse of the RIR $H(z)$ [-]

Figure 3.1(a) and 3.1(b) illustrates the pole zero plot for the minimum phase filter and the inverse minimum phase filter respectively.



(a) The minimum phase filter.

(b) The inverse minimum phase filter.

Figure 3.1: Pole zero plots of the minimum phase filter and its inverse given in equation (3.1) *and* (3.2).

If the RIR is non-minimum phase the corresponding inverse filter will be either unstable or noncausal because at least one pole will be outside the unit circle. However there exist a method to obtain the minimum phase inverse of the non-minimum phase RIR and thereby preserve the amplitude response. In section 2.1 it was described how the non-minimum phase filter can be divided into a minimum phase and an all-pass filter. The following example illustrates how it is done. The starting point is the non-minimum phase filter:

$$H(z) = \frac{1 - 2z^{-1}}{1 - 3/4z^{-1}} \tag{3.3}$$

The pole-zero plot of the filter is shown in figure 3.2.

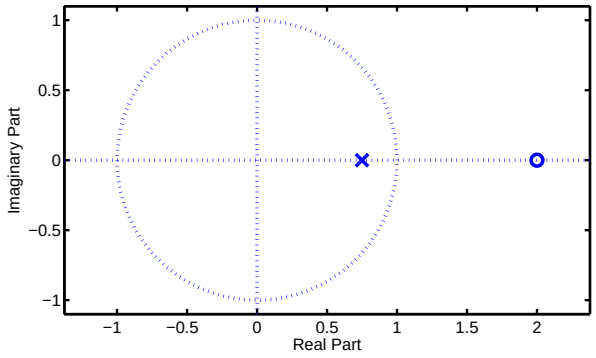


Figure 3.2: Pole-zero plot of the non-minimum phase filter in equation (3.3).

Using the decomposition method described by Oppenheim et al. in [15] the zero lying outside the unit circle (at $z = 2$) is reflected into the unit circle at the reciprocal location ($z = 1/2$). Furthermore a pole is added in the same location. If the filter coefficients are complex it is

necessary to use the reciprocal conjugate.

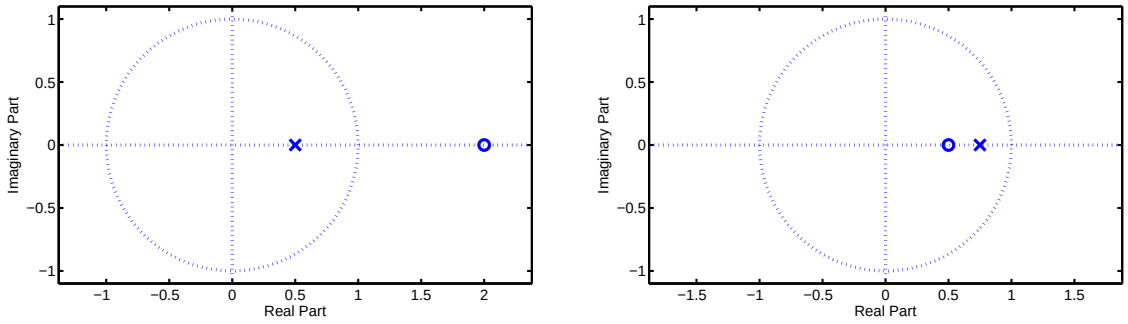
$$H(z) = \frac{1 - 2z^{-1}}{1 - 3/4z^{-1}} \cdot \frac{1 - 1/2z^{-1}}{1 - 1/2z^{-1}} \tag{3.4}$$

$$= 2 \cdot \frac{1/2 - z^{-1}}{1 - 1/2z^{-1}} \cdot \frac{1 - 1/2z^{-1}}{1 - 3/4z^{-1}} \tag{3.5}$$

$$= H_{\text{all}}(z) \cdot H_{\text{min}}(z) \tag{3.6}$$

where:	$H(z)$ is a non-minimum phase RIR	[-]
	$H_{\text{all}}(z)$ is an all-pass filter	[-]
	$H_{\text{min}}(z)$ is a minimum phase filter	[-]

The corresponding pole-zero plots of the two new filters are given in figure 3.3. Notice how the zero in $z = 2$ in figure 3.2 has a corresponding reciprocal pole in the all-pass filter (figure 3.3(a)) and how the zero is reflected into the reciprocal location in the minimum phase filter in figure 3.3(b). Using the definitions of stability and causality given in the introduction to this section, it is evident that the ROC for the two filters can extend outwards and contains the unit circle that is both filters are stable and causal.



(a) The all-pass filter.

(b) The minimum phase filter.

Figure 3.3: Pole-zero plots of the all-pass and minimum phase filters given in equation (3.5).

After the decomposition has been performed it is interesting to analyse the effect on the frequency response of the filters.
The frequency response of the RIR filter in equation (3.3) is illustrated in figure 3.4, and the response of the two filters generated via the decomposition is illustrated in figure 3.5. The all-pass filter has a flat 0 dB amplitude response, hence it's name, and the minimum phase filter has a amplitude response equal to the original non-minimum phase RIR. The sum of the two phase responses equals the phase response of the non-minimum phase RIR. The sampling frequency is set to 1 Hz in the example.

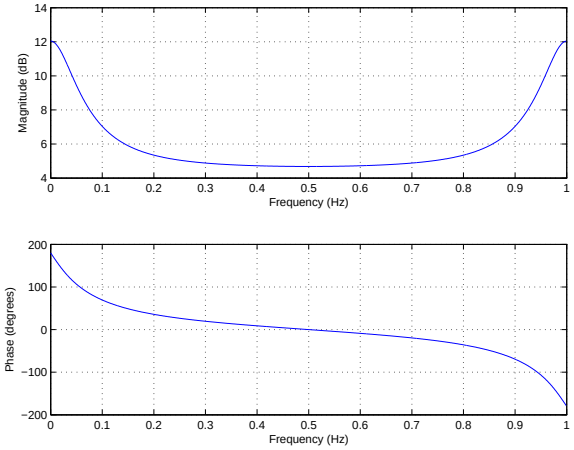
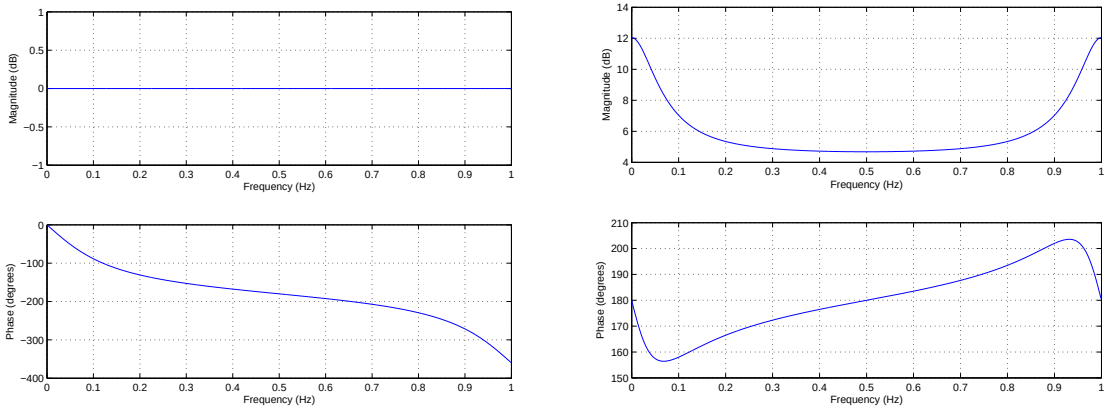


Figure 3.4: Amplitude and phase response of the non-minimum phase filter in equation (3.3).



(a) The all-pass filter.

(b) The minimum phase filter.

Figure 3.5: Amplitude and phase response of the all-pass and minimum phase filters given in equation (3.5).

Inverting the three filters by using the approach in equation (3.2) the pole-zero plots will be as illustrated in figure 3.6 and figure 3.7.

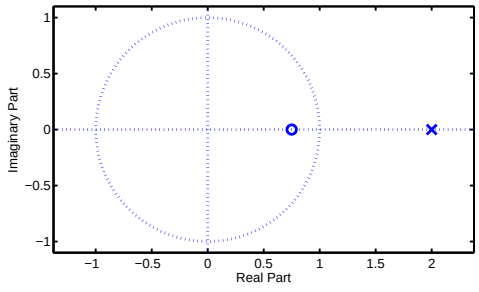
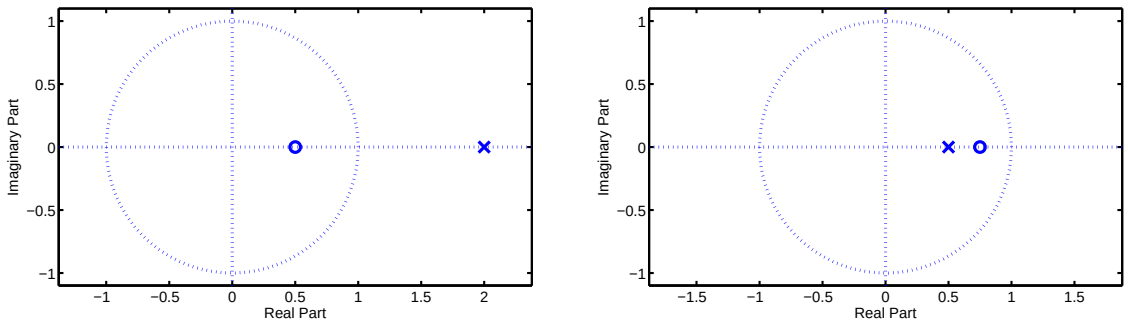


Figure 3.6: Pole-zero plot of the inverse non-minimum phase filter.



(a) The inverse all-pass filter.

(b) The inverse minimum phase filter.

Figure 3.7: Pole-zero plots of the inverse all-pass and minimum phase filters.

According to the definition of stable and causal filters it is obvious that the non-minimum phase RIR and the all-pass filter (figure 3.6 and 3.7(a)) cannot be both stable and causal because the pole in both filters lies outside the unit circle (in $z = 2$). If noncausality is acceptable the filters can be made stable by letting the ROC extend inwards towards the center of the unit circle and thereby containing the unit circle itself. The minimum phase filter in figure 3.7(b) is stable and causal.

The example illustrates how a non-minimum phase filter can be decomposed into an all-pass and a minimum phase filter. Furthermore it shows that the inverse of the non-minimum phase and the all-pass filter cannot be stable and causal at the same time. If an noncausal all-pass filter is permitted it is however possible to implement the inverse of the original non-minimum phase filter via the noncausal inverse all-pass filter and the inverse of the minimum phase filter. The example was made with a filter given in the discrete frequency domain (the z-domain). In the next section the inversion of a filter given in the discrete time domain is described.

3.1 Inversion using the FFT

The perfect equalization of the RIR can be done at the cost of a noncausal equalization impulse response, which however can be made causal by introducing a delay. This is shown for the RIR in equation (3.3) in the following example. The RIR and the frequency response is given in figure 3.8.
When the RIR, defined in the discrete time domain, has to be inverted a useful approach is to Fourier transform the impulse response to the discrete frequency domain and then invert the filter as described in the previous example. When the filter has been inverted the inverse Fourier transform is applied and the inverse RIR is obtained.

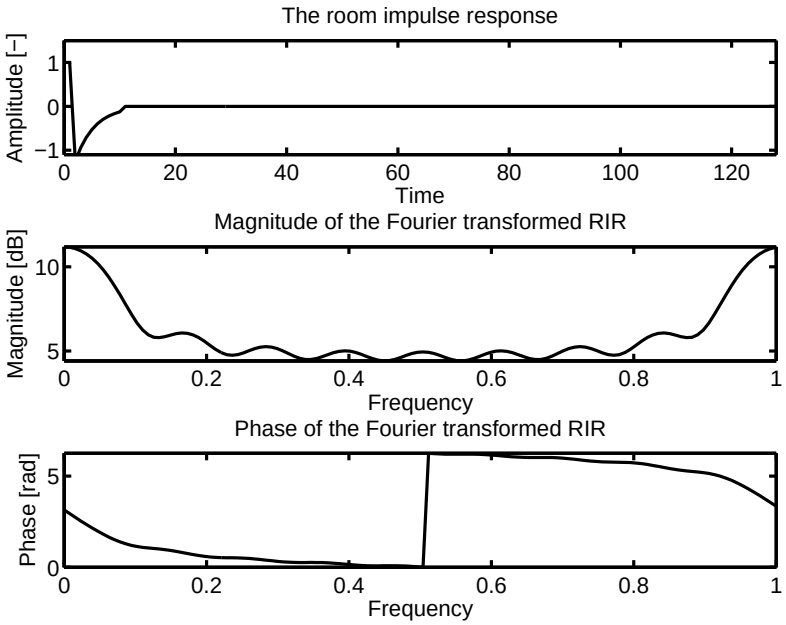


Figure 3.8: Room impulse response, amplitude, and phase response of the non-minimum phase filter in equation (3.3).

Next the approach described in the previous sectin is used to invert the Fourier transformed RIR. The amplitude and phase response is given for the inverse RIR in figure 3.9

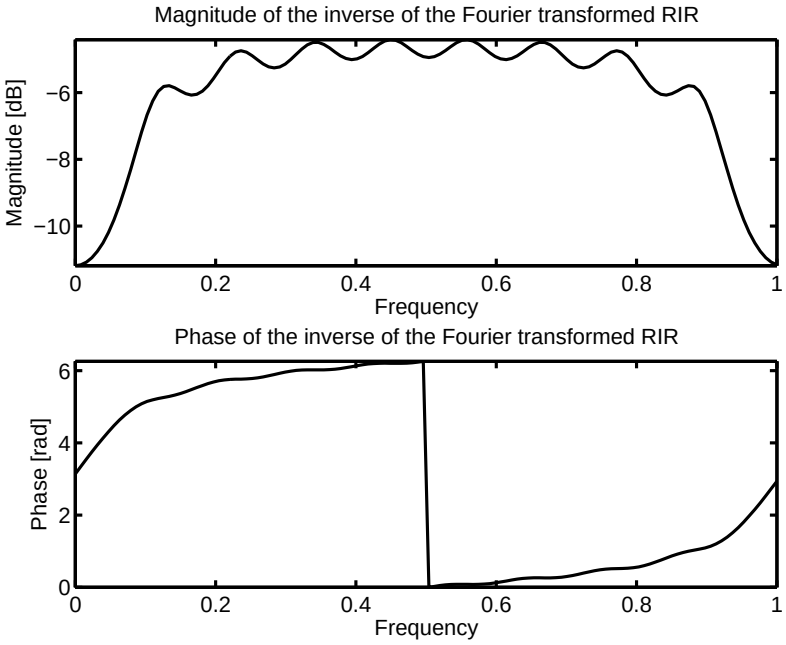


Figure 3.9: Amplitude and phase response of the inverse of the Fourier transformed RIR.

As the phase response shows the group delay is positive, hence the inverse filter is noncausal due to the pole in $z = 2$. The cost of a stable filter is as mentioned introducing a delay, shown in figure 3.10. The delay is implemented by using FFT shifting after the inverted system has been inverse Fourier transformed.

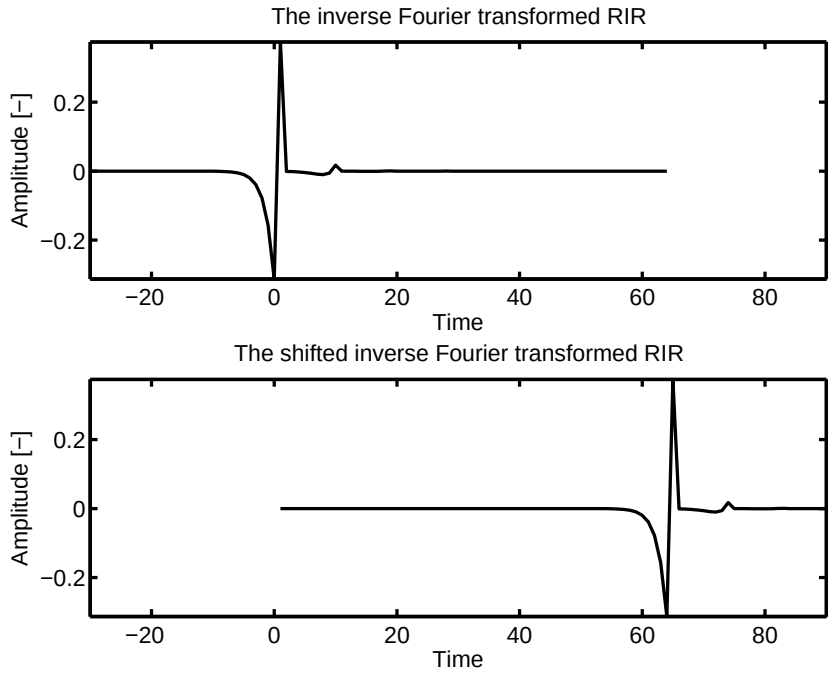


Figure 3.10: Introducing a delay such that the inverse RIR becomes causal.

The convolution of the RIR and its inverse is given in figure 3.11.

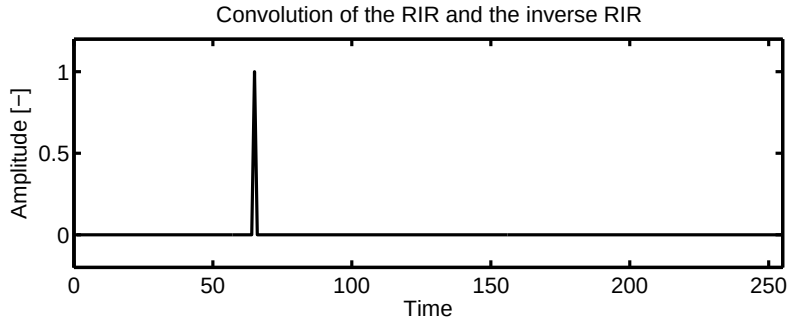


Figure 3.11: Convolution of the RIR and its inverse.

The result of the convolution is a perfect Kroenecker delta with an impulse to noise ratio (INR) equal to infinity, meaning that all the energy of the signal is in the single impulse.

The example showed how the RIR, defined in the time domain can be inverted by using the FFT and the inverse FFT.

In this project the non-minimum phase RIR is unknown and therefore another approach has to be used. A direct estimation of the inverse RIR is desired and the approach will be based on higher-order statistics and an information theoretic approach. Chapter 4 will describe the basics of HOS and the use of information theory. In the following chapters the different approaches are analysed.

4

Deconvolution using Statistical Techniques

In chapter 3 it was shown that the inverse RIR could be determined if the RIR is known. The RIR is however not given which leads to the two general approaches: estimation of the RIR and then finding its inverse or the direct estimation of the inverse RIR. As already mentioned the approach of this project is the direct estimation of the inverse RIR.
In figure 4.1 the general setup for the dereverberation problem, which will be used throughout the report, is shown.

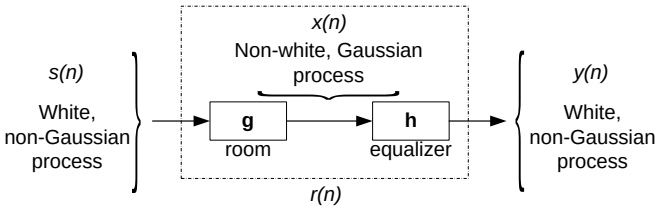


Figure 4.1: Block diagram of the dereverberation problem. $s(n)$ is the desired speech signal, $x(n)$ is the reverberated signal, and $y(n)$ is the dereverberated signal.

The setup consists of the original desired speech signal, $\mathbf{s}$, which is assumed to be a white, non-Gaussian process. That it is white denotes that its samples are zero mean iid random variables. The room represented by the RIR, $\mathbf{g}$, and the speech are unknown, only the received signal $x(n) = s(n) * g(n)$ is known. The objective is therefore to determine the inverse RIR, $h(n)$, such that $r(n) = g(n) * h(n) = \delta(n-d)$, where d denotes the introduced delay, such that $y(n) = s(n-d)$.

Figure 4.2 shows the main approaches, which are considered for use on the blind speech dereverberation problem: a higher-order-statistics approach and an information theoretic approach.

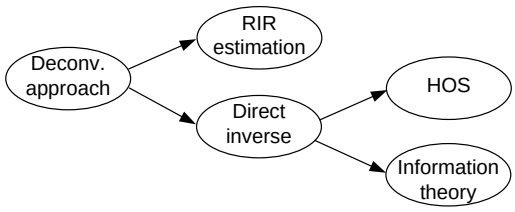


Figure 4.2: Overview of the approaches to the deconvolution problem.

The two following sections will briefly describe the HOS and information theoretic approach for direct estimation of the inverse RIR.

4.1 Higher-Order Statistics

The main advantage of using HOS follows by the assumption that the received signal, $x(n)$, can, by the central limit theorem (CLT), be considered as being Gaussian distributed. The CLT states that the distribution of the sum of independent and identical distributed (iid) signals is approximately Gaussian [16]. A reverberated speech signal is a multipath signal, and it is represented as weighted, delayed, and summed copies of the same signal. This means that the reverberated signal, $x(n)$, can very roughly be considered as being Gaussian distributed with respect to the desired speech signal, $s(n)$. This entails that the samples of $\mathbf{x}$ will be statistically dependent. All Gaussian distributed signals have HOS equal to zero. In the introduction to this chapter it was stated that the input signal to the algorithm was assumed non-Gaussian distributed. The idea is therefore to establish a cost function that maximizes the HOS of the reverberated signal, which entails that the processed signal should obtain a pdf that is non-Gaussian, that is the room impulse response has been removed and any statistical dependencies over time has been removed. The assumption is then that the obtained signal approximates the desired direct speech signal.

In section 4.3 the general theory of HOS will be given. The notation and definitions will be used throughout the report. First however the information approach to blind speech dereverberation will be described.

4.2 Information theory

The concept of information is defined first with inspiration from [17]. Basically information is whatever a source transmits to a receiver over a channel, where the channel in this project is the room. The amount of information transmitted must therefore be related to the uncertainty that occurs at the receiver side. The information is then a function of the probability of occurrence for an event, that is the information of an event $I(event) \triangleq f(P(event))$. If an event has a small possibility of happening the information is high, and if the event is very likely to occur the information is low, hence for event A and B, then $I(A) > I(B)$ for $P(A) < P(B)$. In short this means that the amount of information only depends on the probability of the occurrence and not on the content of the event. The function used for information is defined as the logarithm of the probability of the event, $I(A) \triangleq f(P(A)) \triangleq -\log(P(A))$, where the base of the logarithm is the exponential function in the following chapter.

Relating this to figure 4.1 it is first noted that each speech sample, $s(n)$, will be affected by the room and the resulting observed samples, $x(n)$, will be corrupted by delayed and weighted versions of itself. From an information theoretic point of view this entails that the information each sample, $s(n)$, contains about the speech has been changed. It will therefore be analysed if deconvolution can be performed by altering the information of each sample, $x(n)$.

The information approaches presented in chapter 6, operate on basis of the entropy of the signal **x**. Entropy, denoted $H(event)$, is defined in information theory as the average information each sample carries, that is it describes the average uncertainty of each event. For the event A, $H(A) \triangleq E[I(A)] = -E[log(P(A))]$. If the sample carries no new information the entropy is zero and if the the sample carries no redundant information maximum entropy has been obtained.

The following two chapters will analyse the HOS and information theory methods and their relations. Chapter 5 deals with the kurtosis approach and chapter 6 deals with the information approach. For each method, articles will be presented, implementations will be made and pros and cons will be analysed.

Before the kurtosis method is presented in chapter 5, a brief introduction to the general HOS theory will be given in the following section.

4.3 Higher-Order Statistics Theory

In this section basic HOS theory is presented. The purpose is to explain the definitions of 3rd and 4th order moments and cumulants and the use of them. The main reasons for using HOS (3rd, 4th, ... , nth-order) is that the 2nd order cumulant is phase blind, that is it does not preserve phase information about the filter, and that the 3rd and 4th order cumulants are equal to zero, when the input is a Gaussian process.

Given the system illustrated in figure 4.3 it is shown in the following, that the autocovariance of the output does not contain any information about the phase of the filter.

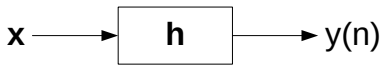


Figure 4.3: A single input single output filter system. The input x *is stationary white noise and the filter* h *is stable and causal.*

The autocorrelation m_2^y of the output y is defined as

$$m_2^y(\tau) \triangleq E\left[y(n)y(n+\tau)\right] \tag{4.1}$$

and the autocovariance is defined as

$$c_2^y(\tau) \triangleq E\left[y(n)y(n+\tau)\right] - \left(E\left[y(n)\right]\right)^2 = m_2^y(\tau) - \left(m_1^y\right)^2 \tag{4.2}$$

where: m_1^y is the mean of y, which is defined as $E\left[y(n)\right]$

If $y(n)$ is zero-mean the autocorrelation is equal to the autocovariance. In appendix A the autocovariance of $y(n)$ is defined in terms of **x** and **h**. Furthermore the autocovariance is

Fourier transformed yielding the power spectrum $S_2^y(f)$ given in equation (A.4) and repeated here

$$\begin{aligned} S_2^y(f) &= \mathcal{F}\left\{C_2^y(\tau)\right\} \\ S_2^y(f) &= |H(f)|^2 S_2^x(f) \end{aligned} \tag{4.3}$$

where: $\mathcal{F}$ is the Fourier transform operator
$S_2^x(f)$ is the power spectrum of $x(n)$ [-]

Examining equation (4.3) it is clear that the phase information is lost, because the frequency response $H(f)$ of the filter is squared, which means only the amplitude response is preserved. Since the phase information is required for system identification when the system is not minimum phase, it is necessary to use a more advanced metric when estimating the system response: the HOS cumulants.

The HOS cumulants are based on HOS moments. Actually the mean and the autocorrelation presented above are the 1st and 2nd order moments, and the autocovariance is the 2nd order cumulant. Given a real, stationary discrete time signal $x(k)$ the nth order moment is defined as [5, Eq. 18]

$$m_n^x(\tau_1, \tau_2, ..., \tau_{n-1}) \triangleq E\left[x(k)x(k+\tau_1)x(k+\tau_2)...x(k+\tau_{n-1})\right] \tag{4.4}$$

The nth order cumulant is given as [18, Eq. 2.5]

$$cum\left[x_1, x_2, ..., x_n\right] = \sum_{p=1}^{n} \left\{ (-1)^{p-1}(p-1)! \cdot E\left[\prod_{i\in s_1} x_i\right] \cdot E\left[\prod_{i\in s_2} x_i\right] \cdots E\left[\prod_{i\in s_p} x_i\right] \right\} \tag{4.5}$$

where: p denotes the partions
s_i is the set of integers in the ith partion

To examine the use of equation (4.5) the 3rd order cumulant is calculated in the following. When $p = 1$, one set is required. Since all numbers from 1 to n,$(n = 3)$ has to be used when generating sets the set for $p = 1$ has to contain all values. When $p = 2$, two sets are required. Again all numbers have to be used, but in this case several partions containing two sets and all numbers can be made [18]

$$\begin{array}{lllll} p=1 & s_1=\{1,2,3\} & & & \\ p=2 & s_1=\{1\} & s_2=\{2,3\} & & \\ & s_1=\{2\} & s_2=\{1,3\} & & \\ & s_1=\{3\} & s_2=\{1,2\} & & \\ p=3 & s_1=\{1\} & s_2=\{2\} & s_3=\{3\} & \end{array}$$

Inserting the sets s_1, s_2, and s_3 into equation (4.5) yields

$$\begin{aligned} cum\left[x_1, x_2, x_3\right] &= \underbrace{E\left[\prod_{i\in s_1} x_i\right]}_{p=1} \underbrace{-E\left[\prod_{i\in s_1} x_i\right] E\left[\prod_{i\in s_2} x_i\right]}_{p=2} \underbrace{+2E\left[\prod_{i\in s_1} x_i\right] E\left[\prod_{i\in s_2} x_i\right] E\left[\prod_{i\in s_3} x_i\right]}_{p=3} \\ &= E\left[x_1x_2x_3\right] - E\left[x_1\right] \cdot E\left[x_2x_3\right] - E\left[x_2\right] \cdot E\left[x_1x_3\right] - E\left[x_3\right] \cdot E\left[x_1x_2\right] \\ &\quad + 2E\left[x_1\right] \cdot E\left[x_2\right] \cdot E\left[x_3\right] \end{aligned}$$

Setting x equal to a real stationary random process that is $x_1 = X(k)$, $x_2 = X(k+\tau_1)$, and $x_3 = X(k+\tau_2)$ the above equation can be rewritten noting that $cum\left[X(k), X(k+\tau_1), X(k+\tau_2)\right] = c_3^x(\tau_1, \tau_2)$ and using the definition of moments in equation (4.4)

$$\begin{aligned} c_3^x(\tau_1,\tau_2) &= E\left[X(k)X(k+\tau_1)X(k+\tau_2)\right] - E\left[X(k)\right]\cdot E\left[X(k+\tau_1)X(k+\tau_2)\right] \\ &\quad - E\left[X(k+\tau_1)\right]\cdot E\left[X(k)X(k+\tau_2)\right] - E\left[X(k+\tau_2)\right]\cdot E\left[X(k)X(k+\tau_1)\right] \\ &\quad + 2E\left[X(k)\right]\cdot E\left[X(k+\tau_1)\right]\cdot E\left[X(k+\tau_2)\right] \\ &= m_3^x(\tau_1,\tau_2) - m_1^x\left[m_2^x(\tau_1) + m_2^x(\tau_2) + m_2^x(\tau_1-\tau_2)\right] + 2(m_1^x)^3 \end{aligned} \tag{4.6}$$

Previously it was determined that the autocovariance does not preserve the phase. In the following it is shown that the higher order cumulants preserve the phase. The starting point is the third order cumulant $c_3^y(\tau_1, \tau_2)$ defined in equation (4.6). Since the moments of order one and two do not preserve phase information the focus is on the third order moment $m_3^y(\tau_1, \tau_2)$. Using the definition in equation (4.4)

$$m_3^y(\tau_1,\tau_2) = E\left[y(k)y(k+\tau_1)y(k+\tau_2)\right] \tag{4.7}$$

$$\begin{aligned} &= E\left[\sum_{m=-\infty}^{\infty} h(m)x(k-m)\sum_{n=-\infty}^{\infty} h(n)x(k+\tau_1-n)\sum_{p=-\infty}^{\infty} h(p)x(k+\tau_2-p)\right] \\ &= E\left[\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\sum_{p=-\infty}^{\infty} h(m)h(n)h(p)x(k-m)x(k+\tau_1-n)x(k+\tau_2-p)\right] \\ &\overset{k'=k-m}{=} \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\sum_{p=-\infty}^{\infty} h(m)h(n)h(p)E\left[x(k')x(k'+m+\tau_1-n)x(k'+m+\tau_2-p)\right] \\ &= \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\sum_{p=-\infty}^{\infty} h(m)h(n)h(p)m_3^x(m+\tau_1-n, m+\tau_2-p) \\ &\overset{n=m+q\ r=m+p}{=} \sum_{m=-\infty}^{\infty}\sum_{q=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} h(m)h(m+q)h(m+r)m_3^x(\tau_1-q,\tau_2-r) \\ &= \sum_{q=-\infty}^{\infty}\sum_{r=-\infty}^{\infty} m_3^h(q,r)m_3^x(\tau_1-q,\tau_2-r) \\ &= m_3^h(q,r) ** m_3^x(\tau_1-q,\tau_2-r) \end{aligned} \tag{4.8}$$

where: $**$ denotes a two dimensional convolution

Fourier transforming the nth order moment function yields [18, Eq. 3.45]

$$M_n^x\left(\omega_1,\omega_2,...,\omega_{n-1}\right) \triangleq \sum_{\tau_1=-\infty}^{\infty}\cdots\sum_{\tau_{n-1}=-\infty}^{\infty} m_n^x\left(\tau_1,\tau_2,...,\tau_{n-1}\right) exp\left[-j\left(\omega_1\tau_1+\omega_2\tau_2+...+\omega_{n-1}\tau_{n-1}\right)\right] \tag{4.9}$$

and rewriting using [18, Eq. 3.47]

$$M_n^x\left(\omega_1,\omega_2,...,\omega_{n-1}\right) = X(\omega_1)\cdot X(\omega_2)\cdots X(\omega_{n-1})\cdot X^*(\omega_1+\omega_2+...+\omega_{n-1}) \tag{4.10}$$

This means equation (4.8) can be transformed to

$$\mathcal{F}\left\{m_3^y(\tau_1,\tau_2)\right\} = M_3^y(f_1,f_2) = \mathcal{F}\left\{m_3^h(q,r) * * m_3^x(\tau_1-q,\tau_2-r)\right\} \tag{4.11}$$

$$\begin{aligned} M_3^y(f_1,f_2) &= \mathcal{F}\left\{m_3^h(q,r)\right\} \cdot \mathcal{F}\left\{m_3^x(\tau_1-q,\tau_2-r)\right\} \\ &= M_3^h(f_1,f_2) \cdot M_3^x(f_1,f_2) \\ &= H(f_1)H(f_2)H^*(f_1+f_2) \cdot M_3^x(f_1,f_2) \end{aligned} \tag{4.12}$$

The above equation illustrates that the 3rd order moment (and its cumulant in equation (4.6)) preserves the phase. The result can be expanded to higher order moments as well. The Fourier transformed 3rd order moment is known as the *bispectrum* and the 4th order moment is referred to as the *trispectrum*.
Now that the 2nd and higher order moments and cumulants have been defined they can be applied to a simple example. Given a minimum phase and a non-minimum phase the autocorrelation (2nd order moment) and the 3rd order moment is calculated to illustrate that the 2nd order moment is incapable of distinguishing between the two systems.

$$\begin{aligned} H_{MP}(z) &= (1-az^{-1})(1-bz^{-1}) \quad &\Rightarrow y_{MP}(k) &= x(k) - (a+b)x(k-1) + abx(k-2) \\ H_{NMP}(z) &= (1-az)(1-bz^{-1}) \quad &\Rightarrow y_{NMP}(k) &= (1+ab)x(k) - bx(k-1) - a(k+1) \end{aligned}$$

where:	$H_{MP}(z)$ is a minimum phase filter with the zeros inside the unit circle	[-]
	$H_{NMP}(z)$ is a non-minimum phase filter with one zero inside and one zero outside the unit circle	[-]
	a is a filter coefficient $\lvert a\rvert < 1$	[-]
	b is a filter coefficient $\lvert b\rvert < 1$	[-]

The moments are calculated using equation (4.1) and (4.7), 2nd and 3rd order moments, respectively. For example the 2nd order moment for the minimum phase filter is (assuming that $x(k)$ is an independent sequence)

$$\begin{aligned} m_2^y(\tau) &= E\left[y_{MP}(k)y_{MP}(k+\tau\right] \\ &= E\left[(x(k)-(a+b)x(k-1)+abx(k-2))(x(k+\tau)-(a+b)x(k-1+\tau)+abx(k-2+\tau))\right] \\ m_2^y(1) &= E\left[(x(k)-(a+b)x(k-1)+abx(k-2))(x(k+1)-(a+b)x(k)+abx(k-1))\right] \\ &= -(a+b)-(a+b)\cdot ab = -(a+b)(1+ab) \end{aligned} \tag{4.13}$$

Results for other combinations of moments and delays is given in table 4.1.

Moment	Minimum phase	Non-minimum phase
$c_2^y(0)$	$1+(a+b)^2+a^2b^2$	$(1+ab)^2+a^2+b^2$
$c_2^y(1)$	$-(a+b)(1+ab)$	$-(a+b)(1+ab)$
$c_2^y(2)$	ab	ab
$c_2^y(\tau>2)$	0	0
$c_3^y(0,0)$	$1-(a+b)^3+a^3b^3$	$1+3ab+3a^2b^2+2a^3b^3$
$c_3^y(1,1)$	$(a+b)^2-(a+b)a^2b^2$	$(1+ab)b^2-a(1+ab)^2$
$c_3^y(2,2)$	a^2b^2	$-ab^2$
$c_3^y(\tau_1>2,\tau_2>2)$	0	0

Table 4.1: 2nd and 3rd order moments for minimum and non-minimum phase filters. Based on [5, Table 1].

Analysing the table it is clear that the 2nd order moment cannot distinguish between the minimum and the non-minimum phase filter. When the 3rd order moment is applied the difference between the two filters becomes apparent.

The nth order cumulant for $x(k)$ being a non-Gaussian, stationary random signal can also be defined as [5, Eq. 19]

$$c_n^x(\tau_1,\tau_2,...,\tau_{n-1}) \triangleq m_n^x(\tau_1,\tau_2,...,\tau_{n-1}) - m_n^G(\tau_1,\tau_2,...,\tau_{n-1}) \tag{4.14}$$

Note that the above equation is only defined for $n=3$ and $n=4$. The 2nd order cumulant is the autocovariance defined in equation (4.2). The parameter m_n^G is the nth order moment of a Gaussian signal with mean and autocorrelation equal to $x(n)$. If $x(k)$ is also Gaussian the cumulant will be zero since $m_n^x = m_n^G$. This proves that the 3rd and 4th order cumulants are not affected by Gaussian noise, which is a very useful property. In [5] it is stated that the cumulants are also zero for all higher orders. Furthermore the 3rd order cumulant is also zero if the examined random process is symmetrically distributed [5].

Finally it is worth mentioning three commonly used parameters, which are defined for zero delay ($\tau_1=\tau_2=\tau_3=0$) and zero mean ($m_1^x=0$), that is the moment equals its cumulant, [5, Eq. 25]

$$c_2^x(0) = E\left[x^2(n)\right] = \gamma_x^2 \tag{4.15}$$

$$c_3^x(0,0) = E\left[x^3(n)\right] = \gamma_x^3 \tag{4.16}$$

$$c_4^x(0,0,0) = E\left[x^4(n)\right] - 3\cdot\left(\gamma_x^2\right)^2 = \gamma_x^4 \tag{4.17}$$

where:
- γ_x^2 is the variance [-]
- γ_x^3 is the skewness [-]
- γ_x^4 is the kurtosis [-]

The latter parameter is defined as the normalized kurtosis when the formula for calculation is [6, eq. (2)]

$$\gamma_x^4 N = \frac{E\left[x^4(n)\right]}{\left(\gamma_x^2\right)^2} - 3 \tag{4.18}$$

The kurtosis is a measure of peakedness and if the analysed distribution has many large values in the middle and small values at the tails it is called leptocurtic and has a positive kurtosis.

If the distribution is the opposite of the leptocurtic that is with large tail values or even just flatly distributed it is platycurtic and has a negative kurtosis.
This concludes the brief presentation of higher-order statistics. The theory will be used in chapter 5

5

Kurtosis Approach

In this chapter the class of kurtosis maximization algorithms performing blind deconvolution is presented. The basic idea is to maximize the kurtosis of the received reverberated speech signal. The assumption is that the incoming signals are iid hence using the CLT the signal can be assumed to be Gaussian distributed. The algorithms are then used to maximize the kurtosis of the signal and since the kurtosis can be thought of as a measure of peakedness it will lead to an algorithm, which moves the signal away from the Gaussian distribution. The most important property is that the non-Gaussian signal is equal to the original speech signal if the kurtosis of the speech equals the kurtosis of the equalized signal, since moving the reverberated speech away from being Gaussian distributed, the impact of the RIR on the speech is removed. The maximization of the kurtosis can be performed and implemented in several different ways, but basically the algorithms can be divided into two categories: the constrained and the unconstrained updating. As illustrated in figure 5.1 the presented unconstrained algorithms are based on steepest ascent algorithms where the presented constrained algorithms are based on either closed-form solutions or steepest ascent methods.

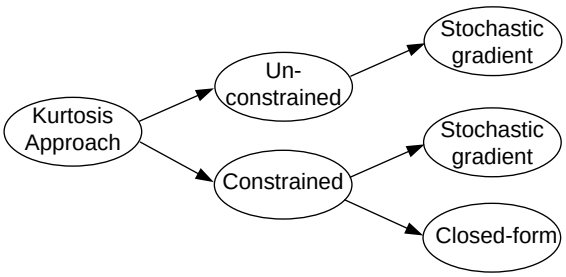


Figure 5.1: Various approaches using the kurtosis approach. They are all presented in this chapter.

First an interesting unconstrained algorithm is presented in the next section.

5.1 Unconstrained Maximization

In [6] Gillispie et al. present an unconstrained optimization algorithm. The LP residuals in clean speech have strong peaks (high kurtosis) due to the glottal pulses but for reverberated speech they are spread in time (low kurtosis). The approach is therefore to maximize the kurtosis in LP residuals. [6] also uses the kurtosis of LP residuals as performance metric for the algorithm. A steepest ascent algorithm is presented where the cost function is given as the normalized kurtosis in equation (4.18). The block diagram for the algorithm is given in figure 5.2.

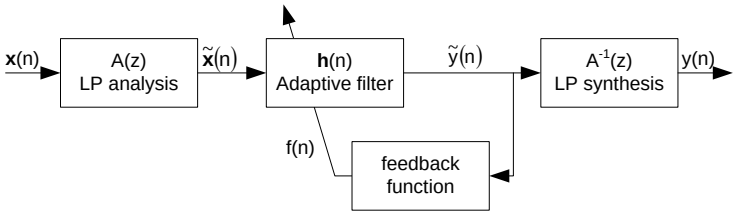


Figure 5.2: Block diagram of the steepest ascent algorithm based on [6, fig. 2 (a)].

The adaptive filter $\mathbf{h}(n)$ is controlled by the feedback function $f(n)$ given by the chosen cost function (described later on). Assuming linearity for the filters, the LP artifacts generated in the LP synthesis, can be avoided by the equivalent block diagram shown in figure 5.3. The cost is an extra filter.

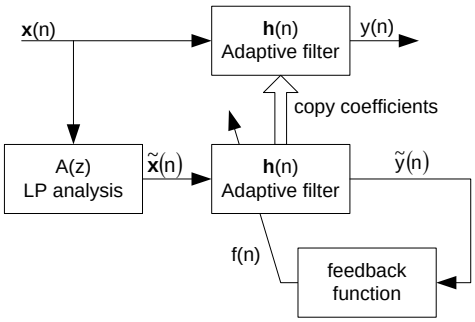


Figure 5.3: Modified block diagram such that LP artifacts are avoided, based on [6, fig. 2 (b)].

The general idea is to maximize the chosen cost function. This is done by differentiating the cost function, $J(n)$, with respect to the filter coefficients $\mathbf{h}(n)$ and then update the filter $\mathbf{h}(n+1)$ in the direction the cost function increases the most. To assure that the algorithm ascends for each time step n the following requirement is made

$$J(\mathbf{h}(n+1)) > J(\mathbf{h}(n)) \tag{5.1}$$

The update is then given by

$$\begin{aligned} \mathbf{h}(n+1) &= \mathbf{h}(n) + \mu\Delta\mathbf{h} \\ &= \mathbf{h}(n) + \mu\frac{\partial J(\mathbf{h}(n))}{\partial \mathbf{h}(n)} \end{aligned} \tag{5.2}$$

where: μ is the step size $0 < \mu < 1$ [-]

The step μ is introduced such that the algorithm does not adapt to some present gradient noise which of course is a trade off to convergence speed. The update step is illustrated in figure 5.4.

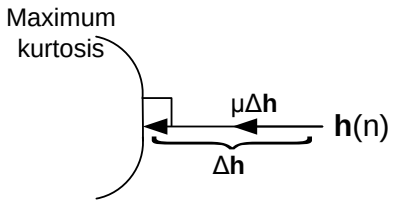


Figure 5.4: The algorithm update using the weighting factor μ such that gradient noise is avoided.

The update in equation (5.2) satisfies the ascent requirement in equation (5.1). This can be shown using the 1st order Taylor expansion around $\mathbf{h}(n)$

$$\begin{aligned} J(\mathbf{h}(n+1)) &\simeq J(\mathbf{h}(n)) + \left(\frac{\partial J(\mathbf{h}(n))}{\partial \mathbf{h}(n)}\right)^T (\mathbf{h}(n+1) - \mathbf{h}(n)) \\ &= J(\mathbf{h}(n)) + \mu \left\| \frac{\partial J(\mathbf{h}(n))}{\partial \mathbf{h}(n)} \right\|^2 \end{aligned} \tag{5.3}$$

Knowing that $0 < \mu < 1$ the requirement in equation (5.1) is satisfied.

The iterative algorithm should maximize the cost function given by the normalized kurtosis of the LP residuals (for simplicity the time index n is ignored). From equation (4.18) the cost function is given by

$$\gamma_x^4 N = J((h)) = \frac{\mathrm{E}[\tilde{y}^4]}{\mathrm{E}^2[\tilde{y}^2]} - 3 \tag{5.4}$$

where: $\tilde{y}(n)$ is the LP residual of $y(n)$ [-]

The gradient of J with respect to $\mathbf{h}$ is now given as

$$\begin{aligned} \frac{\partial J((h))}{\partial \mathbf{h}} &= \frac{4\mathrm{E}[\tilde{y}^3\tilde{x}]\mathrm{E}^2[\tilde{y}^2] - 4\mathrm{E}[\tilde{y}^4]\mathrm{E}[\tilde{y}^2]\mathrm{E}[\tilde{y}\tilde{x}]}{\mathrm{E}^4[\tilde{y}^2]} \\ &= \frac{4\left(\mathrm{E}[\tilde{y}^3\tilde{x}]\mathrm{E}[\tilde{y}^2] - \mathrm{E}[\tilde{y}^4]\mathrm{E}[\tilde{y}\tilde{x}]\right)}{\mathrm{E}^3[\tilde{y}^2]} \end{aligned} \tag{5.5}$$

According to [6, eq. (4)] the gradient can be approximated to

$$\frac{\partial J((h))}{\partial \mathbf{h}} \simeq \left(\frac{4\left(\mathrm{E}[\tilde{y}^2]\tilde{y}^2 - \mathrm{E}[\tilde{y}^4]\right)\tilde{y}}{\mathrm{E}^3[\tilde{y}^2]}\right)\tilde{\mathbf{x}} = f\tilde{\mathbf{x}} \tag{5.6}$$

where: f is the feedback function [-]

$\mathrm{E}[\tilde{y}^2(n)]$ and $\mathrm{E}[\tilde{y}^4(n)]$ are the sampled mean estimates and they can be calculated recursively. The steepest ascent algorithm is then given by

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu \frac{\partial J(\mathbf{h}(n))}{\partial \mathbf{h}(n)} \tag{5.7}$$

$$= \mathbf{h}(n) + \mu f(n)\tilde{\mathbf{x}}(n) \tag{5.8}$$

where

$$f(n) = \frac{4\left(\mathrm{E}[\tilde{y}^2(n)]\tilde{y}^2(n) - \mathrm{E}[\tilde{y}^4(n)]\right)\tilde{y}(n)}{\mathrm{E}^3[\tilde{y}^2(n)]} \tag{5.9}$$

$$\mathrm{E}[\tilde{y}^2(n)] = \beta\mathrm{E}[\tilde{y}^2(n-1)] + (1-\beta)\tilde{y}^2(n) \tag{5.10}$$

$$\mathrm{E}[\tilde{y}^4(n)] = \beta\mathrm{E}[\tilde{y}^4(n-1)] + (1-\beta)\tilde{y}^4(n) \tag{5.11}$$

where: β is the weighting factor in the recursive update [-]

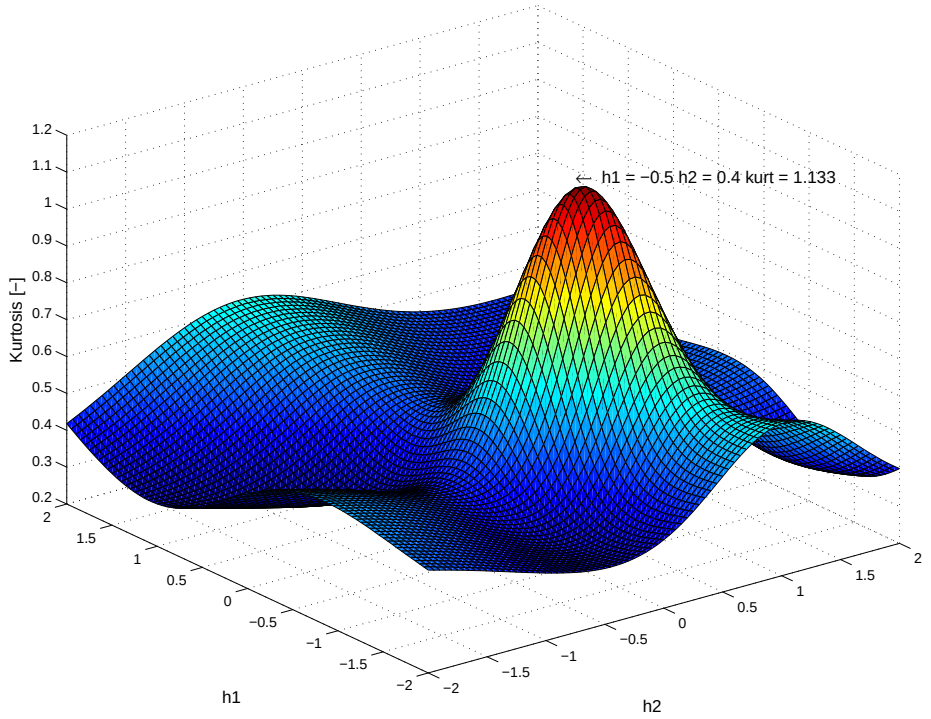
In this section an unconstrained method for kurtosis maximization was presented. In the next section experimental results performed on the algorithm is given.

Experimental Results

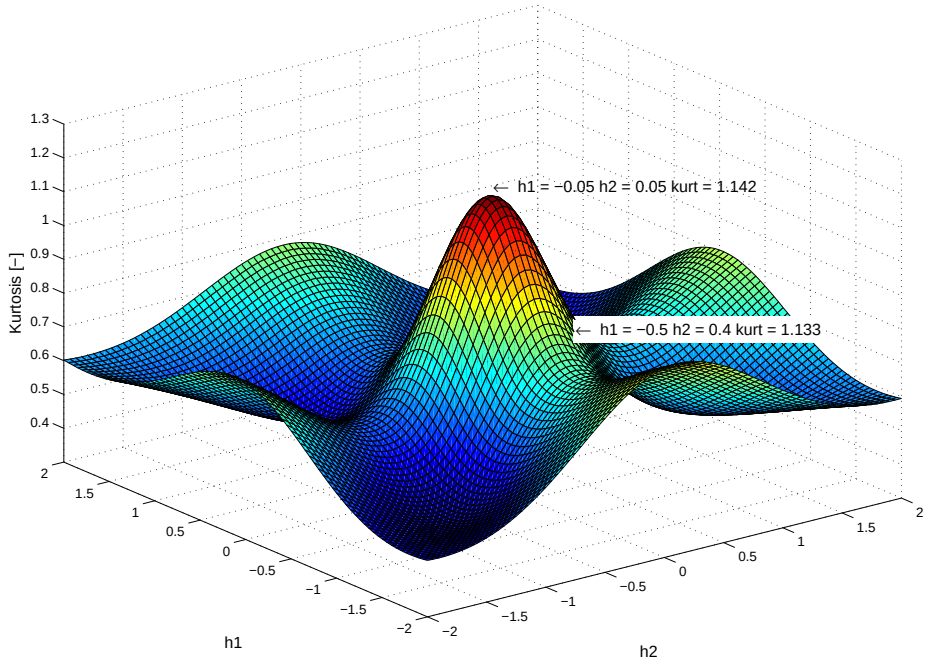
The steepest ascent algorithm in [6] is implemented and tested with regards to the impulse-to-noise ratio, which is defined in equation (2.7). The Matlab implementation is located on the CD in the directory `Algorithms/KurtMax.m`.
The used test signal, $\mathbf{s}$, is pseudo random iid samples drawn from the $\cosh(\cdot)$ distribution. This distribution is described and substantiated in the information approach to blind deconvolution, which is presented in chapter 6. The length of the test signal is set to $100 \cdot 10^3$ samples such that the algorithm will reach convergence. For all tests the recursive weighting factor β is 0.99. Because the non-minimum phase RIR have zeros and poles, the algorithm will be tested on a 16 tap all-pole RIR and a 16 tap all-zero RIR, which are each others inverse. First however the convergence ability of the algorithm is analysed on a simpler 3 tap IIR filter, where the coefficients have been chosen so that they can be reused in the 16 tap filters and approximate the weighting value of a real echo.

$$H_{RIR}(z) = \frac{1}{1 - 0.5z^{-1} + 0.4z^{-2}} \tag{5.12}$$

The above definition means that the RIR can be equalized by the use of a 3 tap FIR filter, that is the algorithm only needs to adjust three filter coefficients. The convergence capability of the algorithm can be examined in several different ways.
First the performance landscape for the algorithm is analysed (illustrated in figure 5.5). The performance landscapes are made by deconvolving the reverberated signal and determining the kurtosis value for every combination of the coefficients h_1 and h_2 in the FIR equalizer filter $H_{EQ}(z) = 1 + h_1 z^{-1} + h_2 z^{-2}$. In figure 5.5(a) the cost function is evaluated directly on the reverberated test signal, that is, the LP block in figure 5.3 is removed. In figure 5.5(b) the performance landscape using the LP block on the same signal is illustrated. The landscape in 5.5(a) shows that the algorithm will be able to converge to the correct inverse filter by maximizing the kurtosis value of the reverberated test signal. Furthermore the landscapes show that if the direct signal, $\mathbf{s}$, consists of iid samples, the LP residuals should not be used, because this will lead to filter coefficients associated with the maximum kurtosis value that do not correspond to the correct inverse RIR.
Previous work show that a more complex RIR will have several local maxima, e.g. [19]. The performance landscape in 5.5(b) shows that even though the correct "hill" is reached by the algorithm the algorithm will not converge to the correct inverse RIR. For the following tests the LP block is therefore removed.



(a) Performance landscape using the reverberated test signal.



(b) Performance landscape using the reverberated LP residuals.

Figure 5.5: Performance landscape for the kurtosis approach using the 3 tap IIR filter defined in equation (5.12).

In figure 5.5(a) the maximum kurtosis value is $K(\mathbf{Y}) = 1.382$ and it appears for $h_1 = -0.54$, $h_2 = 0.43$. This entails that $K(\mathbf{Y}) > K(\mathbf{S})$, where $K(\mathbf{S}) = 1.133$. However the difference in the kurtosis values is very small and will not affect the convergence of the algorithm heavily. In figure 5.6 the performance landscape of the algorithm is plotted using the INR metric.

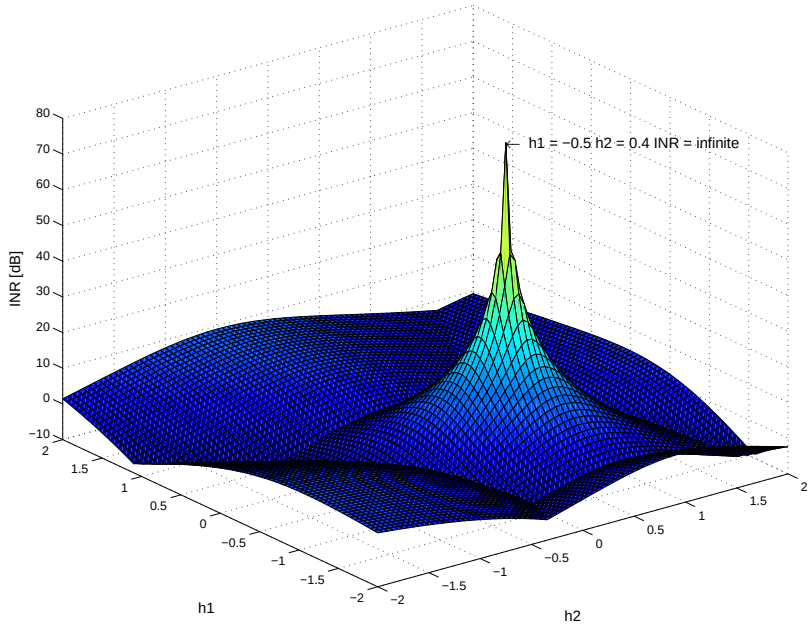


Figure 5.6: The INR performance landscape for the 3 tap IIR. Note that the perfect inverse filter $H_{EQ}(z) = 1 - 0.5z^{-1} + 0.4z^{-2}$ *results in an INR value equal to infinity. For illustration purposes the value has been replaced with* $INR = 10^4 = 80$ *dB.*

The above figure illustrates that there is a direct connection between the performance landscape in figure 5.5(a) and the selected INR performance metric and that the INR metric is very sensitive to the values of the filter coefficients.

Because it now has been established that the performance landscape using kurtosis is valid and that the perfect inverse filter is located in the global maximum (when the LP block is omitted) the algorithm is executed on the 3 tap IIR filter. First the algorithms dependency on the step size is examined simply by measuring the INR and the kurtosis value for several step sizes. The test is made for a filter length $L = 3$ and the results are shown in figure 5.7. The tests have been made for the regular algorithm presented in the previous section and for a modified version, suggested by the project group, where the filter coefficient h_0 is set to one after each iteration. By doing this the algorithm is assisted towards the correct solution, hence better results are obtained. This is also illustrated in figure 5.7 where the modified algorithm achieves an INR value which is 30 dB larger than the regular algorithm. In the rest of the tests presented in this chapter the modified algorithm is therefore used. The modified algorithm will force the first filter coefficient to one, and if the coefficient in the correct inverse RIR is e.g. 0.2 it means that the rest of the filter coefficients in the estimated inverse RIR will be $1/0.2 = 5$ times larger than the correct values. This leads to an amplitude scaling of the output, which is no problem in a Matlab implementation, but in a fixed-point system overflow may occur.

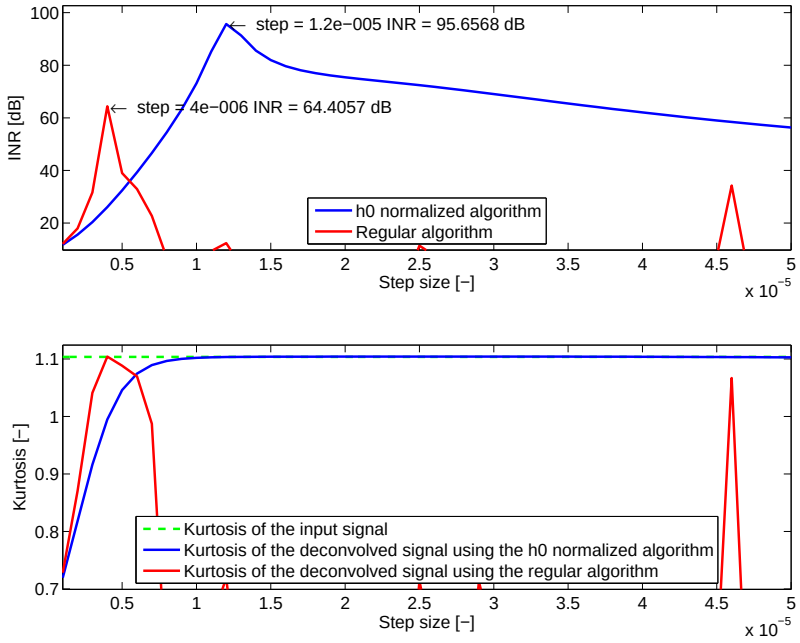


Figure 5.7: The INR and kurtosis metrics as a function of the step size.

The maximum INR is achieved for a step size equal to $1.21 \cdot 10^{-5}$ using the modified algorithm, called h_0 *normalized algorithm*, and therefore this value is used in the following tests. Notice that the algorithm converges to a fairly good result when the step size is larger than $1.21 \cdot 10^{-5}$ as well. Even though the algorithm obtains filter coefficients that are close to the correct solution the INR value drops as compared to the the kurtosis value, which is more or less constant. This difference can be explained by examining the kurtosis and INR performance plots in figure 5.5(a) and 5.6. The latter figure has a large, sharp peak, hence the INR is very sensitive to the filter coefficient, and the former figure has a more wide and flat top which means that the kurtosis value does not change much when the filter coefficients change.
The reason why the regular algorithm obtains less good results is that it has a much slower convergence, because it is not assisted as the modified algorithm is. This is illustrated in figure 5.8 where the two algorithms have been implemented using a step size of $4 \cdot 10^{-6}$ and $1.21 \cdot 10^{-5}$ in the regular and modified algorithms respectively. The figure shows that the modified algorithm converges towards the correct solution and that the regular algorithm does not converge after analysis of the first $100 \cdot 10^3$ samples. As mentioned earlier only the modified algorithm is used in the following tests.

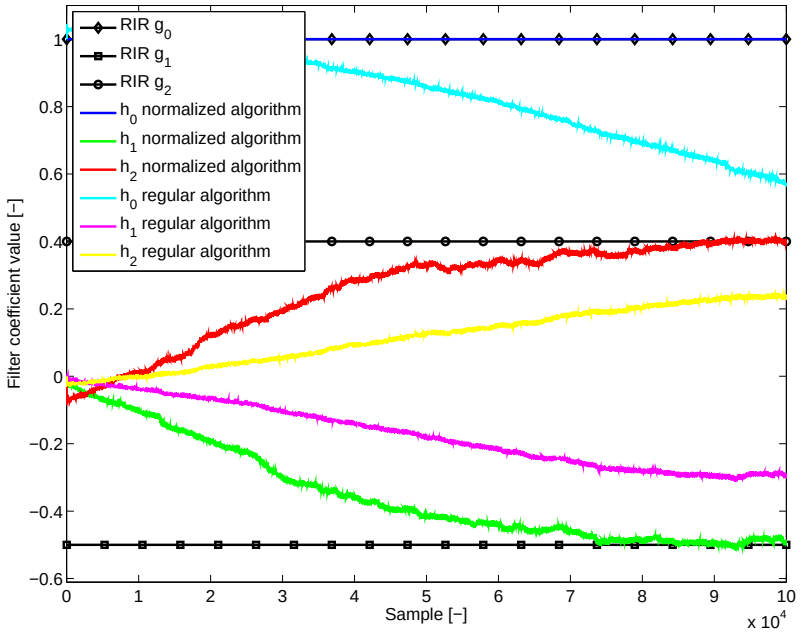


Figure 5.8: The evolution of the three coefficients in the 3 tap FIR equalizer using the regular and the modified kurtosis maximization algorithms. The correct solution is that h_0, h_1, and h_2 equal their corresponding values of the RIR filter.

Figure 5.9 shows the performance landscape in figure 5.5(a) as a contour plot together with 15 of the gradients, which the algorithm calculates during a run. The gradient is defined in equation (5.6). The gradients are stochastic hence a single gradient might have the wrong direction, but the mean of all the gradients for the specific point in the contour plot must aim in the correct direction for the algorithm to work. Therefore the gradients in the plot are an average of gradients in $50 \cdot 10^3$ points. Because of the stochastic gradients the step size is utilized to weigh each gradient, which was also shown in figure 5.4. The figure shows that gradients move towards the minimum which indicates that the algorithm is working properly.

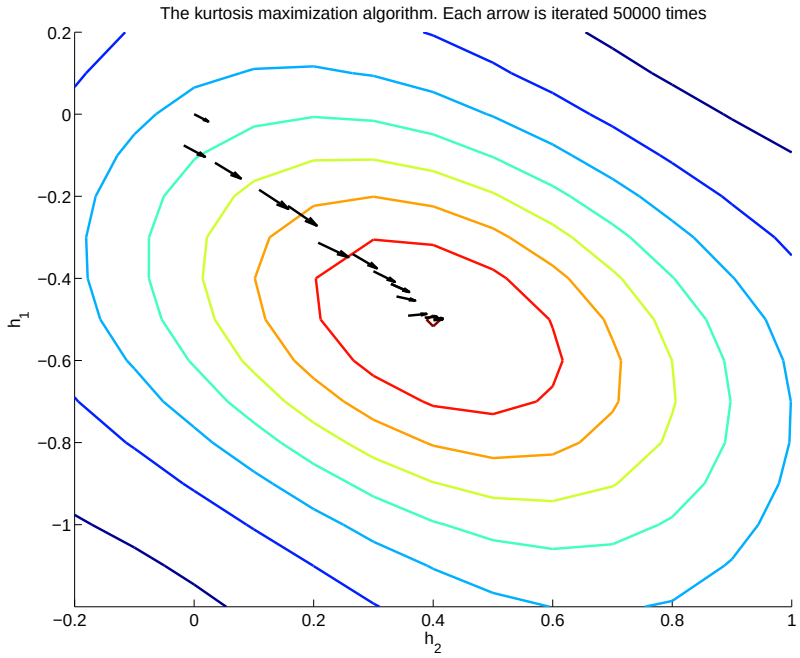


Figure 5.9: A contour plot of the performance landscape and 15 gradients calculated using the kurtosis maximization algorithm. The low kurtosis values are blue and green, and the high kurtosis values are red and brown. The initial value for the filter coefficients h_1 and h_2 is zero.

The purpose of the steepest ascent algorithm is as mentioned earlier to locate the maximum kurtosis, which in the current case is at $h_1 = -0.5$ and $h_2 = 0.4$. Figure 5.9 indicates that the algorithm is able to adjust the filter coefficients towards the inverse RIR, but the figure also illustrates that the algorithm is not capable of making the final adjustments of the coefficients, since the gradients near the maximum drift around the center. If knowledge about the convergence rate existed the step size could be updated as the algorithm approaches convergence. However, this is not the case in blind deconvolution.
The fact that the algorithm can adjust very closely to the correct inverse RIR filter is supported by figure 5.10. The figure illustrates eight convolutions of the RIR and the equalizer filter during the simulation, which was also used to generate figure 5.9.

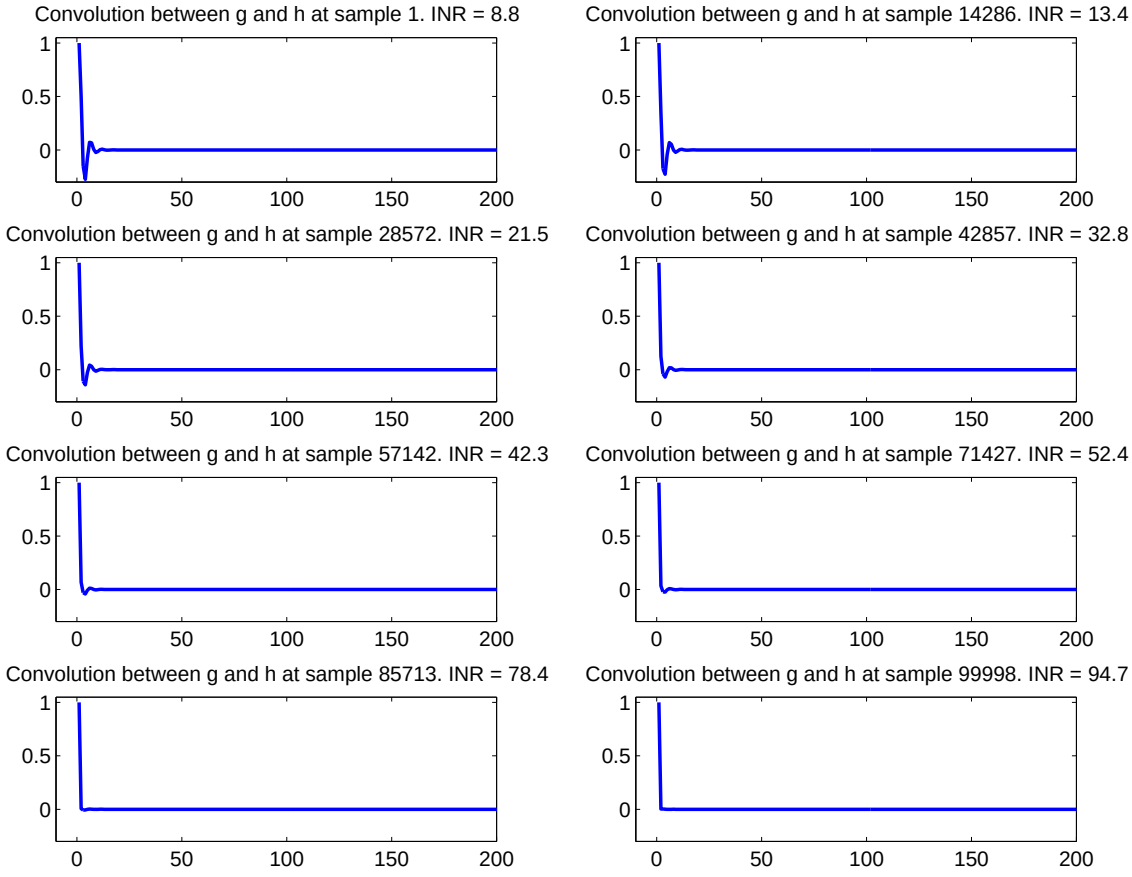


Figure 5.10: Convolution of the RIR g and the equalizer filter h at different points during the simulation of $100 \cdot 10^3$ samples.

Analyzing the figure 5.10 it is clear that the convolution, sample after sample approaches the inverse RIR filter, that is the highest INR. Figure 5.11 shows the final convolution from figure 5.10 and a zoom on the convolution noise, which is in the order of $3 \cdot 10^{-3}$.

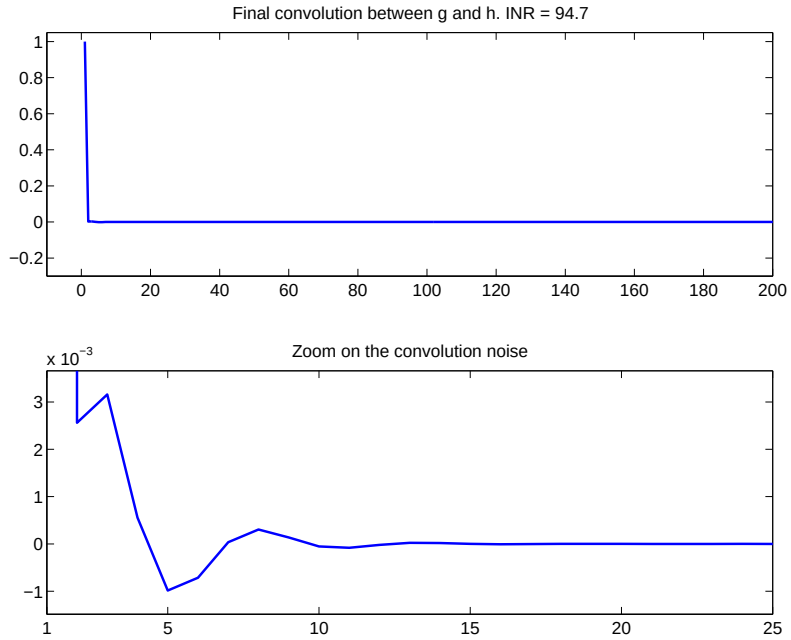


Figure 5.11: Final convolution of the RIR g and the equalizer filter h together with a zoom on the convolution noise.

The fact that the algorithm approaches the correct solution is finally consolidated with figure 5.12 which shows how the filter coefficients change during the simulation. The coefficients moves towards the maximum point, but after that is reached they still fluctuate closely around the correct values.

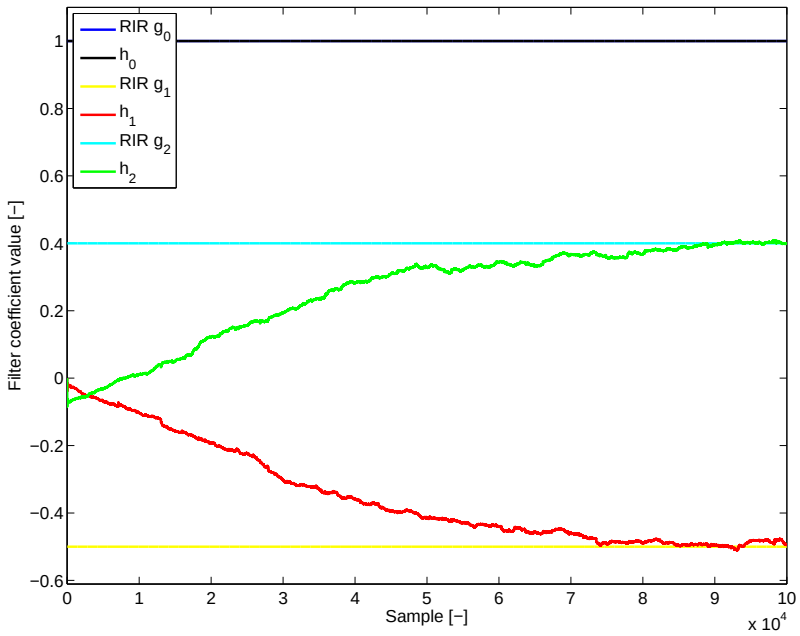


Figure 5.12: The evolution of the three coefficients in the 3 tap FIR equalizer.

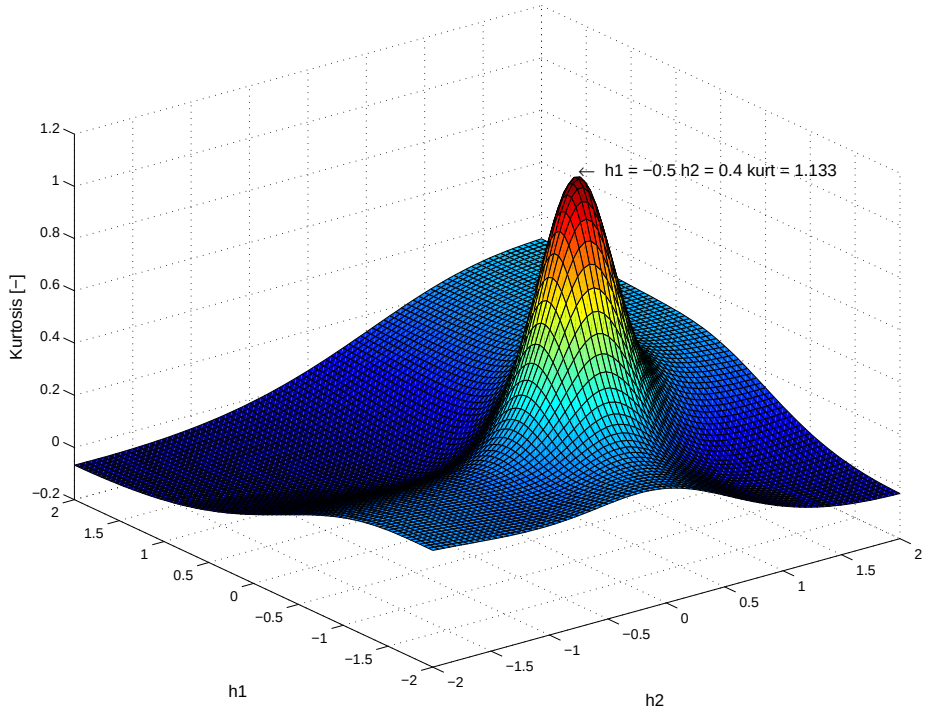
Together figure 5.9, 5.10, and 5.12 illustrate that the kurtosis maxization algorithm is able to adjust towards the correct solution, but that is not able to fine tune the filter coefficients to reach the perfect inverse RIR. The overall performance is however acceptable since the

filter coefficients get quite close to the correct ones and therefore the kurtosis maximization algorithm is applied to more complicated filters.

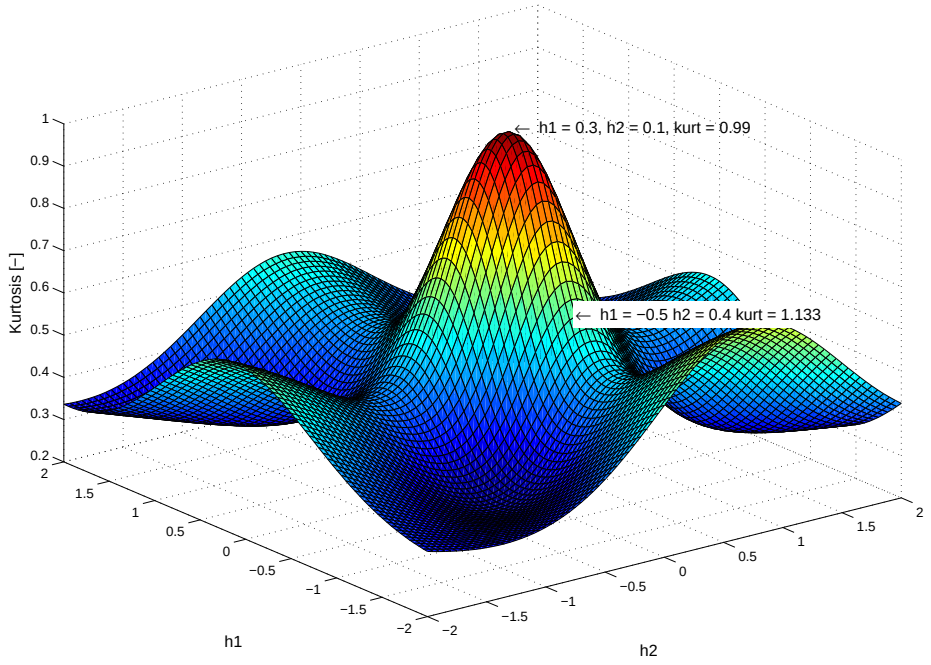
The next RIR, which is used for testing is an infinite impulse response (IIR) filter given by

$$H_{RIR}(z) = \frac{1}{1 - 0.5z^{-11} + 0.4z^{-15}} \tag{5.13}$$

This means that the RIR given by equation (5.13) can be deconvolved with a 16 tap FIR filter. The reasoning behind choosing the 16 tap IIR filter is that it is a little closer to a real RIR as compared to the 3 tap IIR filter in equation 5.12. The delays in the 16 tap filter entail that the coefficients can be considered to add echoes in stead of colourization to the signal, which was the case for the 3 tap IIR filter without delays. In chapter 7 a filter with echoes even further away from the direct speech is tested.
Performance landscapes for the kurtosis approach for this RIR are shown in figure 5.13. In the plots only coefficient h_1 and h_2 in the equalizer filter $H_{EQ}(z) = [1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,h_1\,0\,0\,0\,h_2]$ are changed.



(a) Performance landscape using the reverberated test signal.



(b) Performance landscape using the reverberated LP residuals.

Figure 5.13: Performance landscapes for the kurtosis approach.

The same conclusion as in figure 5.5 is reached, that is when the input signal, $\mathbf{s}$ consists of iid samples the LP block must be omitted from the algorithm. Doing this means that the maximum kurtosis value is located at the perfect inverse RIR.
In figure 5.14 the performance of the algorithm is illustrated using the INR metric. The INR is evaluated as a function of the step size and $\beta = 0.99$ for different filter lengths. The optimal

filter length is expected to be $L = 16$, because the perfect inverse of the RIR, which is an IIR filter, will need to have 16 taps just as the RIR.

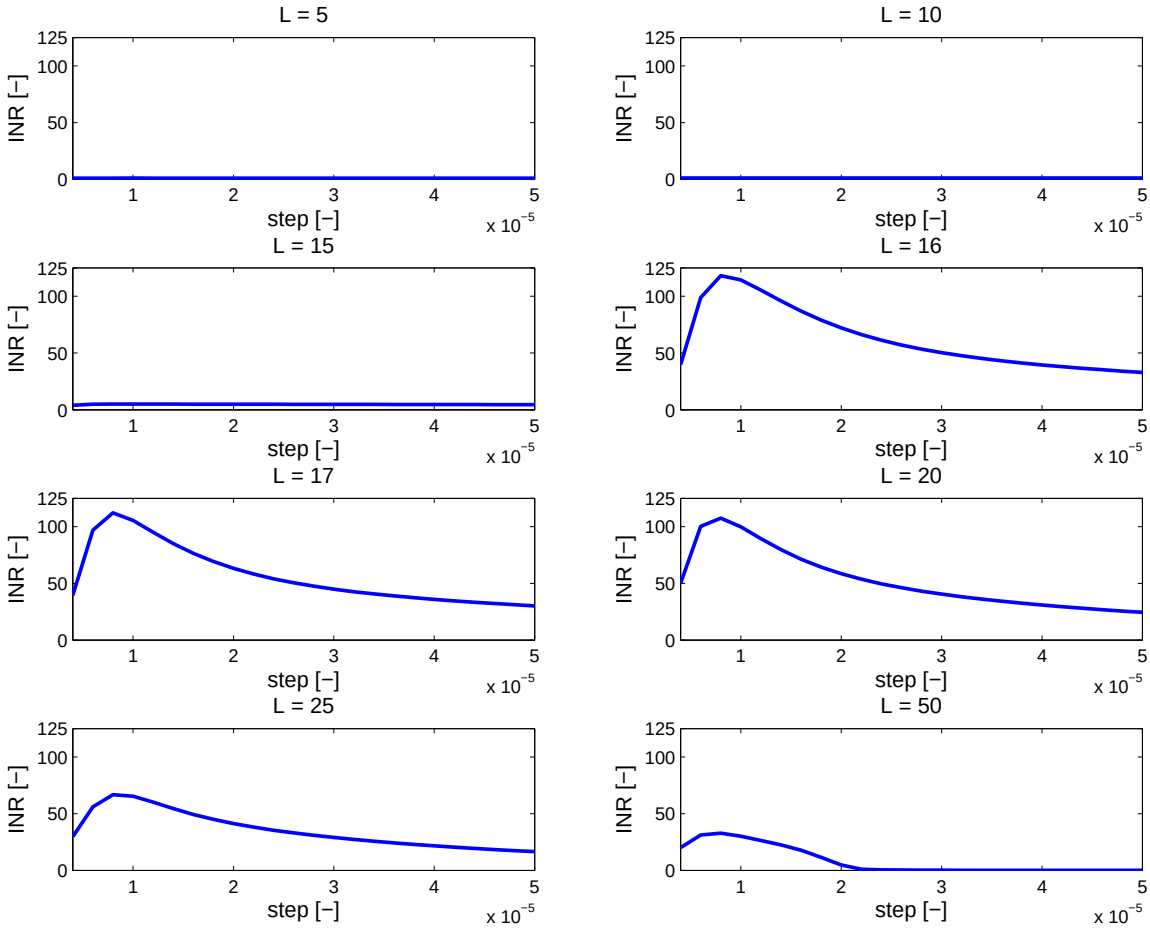


Figure 5.14: Performance of the kurtosis approach used on the RIR in equation (5.13).

It is seen that the performance of the algorithm is sensitive to the step size and that the filter length $L = 17$ is almost as good as the expected and theoretical equalizer filter, which has a filter length $L = 16$. The sensitivity of the step size can be explained by the INR landscape, which is similar to the landscape shown in figure 5.6. The figure illustrates that the INR value changes dramatically, when the filter is close to the correct solution.

Next the algorithm is tested on the all-zero RIR given by

$$H_{RIR}(z) = 1 - 0.5z^{-11} + 0.4z^{-15} \tag{5.14}$$

Because the RIR is a FIR filter the exact inverse filter is an IIR filter. The algorithm only works with FIR filters and therefore it is necessary the estimate the IIR using a FIR. This will lead to worse results than for the opposite situation, since the approximation of the IIR using the FIR adds extra errors to the system. In theory the optimal length of the equalizer is infinite. Note that a real life RIR will be a FIR filter as well. The algorithm has been run with different filter lengths analyzing the dependency on the step size. The results are given in figure 5.15.

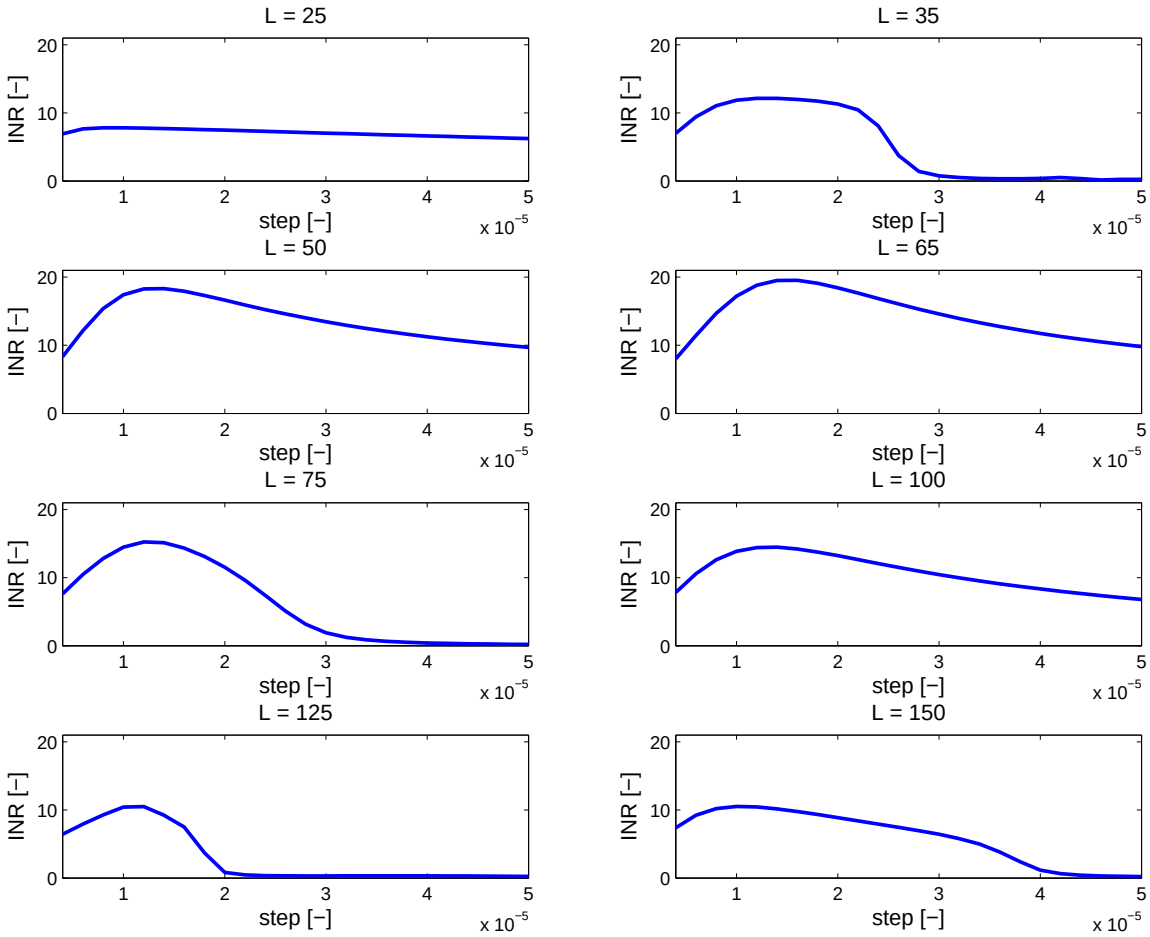


Figure 5.15: Performance of the kurtosis approach used on the RIR in equation (5.14).

As expected the performance is much lower with respect to INR as compared to the previous results. The reason is that the algorithm now have to adjust many more filter coefficients to a more complicated RIR.

5.2 Constrained Maximization

The analysis of unconstrained maximization of the kurtosis of the reverberated LP residuals indicates that approaching the highest kurtosis value does not necessarily lead to the inverse RIR. Constrained maximization of the kurtosis is therefore analysed to evaluate if it is possible to control the maximization. The examined previous work is the work by Jelonnek et al. and Shalvi et al in [20] and [21], respectively. In [20] the following criteria for power and kurtosis is derived such that perfect equalization for the system in figure 4.1 is obtained. The speech signal, $\mathbf{s}$, is assumed to be an iid signal.

$$E[\mathbf{y}] = E[\mathbf{s}] \sum_n |r(n)|^2 \tag{5.15}$$

$$K(\mathbf{y}) = K(\mathbf{s}) \sum_n |r(n)|^2 \tag{5.16}$$

where: $K(\cdot)$ denotes the kurtosis of the signal [-]

The same criteria is used in [21]. Again the idea is to maximize $K(\mathbf{y})$ such that $s(k) = \delta(k-d)$. This leads to the following constrained maximization

$$\begin{aligned} &\max K(\mathbf{y}) \\ &\text{subject to } E[\mathbf{y}] = E[\mathbf{s}] \end{aligned} \tag{5.17}$$

In [20] a steepest ascent algorithm is derived for equation (5.17), where in [21] a closed-form solution based on the eigenvalues is obtained. However the project group questions the use of a constraint on the speech signal $\mathbf{s}$ which is not known. Furthermore the algorithm in [20] is based on the sign of $\mathbf{s}$. Therefore the project group does not suggest the constrained maximization in 5.17 for further analysis. The project group has not come across any other constrained criteria in previous work.

5.3 Summary

In this chapter the kurtosis approach to blind speech dereverberation was analysed. Both constrained and unconstrained approaches were considered, but only the unconstrained approach was by the project group assumed reasonable for blind dereverberation. A steepest ascent algorithm for kurtosis maximization was implemented successfully with a maximum INR value of 94.7 dB for a three tap RIR using $100 \cdot 10^3$ input samples. The LP residuals in the algorithm are not suggested as input to the algorithm if the source signal is iid, because the performance landscape was changed dramatically, when the LP residuals were used. Actually the maximum kurtosis value did not match the correct equalized filter. In general the algorithm experienced difficulties converging to the correct filter. The project group therefore suggested that the first filter coefficient of the equalizing filter is set to one after each iteration. The project group finds this an appropriate solution, because it is just a control of the gain of the direct signal. Furthermore it will constrain the algorithm from producing an entire zero valued output. The fundamental problem for the algorithm is however that it is possible to obtain higher kurtosis values for the processed output as compared to the source signal, that is $K(\mathbf{y}) > K(\mathbf{s})$. This means that even though it can be shown that if $K(\mathbf{y}) = K(\mathbf{s})$, deconvolution has been obtained, then it is possible to exceed the value of $K(\mathbf{s})$, and if $K(\mathbf{y}) >> K(\mathbf{s})$ convergence to the correct solution will not be obtained. This extends the already known problem for the kurtosis approach, that the performance landscape of a complex RIR will suffer from many hills, which means that even though the correct hill is reached by the algorithm, it might not converge as expected.

6

Information Approach

In this chapter an information theoretic approach will be taken to the blind deconvolution problem illustrated in figure 4.1. The idea of using this approach was briefly described in section 4.2.In general there are two approaches when altering the information content of an event (in this context sample). These are shown in figure 6.1.

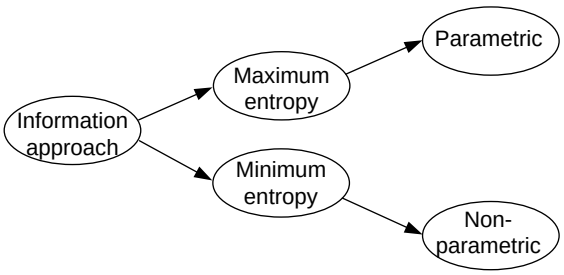


Figure 6.1: The information approaches presented in this chapter.

The method presented in section 6.1 maximizes the entropy because this will remove the redundancy introduced by the room. The second method presented in section 6.3 minimizes the entropy because the entropy metric is maximum when the input is Gaussian. Minimization therefore leads to moving the signal away from being Gaussian, hence removing the room effect. The main difference between the two methods is that the maximum entropy algorithm in its general definition is parametric and the minimum entropy method is nonparametric. Parametric means that a priori knowledge is available and utilized in the given algorithm. Nonparametric means that knowledge about the analysed signal's structure is not available e.g. the pdf of the signal is unknown [22]. An example of a nonparametric measure of a pdf is a histogram. Actually the minimum entropy algorithm only requires the source signal $s(n)$ to be non-Gaussian and white.
The descriptions of the methods are structured as follows. First the idea behind the algorithm and the necessary theory is presented. Then the mathematical formulation of the algorithm is derived and finally the algorithm is implemented and tested. The tests consist of plots of the performance surface and the algorithms dependency on various parameters as compared

to the INR value defined in section 2.3. Furthermore the algorihtms' ability to converge will be examined.

6.1 Maximum Entropy

In chapter 4 the general setup for the blind deconvolution problem was introduced in figure 4.1. As explained by the CLT the RIR, $\mathbf{g}$, caused the reverberated speech, $\mathbf{x}$, to consist of non iid and Gaussian distributed samples. From an information point of view the room output, $\mathbf{x}$, can therefore be viewed as redundant samples and blind deconvolution as redundancy reduction. In redundancy reduction the mutual information, $I(\cdot,\cdot)$, between two samples, denoted $z(i)$ and $z(j)$, must be minimized, $min\ I(z(i), z(j))$, where $\mathbf{z}$ is a random vector. As stated in [9] the mutual information between samples includes cumulants of all orders and not just fourth order cumulants as in the kurtosis maximization approach. The goal in this information approach is therefore to weight these HOS such that statistical independency between samples is obtained.

In [9] Bell et al. present an algorithm for minimizing the mutual information based on the joint entropy, $H(z(i), z(j))$. The joint entropy and the mutual information are related by [9, Eq. 4.1]

$$H(z(i), z(j)) = H(z(i)) + H(z(j)) - I(z(i), z(j)) \tag{6.1}$$

where: $H(z(i))$ is Shannon's entropy of the sample $z(i)$ [-]

Throughout the section Shannon's entropy is just referred to as entropy
From equation (6.1) $I(z(i), z(j))$ is minimized when $H(z(i), z(j))$ is maximized. Bell et al. state that it is not guaranteed that the global minimum of $I(z(i), z(j))$ will be reached because of interference by $H(z(i))$ and $H(z(j))$. The joint entropy is defined by [11, p. 35]

$$H(\mathbf{z}) = -E[\log f_Z(\mathbf{z})] \tag{6.2}$$

where: $f_Z(\mathbf{z})$ is the pdf of $\mathbf{z}$ [-]

When $\mathbf{z}$ is constrained to be finite between the interval $a \leq z(i) \leq b$, $a < b$ for all i, the maximum entropy, $H(\mathbf{z})$, is obtained when the probability of each sample $z(i)$ are all equal, hence the $f_Z(\mathbf{z})$ should correspond to the uniform distribution. This is shown by the following. The index i is ignored for simplicity in the following proof.

$$\begin{aligned} \max\ H(z) &= -\int_a^b f_Z(z) \log f(z) dz \\ \text{subject to } \int_a^b f_Z(z) dz &= 1 \end{aligned} \tag{6.3}$$

The Lagrangian is then

$$\begin{aligned} L(H(z), \lambda) &= -\int_a^b f(z) \log f_Z(z) dz + \lambda \left(\int_a^b f_Z(z) dz - 1 \right) \\ &= \int_a^b \left(-f_Z(z) \log f_Z(z) + \lambda f_Z(z) \right) dz - \lambda \end{aligned} \tag{6.4}$$

The entropy of a continuous random variable is a concave function. Differentiating equation (6.4) with respect to $f_Z(z)$ and setting the result equal to zero the maximum distribution is obtained

$$\frac{\partial}{\partial f_Z(z)} L(H(z), \lambda) = 0 \Leftrightarrow 0 = -1 - \log f_Z(z) + \lambda$$
$$f_Z(z) = e^{\lambda - 1} \tag{6.5}$$

The constraint is then used to determine the Lagrangian multiplier, λ

$$\int_a^b e^{\lambda-1} dz = 1$$
$$e^{\lambda-1} = \frac{1}{b-a} \tag{6.6}$$

The uniform distribution is thereby obtained

$$f_Z(z) = \begin{cases} \frac{1}{b-a} & \text{for} \quad a \leq z \leq b \\ 0 & \text{otherwise} \end{cases} \tag{6.7}$$

and the proof is complete.
An input-output function that ensures an uniform density function on the output is the cumulative density function (cdf), $F_Z(z)$, [23]. This approach is illustrated in figure 6.2.

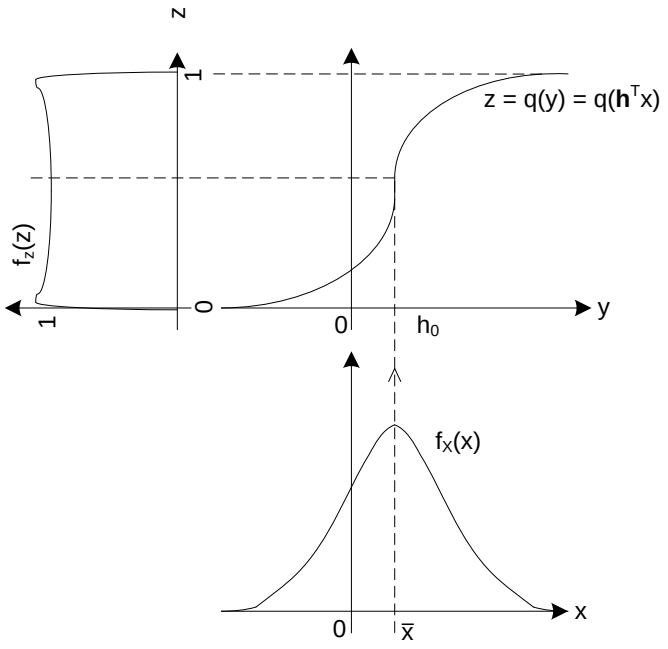


Figure 6.2: The input-output function is approximated to the cdf of the input producing in the optimum case an uniform distribution on the output z. Based on [9, Figure 1].

Because the input-output function is a cdf then $0 \leq \mathbf{z} \leq 1$ and the constraint for maximizing $H(\mathbf{z})$ is thereby maintained. By approximating the corresponding cdf to the input, each input sample of $\mathbf{x}$ is proportional to its expected probability producing an uniform distribution on the output in the optimum case [23]. This approach introduces the block diagram in figure 6.3.

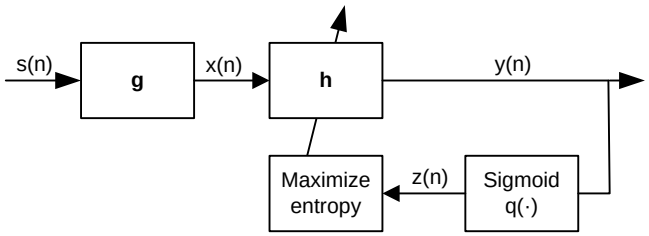


Figure 6.3: The maximum entropy principle applied to a blind deconvolution problem. Based on [10, Figure 1a].

In figure 6.3 the approximated cdf is denoted "sigmoid", $q(y(n))$, that is $q(y(n))$ is a monotone increasing, continuous function, satisfying $\lim_{y(n)\to-\infty} = 0$ and $\lim_{y(n)\to\infty} = 1$ for all n, [24]. The procedure for the algorithm presented in the above block diagram is as follows: first choose a sigmoid $q(\mathbf{y})$ that in the ideal case is the cdf of the speech $\mathbf{s}$, then update the filter coefficients of $\mathbf{h}$ according to this sigmoid. If the algorithm converges to the correct filter $\mathbf{h}$, that is $g(n) * h(n) = \delta(n-d)$, an uniform distribution will be obtained for $\mathbf{z}$. Note that the dereverberated speech is $\mathbf{y}$ and not $\mathbf{z}$. In this algorithm $\mathbf{z}$ is just a part of the cost function J used to update the equalizer filter $\mathbf{h}$. The algorithm for updating $\mathbf{h}$ is derived next.

Let $\mathbf{x} = [x(1)\ x(2)\ ...\ x(M)]^T$, $\mathbf{y} = [y(1)\ y(2)\ ...\ y(M)]^T$, and the filter coefficients $\mathbf{h} = [h_0\ h_1\ ...\ h_{\mathrm{L}-1}]^T$ be defined by the $M \times M$ convolution matrix $\mathbf{H}$, where $M \geq L$, such that the output of the algorithm is given by

$$\begin{bmatrix} y(1) \\ \vdots \\ y(M) \end{bmatrix} = \begin{bmatrix} h_0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ h_1 & h_0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ h_{\mathrm{L}-1} & h_{\mathrm{L}-2} & \cdots & h_0 & 0 & \cdots & 0 \\ 0 & h_{\mathrm{L}-1} & h_{\mathrm{L}-2} & h_1 & h_0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & h_{\mathrm{L}-1} & h_{\mathrm{L}-2} & \cdots & h_0 \end{bmatrix} \begin{bmatrix} x(1) \\ \vdots \\ x(M) \end{bmatrix} = \mathbf{Hx} \tag{6.8}$$

The relationship $\mathbf{z} = q(\mathbf{y}) = q(\mathbf{Hx})$ was illustrated in figure 6.2. The pdfs $f_Z(\mathbf{z})$ and $f_X(\mathbf{x})$ are in [25, p. 183] related by

$$f_Z(\mathbf{z}) = \frac{f_X(\mathbf{x})}{\mathrm{det}\mathbf{J}(\mathbf{x})} \tag{6.9}$$

Where the jacobian, $\mathbf{J}(\mathbf{x})$, is an $M \times M$ matrix given by

$$\mathbf{J}(\mathbf{x}) = \frac{\partial z(i)}{\partial x(j)} = \begin{bmatrix} \frac{\partial z(1)}{\partial x(1)} & \frac{\partial z(1)}{\partial x(2)} & \cdots & \frac{\partial z(1)}{\partial x(M)} \\ \vdots & \vdots & & \vdots \\ \frac{\partial z(M)}{\partial x(1)} & \frac{\partial z(M)}{\partial x(2)} & \cdots & \frac{\partial z(M)}{\partial x(M)} \end{bmatrix}, \begin{matrix} 1 \leq i \leq M \\ 1 \leq j \leq M \end{matrix} \tag{6.10}$$

From equation 6.9 the entropy $H(\mathbf{z})$, defined in equation 6.2, which should be maximized, can be written as

$$\begin{aligned} -\log f_Z(\mathbf{z}) &= -\log f_X(\mathbf{x}) + \log|\mathrm{det}\mathbf{J}(\mathbf{x})| \\ H(\mathbf{z}) &= H(\mathbf{x}) + E[\log|\mathrm{det}\mathbf{J}(\mathbf{x})|] \end{aligned} \tag{6.11}$$

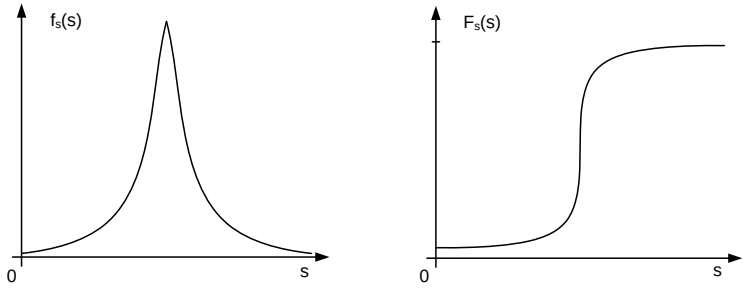
$H(\mathbf{x})$ is not affected by the filter matrix $\mathbf{H}$, so maximizing $H(\mathbf{z})$ corresponds to maximizing $\det\mathbf{J}(\mathbf{x})$. For doing so it is convenient to rewrite $\mathbf{J}(\mathbf{x})$ using the chain rule $\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \cdot \frac{\partial y}{\partial x}$

$$\begin{aligned}
\mathbf{J}(\mathbf{x}) &= \begin{bmatrix} \frac{\partial z(1)}{\partial y(1)} \cdot \frac{\partial y(1)}{\partial x(1)} & \frac{\partial z(1)}{\partial y(1)} \cdot \frac{\partial y(1)}{\partial x(2)} & \cdots & \frac{\partial z(1)}{\partial y(1)} \cdot \frac{\partial y(1)}{\partial x(M)} \\ \vdots & \vdots & & \vdots \\ \frac{\partial z(M)}{\partial y(M)} \cdot \frac{\partial y(M)}{\partial x(1)} & \frac{\partial z(M)}{\partial y(M)} \cdot \frac{\partial y(M)}{\partial x(2)} & \cdots & \frac{\partial z(M)}{\partial y(M)} \cdot \frac{\partial y(M)}{\partial x(M)} \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial y(1)}{\partial x(1)} & \frac{\partial y(1)}{\partial x(2)} & \cdots & \frac{\partial y(1)}{\partial x(M)} \\ \vdots & \vdots & & \vdots \\ \frac{\partial y(M)}{\partial x(1)} & \frac{\partial y(M)}{\partial x(2)} & \cdots & \frac{\partial y(M)}{\partial x(M)} \end{bmatrix} \odot \begin{bmatrix} \frac{\partial z(1)}{\partial y(1)} \\ \vdots \\ \frac{\partial z(M)}{\partial y(M)} \end{bmatrix} \\
&= \mathbf{H} \odot \begin{bmatrix} \frac{\partial z(i)}{\partial y(i)} \end{bmatrix}
\end{aligned} \tag{6.12}$$

Where $\odot$ denotes that the vector $\begin{bmatrix} \frac{\partial z(i)}{\partial y(i)} \end{bmatrix}$ is multiplied element-wise with each column in $\mathbf{H}$. Noting that the convolution matrix $\mathbf{H}$ is a lower triangular matrix the determinant of $\mathbf{J}(\mathbf{x})$ is given by

$$\begin{aligned}
\det\mathbf{J}(\mathbf{x}) &= \det\mathbf{H} \cdot \prod_i^M \frac{\partial z(i)}{\partial y(i)} \\
&= h_0^M \cdot \prod_i^M \frac{\partial z(i)}{\partial y(i)}
\end{aligned} \tag{6.13}$$

Before maximizing the determinant $\det\mathbf{J}(x)$ with respect to the filter coefficients $\mathbf{h}$, the sigmoid $q(\mathbf{y})$ is chosen. As stated previously in this section the sigmoid is in the ideal case equal to the cdf of the speech $\mathbf{s}$. This means that the sigmoid is chosen based on a priori knowledge of the input $\mathbf{s}$, and hence not estimated via the reverberated speech $\mathbf{x}$. The fact that the sigmoid is fixed when running the algorithm, entails that the performance of the algorithm relies much on this a priori knowledge of $\mathbf{s}$. In figure 6.4 a typical pdf, $f_S(s)$, and cdf, $F_S(s)$ of speech is shown. The pdf and cdf are for a single random variable s.



(a) A typical pdf for speech. (b) A typical cdf for speech.

Figure 6.4: Typical pdf and cdf for speech, based on[9, fig. 2(b)].

The hyperbolic tangent function, $\tanh(\cdot)$, is suggested in [9] for approximating the cdf in figure 6.4(b). However $\mathbf{z} = \tanh(\mathbf{y})$ is bounded for the interval $-1 \leq z(i) \leq 1$. The project group therefore suggests $\mathbf{z} = 0.5(1 + \tanh(\mathbf{y}))$, for further analysis. This entails that $0 \leq z(i) \leq 1$ which corresponds to the requirement of a cdf. Using this choice for the sigmoid, the

update rule for the filter $\mathbf{h}$ is derived. Taking the logarithm to equation (6.13) and letting $\mathbf{z} = 0.5(1 + \tanh(\mathbf{y}))$, equation (6.13) becomes

$$\begin{aligned}\log|\det\mathbf{J}(\mathbf{x})| &= M \cdot \log h_0 + \sum_{i=1}^{M} \log\left|\frac{0.5}{\cosh^2(y(i))}\right| \\ &= M \cdot \log h_0 + \sum_{i=1}^{M} \log 0.5 - 2\sum_{i=1}^{M} \log|\cosh(y(i))| \end{aligned} \tag{6.14}$$

First the update of the leading term in $\mathbf{h}$ that is h_0 is derived. This variable has the objective to align the sigmoid with the input pdf of $\mathbf{x}$ [9], see the dashed line in figure 6.2. Differentiating equation (6.14) with respect to h_0 the update becomes

$$\begin{aligned}\Delta h_0 &= M \cdot \frac{1}{h_0} - 2\sum_{i=1}^{M} \frac{1}{\cosh(y(i))} \cdot \sinh(y(i)) \cdot x(i) \\ &= M \cdot \frac{1}{h_0} - 2\sum_{i=1}^{M} \tanh(y(i)) \cdot x(i) \end{aligned} \tag{6.15}$$

For the remaining terms in h, that is $h_{\mathrm{j}+1}$, $j = \{0, ..., L-2\}$, the update rule is

$$\Delta h_{\mathrm{j}+1} = -2\sum_{i=j+1}^{M} \tanh(y(i)) \cdot x(i-j) \tag{6.16}$$

Equation (6.15) and (6.16) are implemented as a steepest ascent algorithm. The general form of the steepest ascent was introduced for the kurtosis maximization approach in equation (5.7).

Evaluation of Sigmoids

The choice of $y = 0.5 \cdot (1 + \tanh(\cdot))$ as a good estimator for the cdf of the desired speech by [9] is evaluated next. In figure 6.5 an example of a short speech sequence is shown together with the histogram approximating the pdf of the sequence. The speech was recorded by Bang & Olufsen, [26], in an anechoic chamber as described in appendix F. The bottom figure shows the approximated cdf of the speech (using the Matlab function `ecdf()`) and the chosen sigmoid $y = 0.5 \cdot (1 + \tanh(\cdot))$. It is seen that the sigmoid is not a good estimator for the cdf of the speech. The slope of the cdf for the speech is much steeper, than for the sigmoid, around zero.

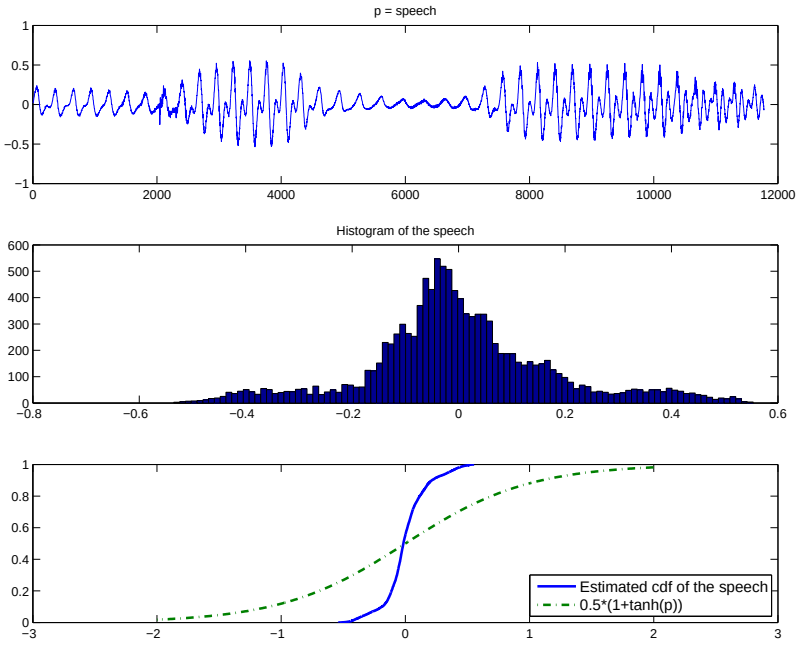


Figure 6.5: An example of a speech sequence. The histogram of the speech approximates the pdf of the speech. The bottom figure shows the approximated cdf of the speech together with the chosen sigmoid $y = 0.5 \cdot (1 + \tanh(\cdot))$.

Figure 6.6 shows the same as figure 6.5 but for a zero-mean, unit variance Gaussian distributed sequence. It is clear that the sigmoid is a much better estimator of the cdf for a Gaussian sequence than for a speech sequence.

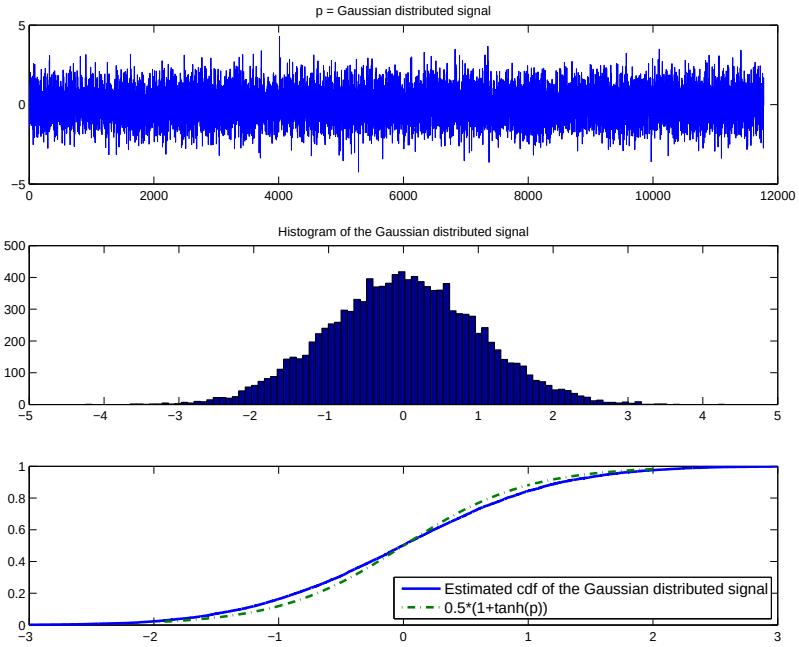
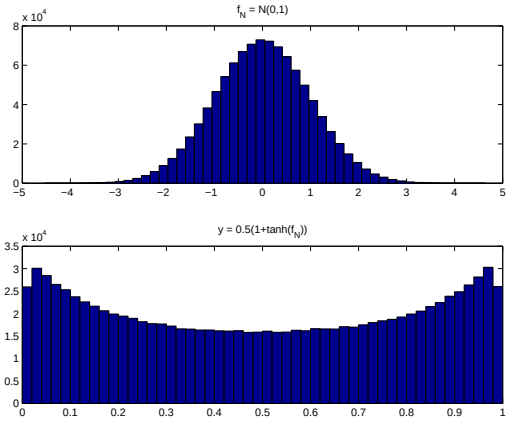


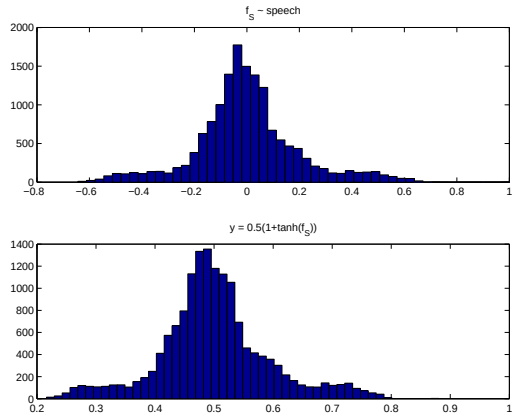
Figure 6.6: A Gaussian distributed sequence with the matching histogram. The bottom figure shows the approximated cdf of the Gaussian distributed sequence together with the chosen sigmoid $y = 0.5 \cdot (1 + \tanh(\cdot))$.

This observation is further illustrated in figure 6.7. Figure 6.7(a) and 6.7(b) show histograms

of the input and output of the sigmoid for a zero-mean and unit variance Gaussian distributed sequence and speech sequence, respectively. The same speech sequence as in figure 6.5 is used. As previously stated, if the sigmoid equals the cdf of the input an uniform distribution will be obtained for the output of the sigmoid. Figure 6.7(b) therefore shows that $y = 0.5 \cdot (1 + \tanh(\cdot))$ is not a good estimator for the cdf of the speech sequence. However, an approximately uniform distribution is obtained for the Gaussian distribution on the output of the sigmoid. This entails that by choosing this sigmoid the algorithm will converge closely to a Gaussian distribution and not a non-Gaussian distribution as desired. Therefore the project group does not suggest the sigmoid $y = 0.5 \cdot (1 + \tanh(\cdot))$ as an appropriate estimator for the cdf of the speech.



(a) A Gaussian distributed sequence and histogram of the output of the sigmoid $y = 0.5 \cdot (1 + \tanh(\cdot))$.



(b) Speech sequence and histogram of the output of the sigmoid $y = 0.5 \cdot (1 + \tanh(\cdot))$.

Figure 6.7: The effect of $y = 0.5 \cdot (1 + \tanh(\cdot))$ as sigmoid.

The sigmoid mismatching with the desired cdf is an important research area for further development. Bell et al. suggest the flexible logistic differential equation, $\partial z / \partial y = z^p (1 - z)^r$, where real numbers p and r are chosen according to the a priori knowledge of the source. In the case of speech Bell et al. suggest $p = r = 5$. However the logistic differential equation must be solved by numerical integration followed by a parametric estimation of the sigmoid $\mathbf{y}$. Bell et al. do not pursuit this approach any further, and experimental results of the algorithm are only reported for the equations (6.15) and (6.16).

To overcome the cdf mismatch a simple and flexible sigmoid, that can easily be adjusted to the a priori knowledge of the source is needed. Furthermore a sigmoid that leads to a simple update of the filter coefficients in $\mathbf{h}$ is desired. Figure 6.5 shows that the slope of $\tanh(\cdot)$ around the mean value of the speech is too small compared to the slope of the estimated cdf of the speech. It will therefore be desirable to adjust the slope of the sigmoid to the desired cdf. The project group therefore suggests that this is done by multiplying the argument of $\tanh(\cdot)$ with a constant a, $a \in R$. This sigmoid proposal also has the desired advantage of a simple update of $\mathbf{h}$. The sigmoid $\mathbf{z}$, proposed by the project group, and its matching pdf $\frac{\partial \mathbf{z}}{\partial y}$ then becomes

$$\mathbf{z} = 0.5(1 + \tanh(a \cdot \mathbf{y})) \tag{6.17}$$

$$\frac{\partial \mathbf{z}}{\partial y} = \frac{a}{2\cosh^2(a \cdot \mathbf{y})} \tag{6.18}$$

It can easily be shown that $\int_{-\infty}^{\infty} \frac{a}{2\cosh(a\mathbf{y})} = 1$ for $a \in R$ which meets the requirements of a pdf. In figure 6.8 the proposed sigmoid with $a = 5$ is compared to the estimated pdf of the speech

sequence, the sigmoid for $a = 1$, and the zero-mean, unit variance Gaussian distribution. The estimated cdf is for the same speech sequence as before.

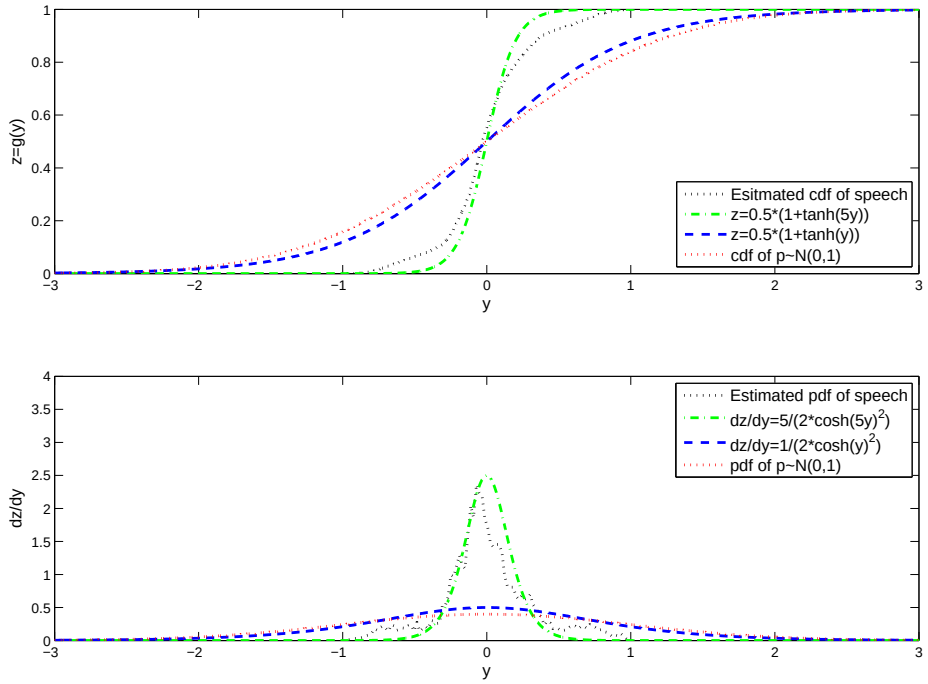


Figure 6.8: Comparison of the proposed sigmoids.

From the figure it is seen that the proposed sigmoid closely fits the estimated cdf of the speech. The matching pdf also obtains a much higher peakedness (high kurtosis) as compared to the Gaussian and regularly $\tanh(\cdot)$, hence it is super-Gaussian, and fits the estimated pdf of the speech. The estimated pdf is based on the normalized histogram of the speech. Based on this, the flexible sigmoid in equation 6.17 meets the requirements and is used in the further development of the algorithm. The update of $\mathbf{h}$ then becomes

$$\Delta h_1 = M \cdot \frac{1}{h_1} - 2a \sum_{i=1}^{M} \tanh(a \cdot y(i)) \cdot x(i) \tag{6.19}$$

$$\Delta h_{j+1} = -2a \sum_{i=j+1}^{M} \tanh(a \cdot y(i)) \cdot x(i-j) \tag{6.20}$$

Experimental Results

The maximum entropy approach is implemented using equation (6.19) and (6.20) and as in the case with the kurtosis approach, described in section 5.1, tested for its ability to deconvolve all-pole RIR and all-zero RIR filters. The performance landscape of the cost function will also be analysed. The Matlab implementation is located on the CD in the directory `Algorithms/InfoMax.m`.

It is important to test the algorithm with a known source signal $\mathbf{s}$ which has been sampled from a known distribution. This entails that the used sigmoid has to correspond with the cdf of $\mathbf{s}$. In the evaluation of sigmoids, the project group suggested the sigmoid $\mathbf{z} = 0.5(1 + \tanh(a\mathbf{y}))$

in equation (6.17), where a could be adjusted according to the source signal. The source signal should therefore be distributed according to the pdf given by (6.18)

$$f_S(\mathbf{s}) = \frac{a}{2\cosh^2(a \cdot \mathbf{s})} \tag{6.21}$$

In figure 6.8 the value $a = 5$ seems like a reasonable choice for speech, and is therefore chosen when generating a test sequence to make sure that the test sequence approximates the pdf of speech. In order to be able to sample from the distribution in equation (6.21) an accept/reject algorithm has been implemented. The algorithm is described in appendix B and located at the enclosed CD in the directory `Algorithms/rejectsample.m`. In figure 6.9 the distribution is shown in the uppermost plot. The histogram of the output of the cdf closely approximates the uniform distribution which indicates that the sampling is in accordance with the desired pdf. For illustration purposes the pdf and corresponding cdf have been compared to the estimated pdf and cdf of a zero-mean, unit variance Gaussian distribution. The distributions diverges which indicates that the output of the algorithm will not move towards a Gaussian distributed sequence. As mentioned earlier this means that the room effect is removed from the received signal.

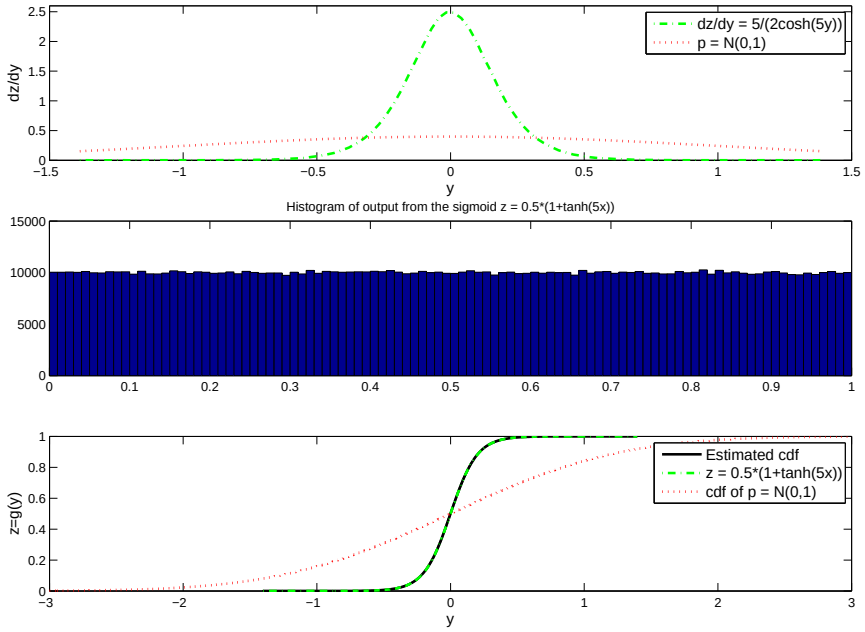


Figure 6.9: The estimated pdf and cdf of the test sequence compared to a Gaussian distribution.

First the algorithms ability to converge is tested. As in the kurtosis approach the convergence is analysed on a simple RIR that is a 3 tap all-pole filter with the coefficients [1 -0.5 0.4]. The filter is defined in equation (5.12). It is thereby possible to make a perfect inverse deconvolution using a 3 tap FIR filter, that is $L = 3$. The sequence $\mathbf{x}$ (room output) consists of a total of $30 \cdot 10^3$ samples and $M = 100$ samples. The sigmoid is used with $a = 5$ and an initial filter equal to $\mathbf{h}$ =[1 0 0] is used. Furthermore the modification, where the filter coefficient h_0 is set to one, is also utilized. This idea was presented during the tests of the kurtosis maximization algorithm in section 5.1. The project group has performed the same tests as in that section and obtained results similar to the ones shown in figure 5.7 and 5.8. Therefore the modification is also applied to the maximum entropy algorithm.
In order to analyse the ability of the algorithm to converge to the correct filter a performance landscape is made. The performance landscape is a histogram based on the output of the

sigmoid, z. The entropy is then calculated for the filter coefficients $-2 \leq h_i \leq 2$, $i = \{1, 2\}$ in the equalizer filter. Figure 6.10 shows the performance landscape with the coefficient h_1 that is the filter is set to [1 h_1 0.4].

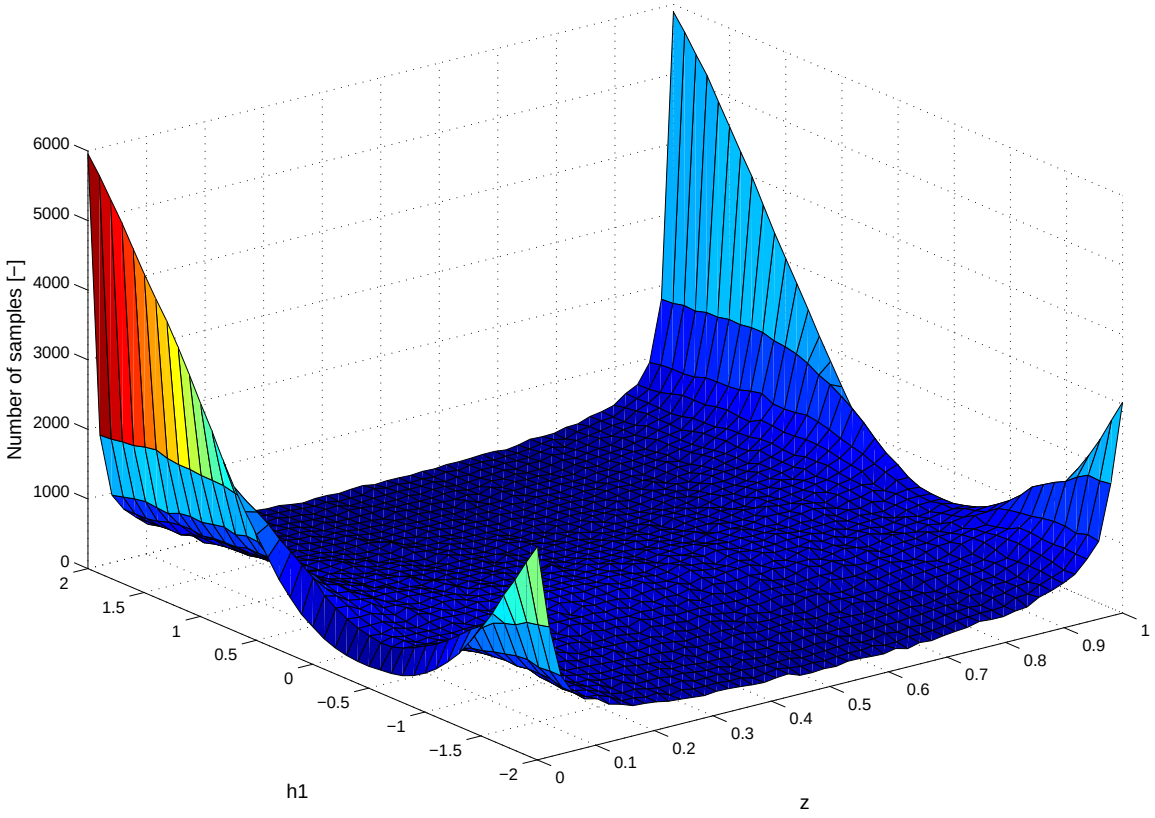


Figure 6.10: Performance plot for the distribution of z, where h_1 is varied.

As expected an almost uniform histogram is obtained for $h_1 = -0.5$. For $h_1 < -1$ and $h_1 > 0$ the number of samples, with values close to $z = 0$ and $z = 1$, are significantly increased which emphasizes that the sigmoid does not match the cdf of the samples. The performance landscape also shows that the algorithm could have difficulties to converge to the correct equalizing filter because the histogram is close to uniform in the range of $-1 \leq h_1 \leq 0$. However it will later be shown that this is not the case. It should be mentioned that the performance landscape cannot be illustrated as a function of both h_1 and h_2, which would change the landscape because it would require a four dimensional plot.

Figure 6.11 illustrates the INR performance surface for the algorithm, when using the 3 tap IIR filter. The conclusion is equal to the one stated in the kurtosis approach namely that the perfect inverse results in the largest INR value. In addition the figure shows that the INR value is very sensitive to the filter coefficients hence it is a good metric, when evaluating the equalizer filter. The figure furthermore indicates that the step size of the algorithm is very critical because if the step size is too big the filter coefficients cannot be adjusted properly, which again will lead to a significantly decreased INR.

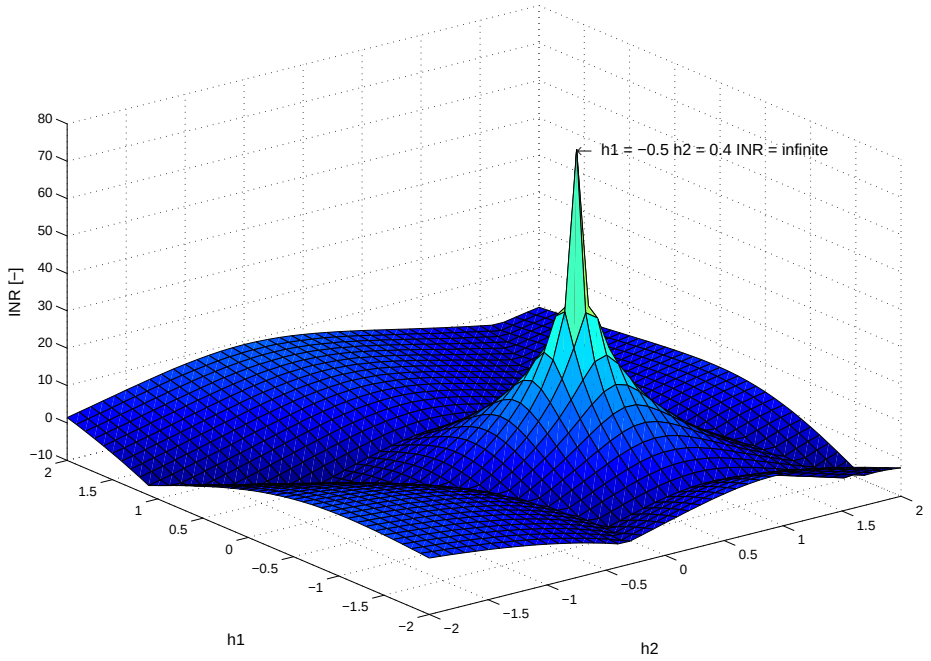


Figure 6.11: INR performance surface for the maximum entropy algorithm equalizing the 3 tap IIR filter. Notice that the perfect inverse is set equal to 80 dB for illustration purposes. The real value is equal to infinity.

Now that the regular and the INR performance surfaces have been evaluated and it has been concluded that the performance surface will lead to the correct inverse RIR, the algorithm is applied to a reverberated signal. First however the maximum entropy algorithm is tested with regards to its sensitivity to the step size. An initiating test showed that a step size in the range of $10^{-6} \leq step \leq 10^{-5}$ gained the best results with regards to INR. In figure 6.12 the INR is calculated as a function of the step size in the previously mentioned step size region.

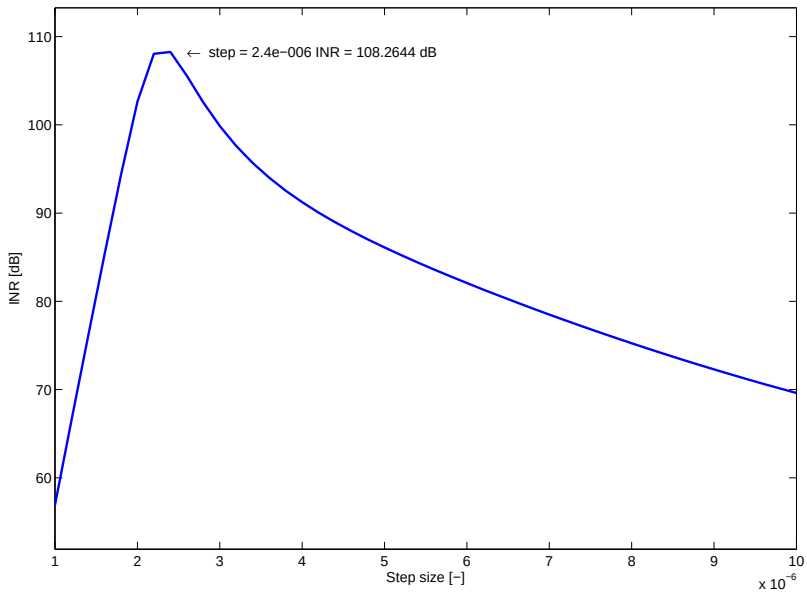


Figure 6.12: INR as a function of the step size.

The test shows that a 50 dB change in the INR is obtained within the plotted step size range.

This emphasizes that the algorithm is very sensitive to the change in step size. The maximum INR is 108.26 dB for a step size of $2.4 \cdot 10^{-6}$. It is assumed that the step is a reasonable size for other filters as well and therefore this step size is used and hold fixed for the following tests.
The step size is very dependent on the input length in the sense that a small step size requires more samples because it makes the algorithm converge slower. If the algorithm converges with a smaller step size it will also lead to better results because the algorithm can adjust the filter coefficients more accurately. For comparison reasons, described in chapter 7, the maximum entropy algorithm has been tested on the same 3 tap IIR RIR as above, but this time $100 \cdot 10^3$ samples were used. The optimal step size was determined to be $1 \cdot 10^{-6}$, and it resulted in a maximum INR value equal to 111.71 dB. This result confirms the theory that a larger number of samples combined with a smaller step size leads to a better filter hence a better INR value.

Next the algorithm is used to equalize the 3 tap IIR filter with the step size equal to $2.4 \cdot 10^{-6}$. The other parameters in the algorithm are set to the previously mentioned values. Figure 6.13 shows the convolution of the RIR and the equalizer filter at the beginning and end of the sample sequence and at six intermediate points. Analyzing the figure it is clear that the algorithm converges towards the correct solution since the subplots approach a Kronecker delta function and because the INR continually increases.

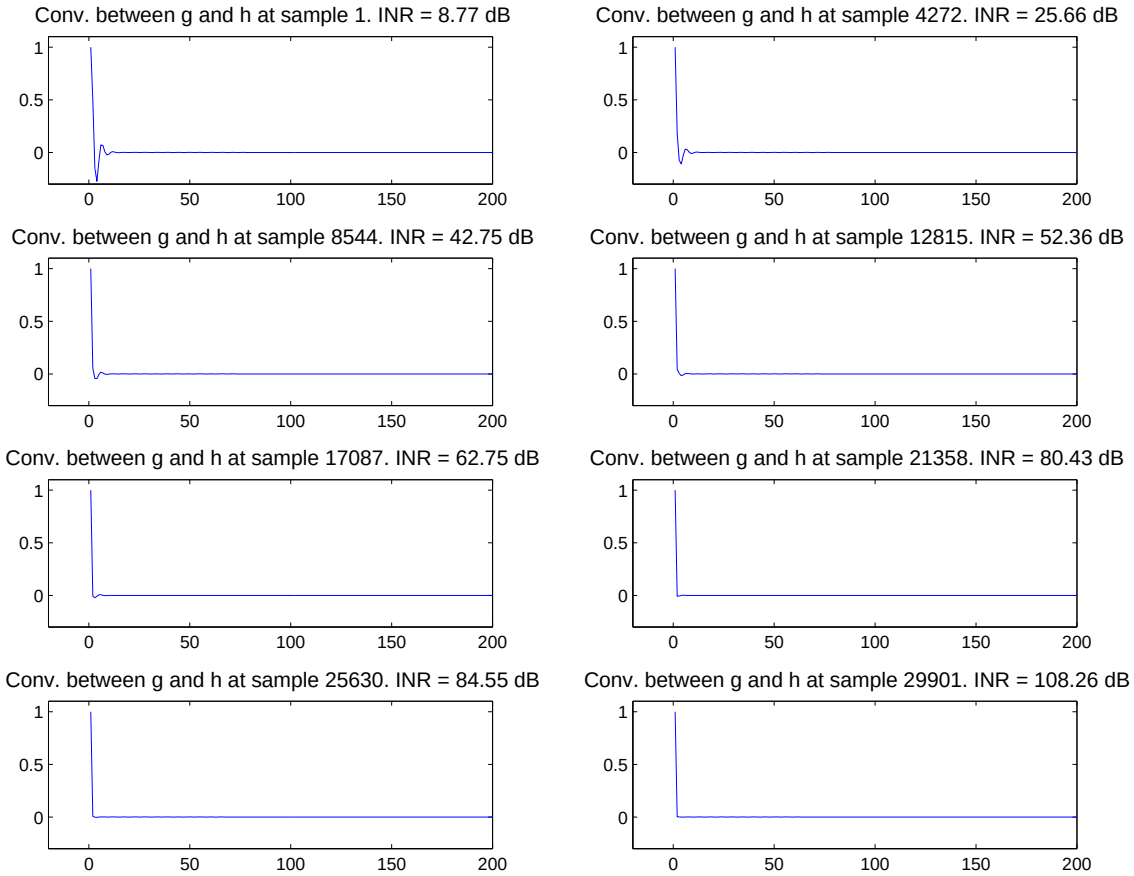


Figure 6.13: Convolution of the RIR g and the equalizer filter h. In the title of each subplot the INR value for the current convolution is given together with the current sample number.

Figure 6.14 shows the final convolution from figure 6.13 and a zoom on the convolution noise of that final convolution.

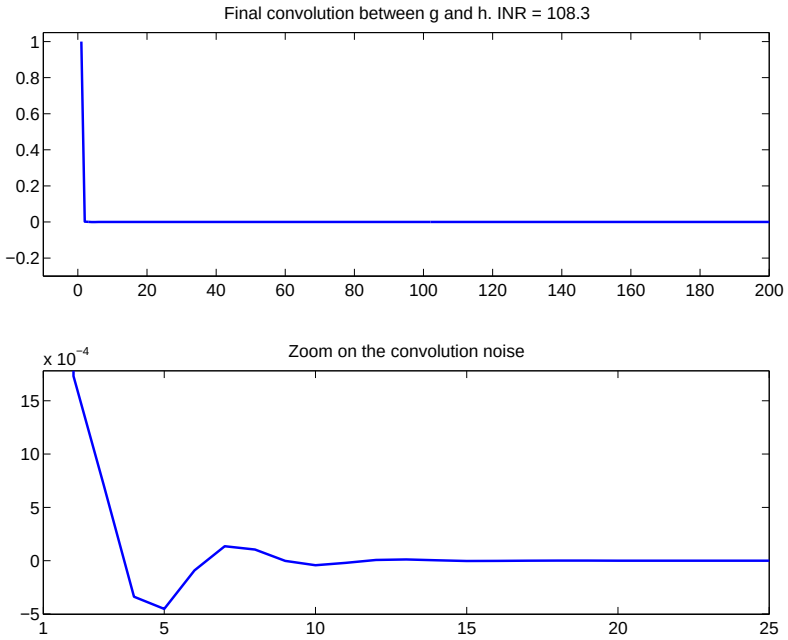


Figure 6.14: Final convolution of the RIR g and the equalizer filter h. The lower subplot is a zoom on the convolution noise in the upper subplot.

Figure 6.15 substantiates the conclusion that the maximum entropy algorithm converges to the correct inverse filter. The figure shows how the three filter coefficients in the equalizer filter approach the values of the filter coefficients in the RIR, which is the desired behaviour of the algorithm. It is therefore concluded that the algorithm is able to converge to the correct solution for a simple 3 tap IIR RIR.

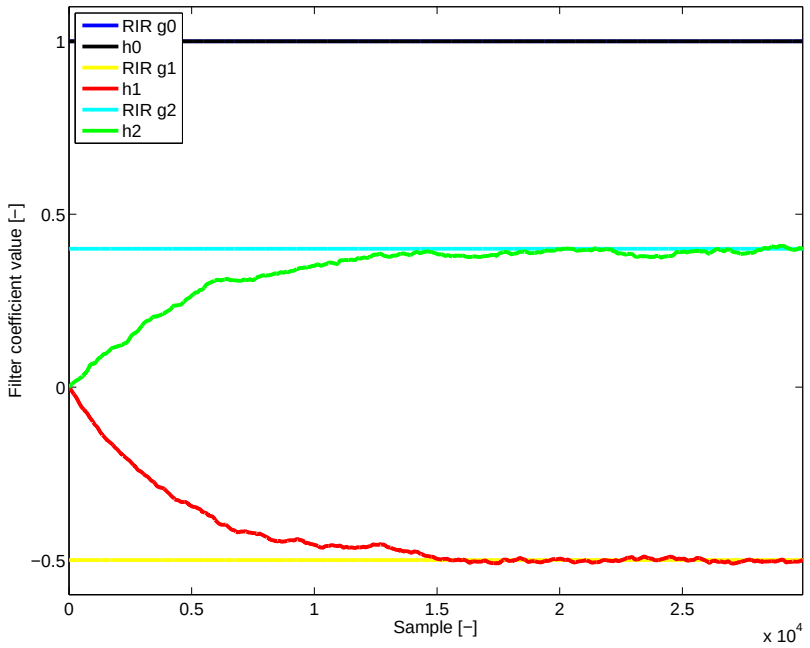


Figure 6.15: Evolution of the three filter coefficients in the equalizer FIR filter.

Next the algorithm is tested with regards to a more complicated all-pole filter given in the

z-domain

$$H_{RIR}(z) = \frac{1}{1 - 0.5z^{-11} + 0.4z^{-15}} \tag{6.22}$$

This means the perfect equalizer filter will be a 16 tap FIR filter.
Once again $M = 100$ and the filter length L is tested as a function of the parameter a. The parameter a is set such that it is possible to obtain sigmoids for sub-Gaussian ($a << 1$), Gaussian ($a \approx 1$) and super-Gaussian sequences ($a >> 1$) where the initial filter is $\mathbf{h}$ =[1 0 ... 0]. Figure 6.16 shows the results.

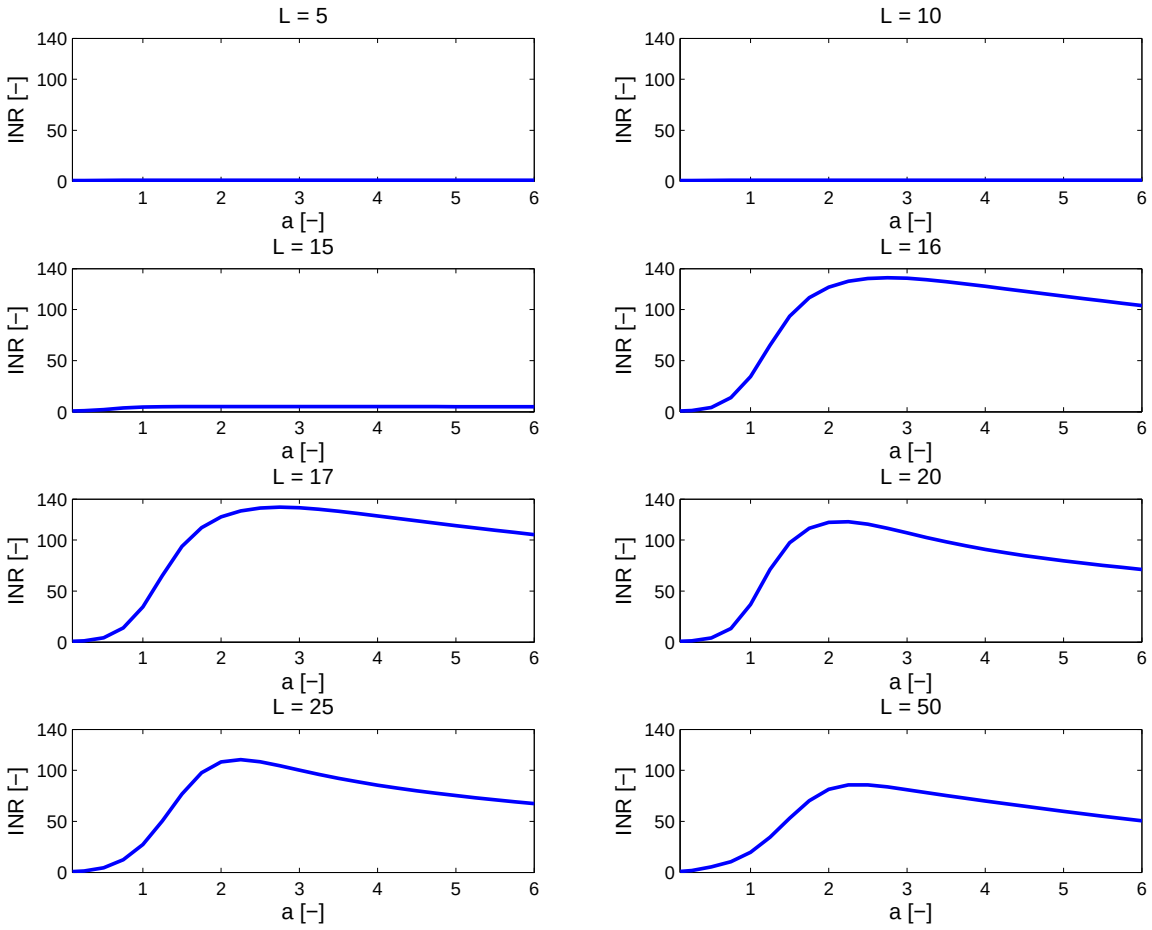


Figure 6.16: INR as a function of the parameter a for different filter lengths deconvolving an IIR filter.

It would be expected that the maximum INR would be obtained for $a = 5$ for all filter lengths because then the sigmoid equals the cdf of the source signal $\mathbf{s}$. However as seen from the plots the maximum INR values is located in the range $2 \leq a \leq 5$. This can be explained from the high sensitivity to the chosen step size. The advantage however, is that the results show that the algorithm will perform well even though the sigmoid is not matched perfectly with the cdf of the source, that is the sensitivity to a is not high. It is seen that a too small order of the filter $\mathbf{h}$ results in a highly reduced INR. If the order of the filter is increased to much the INR also starts to decrease because the extra filter coefficients adds convolution noise to the system since the algorithm cannot terminate them.

A much harder RIR to deconvolve is the all-zero FIR filter given by

$$H_{RIR}(z) = 1 - 0.5z^{-11} + 0.4z^{-15} \tag{6.23}$$

The reason is that an IIR filter is required when equalizing the FIR in the equation above. This means a much longer filter length is required since the IIR has to be approximated by a FIR in the algorithm. In theory the optimal length is infinite. Note that a real life RIR will be a FIR filter as well.
In this case $M = 200$ otherwise the parameters a and $\mathbf{h}$ are unchanged. In figure 6.17 the results for this approach are shown.

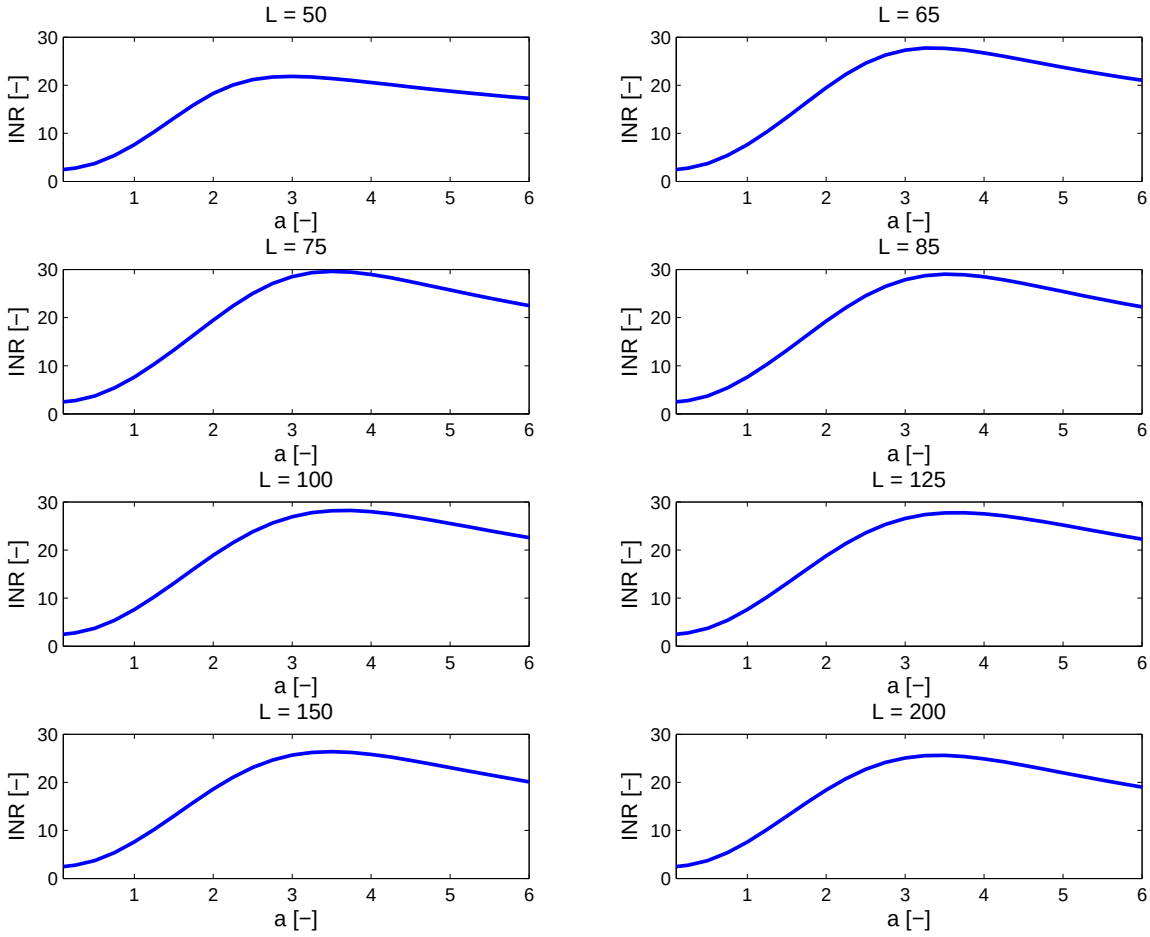


Figure 6.17: INR as a function of the parameter a for different filter lengths equalizing a FIR filter.

Notice that the INR values are decreased significantly compared to the all-pole RIR. The same conclusions can be drawn as in the case of the all-pole RIR with respect to the choice of a. The maximum INR value is obtained for a filter length of $L = 75$. If the filter length is higher than this, it is simply better just to terminate these extra coefficients.

6.2 Summary

In this section the maximum entropy approach was considered for speech dereverberation. An steepest ascent algorithm for this purpose was successfully implemented with a maximum INR

of 111.71 dB for a tree tap RIR and an input of 10^5 samples. The project group presented a flexible sigmoid, that easily can be adjusted to the a priori knowledge of the source. Tests of the algorithm were made under optimum conditions, where the sigmoid where perfectly matched with the pdf of the source signal. The project group suggested to hold the first filter coefficient of the equalizing filter fixed and this resulted in better INR values as compared to the original algorithm.

6.3 Minimum Entropy

In this section the principle of minimizing the entropy of the equalized signal ($y(n)$ in figure 4.1) is presented. Many algorithms based on the principle of entropy minimization have been published, but in this section the focus is on the work by Erdogmus et al. [10] and Principe et al. [27], which are two closely connected articles. The original idea was presented by Bercher et al. in [28]. Furthermore a method which is used for nonparametric estimation of probability density functions is described in this section. This method is applicable to the minimum entropy algorithm.

As mentioned in the introduction to chapter 4 the CLT can be applied to the received, reverberated speech and the signal can therefore be assumed to be Gaussian distributed. The metric known as Shannon's entropy presented in [11] becomes maximum when the input has a Gaussian pdf and the variance is constant. This can be proved by the following equations

$$H(x) = -\int f_X(x) \log f_X(x) dx \tag{6.24}$$

The constraints are

$$\sigma^2 = \int f_X(x) x^2 dx \qquad 1 = \int f_X(x) dx \tag{6.25}$$

Combining the function, which is to be maximized, and the constraints yields

$$\begin{aligned} L(H(x), \lambda_1, \lambda_2) &= -\int f_X(x) \log f_X(x) dx + \lambda_1 \left(\int f_X(x) x^2 dx - \sigma^2 \right) + \lambda_2 \left(\int f_X(x) dx - 1 \right) \\ &= \int \left(-f_X(x) \log f_X(x) + \lambda_1 f_X(x) x^2 + \lambda_2 f_X(x) \right) dx - \lambda_1 \sigma^2 - \lambda_2 1 \end{aligned} \tag{6.26}$$

Using the fact that the entropy of a continuous random variable is concave, and differentiating the Lagrangian, $L(H(x), \lambda_1, \lambda_2)$, with respect to $f_X(x)$ and setting the result equal to zero gives

$$\begin{aligned} \frac{\partial}{\partial f_X(x)} L(H(x), \lambda_1, \lambda_2) &= -1 - \log f_X(x) + \lambda_1 x^2 + \lambda_2 = 0 \\ &= e^{\lambda_1 x^2} e^{\lambda_2 - 1} \end{aligned} \tag{6.27}$$

Recognizing the similarities between the zero-mean Gaussian distribution and the Lagrangian multipliers

$$\lambda_1 = -\frac{1}{2\sigma^2} \qquad \lambda_2 = 1 - 0.5 \log(2\pi\sigma^2) \tag{6.28}$$

the pdf $f_X(x)$ equals the zero-mean Gaussian distribution and the constraints are therefore satisfied. This completes the proof.

If the Shannon entropy can be minimized by changing the filter it entails that the signal is moved away from being Gaussian towards non-Gaussianty hence the room effect on the signal is removed. The block diagram performing the minimization task is illustrated in figure 6.18. Unfortunately nonparametric estimation of Shannon's entropy, which is desirable, is in general not usable together with adaptive filtering [10], [29, Chapter 7].

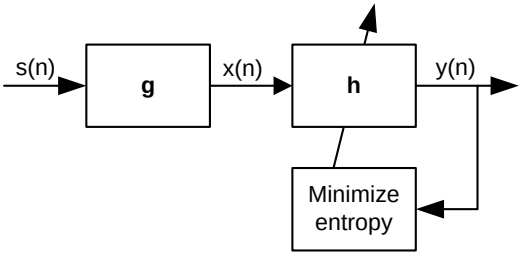


Figure 6.18: The minimum entropy principle applied on a blind deconvolution problem. Based on [10, Figure 1b].

In figure 4.1 it is stated that the input signal is white. When using the principle of entropy maximization it is furthermore necessary to know the pdf of the input a priori, because the algorithm requires an estimate of the corresponding cdf. The minimum entropy approach does not require any a priori knowledge about the pdf hence it is more widely applicable. Actually the approach only requires the input process to be non-Gaussian and white [27].

Erdogmus et al. suggest that Renyi's entropy, which originally was presented by Renyi in [30], is used as a generalization of the Shannon entropy. Renyi's entropy is defined as [10, Eq. 2]

$$H_\alpha(Y) = \frac{1}{1-\alpha} \log \int_{-\infty}^{\infty} f_Y^\alpha(y) dy \tag{6.29}$$

where: $H_\alpha(Y)$ is Renyi's entropy of order α
Y is a continuous random variable with iid samples $\{y_1, y_2, ...\}$
$f_Y(y)$ is the pdf of Y

Note that Renyi's entropy is parametric that is it requires knowledge about the input Y specifically the pdf of Y.
When $\alpha \to 1$ Renyi's entropy $H_\alpha(Y)$ goes towards Shannon's entropy $H_S(Y)$ [10, Eq. 3]. The equation is obtained by use of l'Hôpital's rule.

$$H_S(Y) = -\int_{-\infty}^{\infty} f_Y(y) \log f_Y(y) dy = -E_Y\left[\log f_Y(Y)\right] \tag{6.30}$$

As proven earlier the Shannon entropy is the ideal Gaussianity measure in the sense that the entropy is maximum when the process is Gaussian. Renyi's entropy of orders different than 1 are however not equally good measures since they are not guaranteed to attain their maximum value when the input process is Gaussian, but because the nonparametric estimation of Shannon is difficult to use together with adaptive filters Renyi's entropy of other orders is analysed further.

The idea behind the minimum entropy approach is the same as for the kurtosis approach,

chapter 5, that is minimum entropy will lead to a distribution that is as far away from the Gaussian distribution as possible. However as shown by the experimental results for the kurtosis approach in figure 5.5(a), there exist no direct link between a perfect deconvolution (filtering with a pure delay) and the maximum kurtosis value. That is, in some experimental results $K(\mathbf{y}) > K(\mathbf{s})$. It will now be shown that the minimum entropy of the processed signal $H_\alpha(Y)$ is always an upper bound for the entropy of the source signal, $H_\alpha(S)$. Furthermore it will be shown that equivalence between the entropies, that is $H_\alpha(S) = H_\alpha(Y)$, can only occur if and only if perfect deconvolution is obtained. First it is shown that entropy is not scale invariant that is $H_\alpha(aS) \neq H_\alpha(S)$, where the scaling factor $a \in R$. The correct result is

$$H_\alpha(aS) = H_\alpha(S) + \log|a| \tag{6.31}$$

Let $Y = aS$ then $\int \frac{1}{|a|} f_S(\frac{y}{a}) ds = 1$ which entails that $f_Y(y) = \frac{1}{|a|} f_S(\frac{y}{a})$. Solving for $H_\alpha(Y)$

$$\begin{aligned}
H_\alpha(Y) = H_\alpha(aS) &= \frac{1}{1-\alpha} \log \int_{-\infty}^{\infty} f_Y^\alpha(y) dy \\
&= \frac{1}{1-\alpha} \log \int_{-\infty}^{\infty} \left(\frac{1}{|a|} f_S\left(\frac{y}{a}\right) \right)^\alpha dy \\
&= \frac{1}{1-\alpha} \log \int_{-\infty}^{\infty} \frac{1}{|a|^{\alpha-1}} f_S^\alpha\left(\frac{y}{a}\right) ds \\
&= \frac{1}{1-\alpha} \log \left(\frac{1}{|a|^{\alpha-1}} \int_{-\infty}^{\infty} f_S^\alpha(s) ds \right) \\
&= \frac{1}{1-\alpha} \log \frac{1}{|a|^{\alpha-1}} + \frac{1}{1-\alpha} \log \int_{-\infty}^{\infty} f_S^\alpha(s) ds \\
&= H_\alpha(S) + \log|a|
\end{aligned} \tag{6.32}$$

The result in equation (6.32) is now used for the convolution $y(n) = \sum_i r(i)s(n-i)$, where $r(n) = g(n) * h(n)$, see figure 4.1.

$$\begin{aligned}
H_\alpha(Y) &= H_\alpha\left(\sum_i r(i)s(n-i) \right) \\
&\geq H_\alpha\left(r(i)s(n-i) \right) && (6.33) \\
&= H_\alpha(S) + \log|r(i)| \quad , \quad \forall \quad i && (6.34)
\end{aligned}$$

Where the Entropic Young's inequality, [28], has been used in equation (6.33).
Equation (6.34) states that $H_\alpha(Y) = H_\alpha(S)$ if and only if the filter $\mathbf{r}$ is the Kronecker-delta with or without a delay. Furthermore it shows that $H_\alpha(Y)$ is the upper bound of $H_\alpha(S)$. This means that minimization of $H_\alpha(Y)$ leads to minimization of $H_\alpha(S)$ as well. From equation (6.34) it is concluded that entropy minimization is an applicable approach for blind deconvolution.

The next step is to define a cost function J based on Renyi's entropy. If the cost function is solely based on minimizing the entropy with no contraints, the filter coefficients will always approach the trivial solution (all coefficients are equal to zero). Filtering with zero-valued coefficients means the output will become zero hence Renyi's entropy approaches $-\infty$. Erdogmus et al. suggest a cost function that depends on the variance of the processed signal $\mathbf{y}$ to avoid the zero-valued solution. The cost function is $J(Y) = H_\alpha(Y) - \frac{1}{2}\log Var(Y)$. However experimental results with this solution changed the performance landscape dramatically. The minimum entropy were no longer located at the coefficients for the correct deconvolution filter.

The project group therefore suggests the solution, which was also presented for the kurtosis maximization and maximum entropy approaches, that is is to hold the first coefficient fixed, $h_0 = 1$. This will constrain the algorithm to not approach the trivial solution. Furthermore from an implementation point of view the computations required for calculating $Var(Y)$ are saved.

For the maximum entropy approach presented in the previous section, the pdf of the source signal $\mathbf{s}$ was needed. This requires knowledge about the pdf of the source signal, $\mathbf{s}$, for which the entropy is calculated, that is the entropy is parametric and the choise of pdf is based on a priori knowledge. The use of parametric functions is problematic because the statistics including the pdf of the source signal are inherently unknown in a blind deconvolution setup. The minimum entropy approach has the advantage that it requires the pdf for the room output, $\mathbf{x}$, which can be estimated using nonparametric estimation of the pdf. One such method is the Parzen window estimate which was presented by Parzen in [31]. The estimated pdf $\hat{f}_Y(y)$ is [10, Eq. 1]

$$\hat{f}_Y(y_k) = \frac{1}{N} \sum_{i=k}^{N+k-1} \kappa_\sigma(y_k - y_i) \tag{6.35}$$

where: N is the length of the window
κ is the kernel function
σ is the size of the kernel

The use of nonparametric density estimation is described in appendix C. Another common nonparametric density estimator is the histogram. For the minimum entropy algorithm the Parzen window is chosen because it is a smooth and continuous estimate, which can be differentiated. According to Principe et al. [27] the kernel is often chosen to be a Gaussian pdf with zero mean. The kernel size is usually equal to the standard deviation of the kernel.
Combining Renyi's entropy with the approximated pdf yields

$$\begin{aligned}
\hat{H}_\alpha(Y) &= \frac{1}{1-\alpha} \log \left[E_Y \left[\hat{f}_Y^{\alpha-1}(y_k) \right] \right] && (6.36)\\
&= \frac{1}{1-\alpha} \log \left[\frac{1}{N} \sum_{j=k}^{N+k-1} \hat{f}_Y^{\alpha-1}(y_j) \right] \\
&= \frac{1}{1-\alpha} \log \left[\frac{1}{N^\alpha} \sum_{j=k}^{N+k-1} \left(\sum_{i=k}^{N+k-1} \kappa_\sigma(y_j - y_i) \right)^{\alpha-1} \right] && (6.37)
\end{aligned}$$

where

$$\kappa_\sigma(y_j - y_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(y_j - y_i)^2 \right) \tag{6.38}$$

In equation (6.36) the definition of expectation is used on Renyi's entropy, which was defined in equation (6.29).
The minimization of the estimated entropy in equation (6.37) can be performed by differentiating with respect to the filter coefficients $\mathbf{h}$. The cost function J with respect to h_m, $m = \{0, 1, ..., L-1\}$, where $y_k = \mathbf{h}^T \mathbf{x}_k$, $\mathbf{x}_k = [x_k, x_{k-1}, ..., x_{k-L+1}]^T$, is given by

$$J(\mathbf{h}) = \frac{1}{1-\alpha} \log \left[\frac{1}{N^\alpha} \sum_{j=k}^{N+k-1} \left(\sum_{i=k}^{N+k-1} \kappa_\sigma(y_j - y_i) \right)^{\alpha-1} \right] \tag{6.39}$$

The differentiation of the cost function with respect to the equalizer filter $\mathbf{h}$ yields

$$\begin{aligned}
\frac{\partial J(\mathbf{h}(k))}{\partial \mathbf{h}} &= \frac{1}{1-\alpha} \frac{1}{\frac{1}{N^\alpha}\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-1}} \cdot \frac{\partial}{\partial \mathbf{h}}\left(\frac{1}{N^\alpha}\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-1}\right) \\
&= \frac{1}{1-\alpha} \frac{(\alpha-1)\frac{1}{N^\alpha}\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-2}}{\frac{1}{L^\alpha}\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-1}} \cdot \frac{\partial}{\partial \mathbf{h}}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right) \\
&= \frac{\alpha-1}{1-\alpha} \frac{\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-2}\left(\sum_{i=k}^{N+k-1}\kappa'_\sigma(y_j-y_i)\left(\frac{\partial y_j}{\partial \mathbf{h}}-\frac{\partial y_i}{\partial \mathbf{h}}\right)\right)}{\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-1}} \\
&= -\frac{\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-2}\left(\sum_{i=k}^{N+k-1}\kappa'_\sigma(y_j-y_i)\left(\mathbf{x}_j-\mathbf{x}_i\right)\right)}{\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\kappa_\sigma(y_j-y_i)\right)^{\alpha-1}}
\end{aligned} \tag{6.40}$$

where κ'_σ is the scalar in the differentiated Parzen window kernel given by

$$\kappa'_\sigma(y_j-y_i) = -\frac{1}{\sqrt{2\pi}\sigma^3}(y_j-y_i)\exp\left(-\frac{1}{2\sigma^2}(y_j-y_i)^2\right) \tag{6.41}$$

The differentiated cost function is inserted into a standard steepest descent algorithm. The descent algorithm is equal to the ascent algorithm given in equation (5.7) except that the sign of the step is changed.

Experimental Results

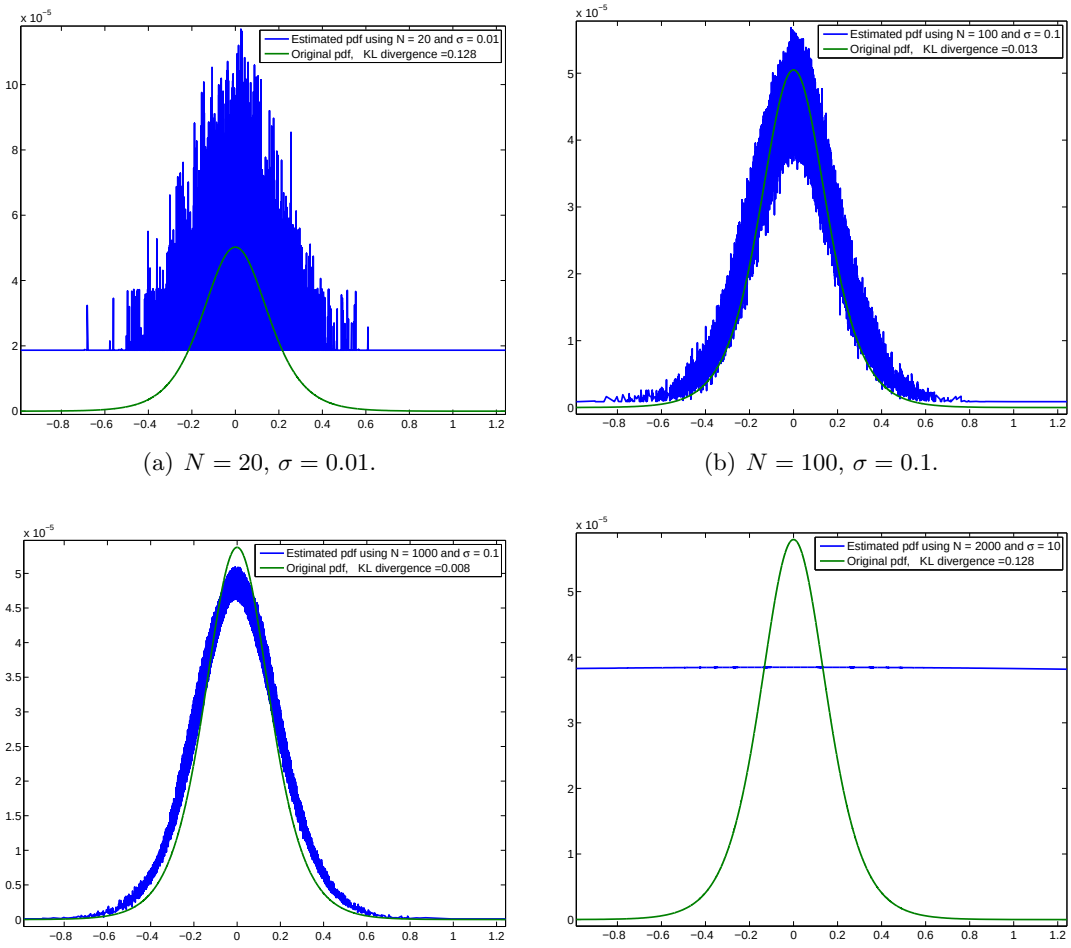
In this section the implemented minimum entropy algorithm is tested. The Matlab implementation is located on the CD in the directory `Algorithms/MinEnt.m`. The idea is to test the algorithm on the RIR filters, which the kurtosis maximization and maximum entropy algorithms where applied to in section 5.1 and 6.1 respectively. The filters are a 3 tap IIR, a 16 tap IIR and a 16 tap FIR. The input is the iid sequence, which was defined in section 6.1 using the accept/reject method described in appendix B. The goal is to test the algorithm's ability to converge to the correct inverse filter and the precision of the obtained filter coefficients. Before the algorithm itself can be tested it is however necessary to validate that the Parzen window is able to estimate the pdf of the received signal $x(n)$.

The validation of the estimate is made by using a measure of the divergence between the estimated pdf and the original pdf defined in equation (6.21). The utilied measure is called the Kullback Liebler divergence (KL divergence) and it was first presented in [32]. The divergence $D(f_1(x)||f_2(x))$ between two pdfs $f_1(x)$ and $f_2(x)$ is defined as [32, Eq. 2.4]

$$D(f_1(x)||f_2(x)) = \int f_1(x)\log\frac{f_1(x)}{f_2(x)}dx \tag{6.42}$$

The KL divergence has the special property that $D(f_1(x)||f_2(x)) \geq 0$, with equality if and only if $f_1(x) = f_2(x)$. This property was presented and proved in Lemma 3.1 in [32].

The Parzen window's ability to estimate the pdf was tested for several window sizes N and standard deviations σ, which is equal to the kernel size in the Gaussian kernel. Furthermore the effect of the location of the Parzen window with respect to the current sample was examined. The test was made for a window located backwards in time, that is the current sample is the newest one, for a centered window where the current sample is in the center of the window and for a window using future samples, that is the current sample is the oldest one in the sequence. All results are given in appendix D.
The conclusion based on the results in appendix D is that the position of the window does not affect the pdf estimate. Furthermore it can be deduced that a large window size, as expected, combined with a small standard deviation gives the best estimate when the input sequence is iid. Using a standard deviation equal to 0.1 generally results in the lowest divergence hence the best estimate, independently of the window size. The problem with a large window size is that it leads to more computations per estimation as compared to a smaller window. Furthermore a small standard deviation results in a more varying estimate as compared to a large kernel which leads to smooth, but as the results show incorrect estimates.
Figure 6.19 shows the estimated pdfs for four different combinations of window size N and standard deviation σ. The used window type is located backwards in time.



(a) $N = 20$, $\sigma = 0.01$.

(b) $N = 100$, $\sigma = 0.1$.

(c) $N = 1000$, $\sigma = 0.1$.

(d) $N = 2000$, $\sigma = 10$.

Figure 6.19: Pdfs estimated with the Parzen window and the matching original pdfs.

Figure 6.19(a) and 6.19(d) show two estimated pdfs, which do no approximate the original pdf very well. This is also reflected in the KL divergence which is 0.128 (written in the subplot

legend) for each of the two estimates.
Figure 6.19(b) and 6.19(c) on the other hand estimates the original pdf reasonably well. When using $N = 1000$ the KL divergence is lower and the estimated pdf is less noisy as compared to the estimate for $N = 100$, but window size $N = 100$ is believed to a be a good trade off between the number of computations and the quality of the pdf estimate.
The conclusion on the test is that the Parzen window is a valid pdf estimate, when the window size and standard deviation are selected carefully. Bad choices with regards to these two parameters results in useless pdf estimates. The test shows that a window size $N = 100$ and a standard deviation $\sigma = 0.1$ gives good estimates.

Now that the Parzen window has been validated the next task is to test the algorithm. First the algorithm is applied to the simple 3 tap IIR defined in equation (5.12) and repeated here for convenience.

$$H_{RIR}(z) = \frac{1}{1 - 0.5z^{-1} + 0.4z^{-2}} \tag{6.43}$$

As mentioned in earlier sections the RIR can be perfectly deconvolved using a 3 tap FIR filter. The reasoning for using the simple filter is to analyse the effect of Parzen window length N, standard deviation of the Gaussian kernel σ, filter length L and the step size in the steepest descent algorithm. The effect of the entropy order is not analysed, because Erdogmus et al. in [10, Fig. 3] showed that the signal-to-interference-ratio (SIR in the figure in the article [10]), which is equal to INR, is not affected by the entropy order ranging from 0.1 to 6. The result for *Rmin for Laplacian Source* in [10, Fig. 3] is the interesting one, since this is for minimum entropy and a super-Gaussian source.

Before the minimum entropy algorithm is used to deconvolve the RIR given in equation 6.43 the performance surface is analysed. The surface itself, shown in figure 6.20 is determined using the definition of Renyi's entropy of order 2 and a Parzen window in equation (6.39).

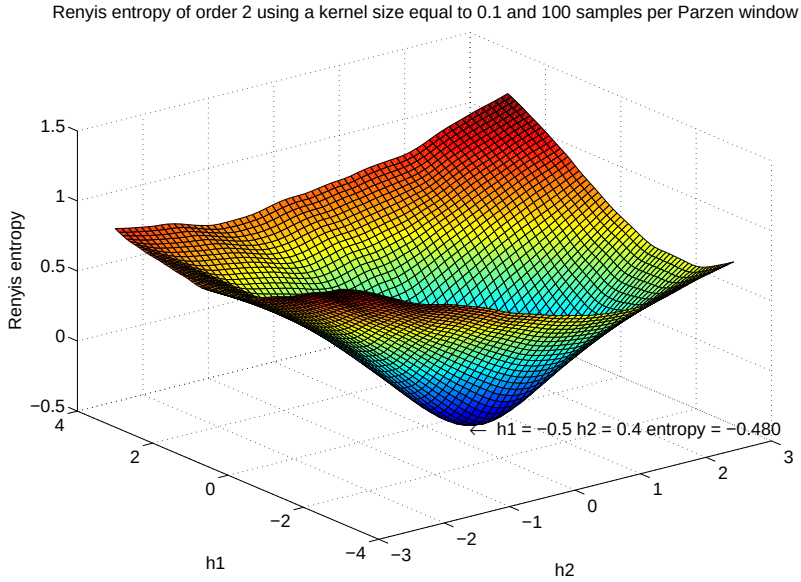


Figure 6.20: Performance surface for the minimum entropy algorithm. Based on the cost function in equation (6.39).

As expected the global minimum of the cost function is located at the perfect inverse of the RIR. Figure 5.6 illustrates the matching INR plot of the performance surface, and it confirms

that the cost function will lead to a single solution.
Now that it has been validated that the cost function, hence the performance surface, results in one global minimum the next step is to apply the algorithm to the signal which has been passed through the filter in equation (6.43). The current algorithm is however slightly more complicated than the kurtosis maximization and maximum entropy algorithms, because it has several parameters which can be adjusted. Earlier in this section the Parzen windows ability to estimate the pdf was evaluated and it was concluded that a window length which is as large as possible is desirable, when the input sequence is iid. A very large window will unfortunately lead to many more calculations as compared to a smaller window, and furthermore the correlation, caused by the RIR, between the first and the last sample in such a large window will be negligible. E.g. the current 3 tap IIR filter only affects roughly 20 samples at a time as illustrated by figure 6.21 where it is seen that the "tail" of the impulse response is approximately zero after 20 samples.

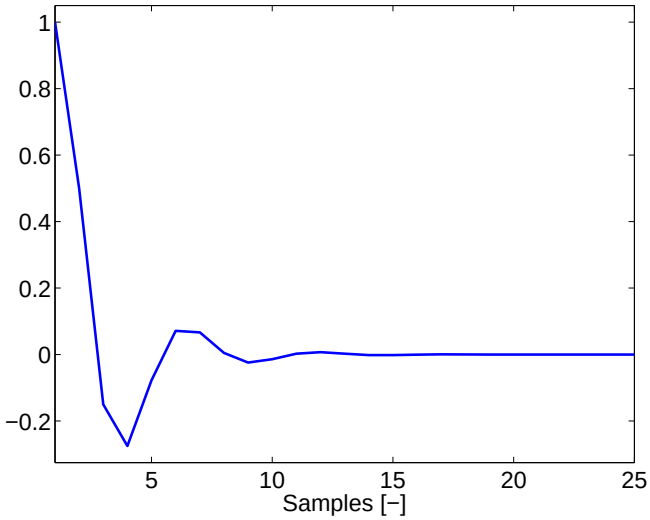
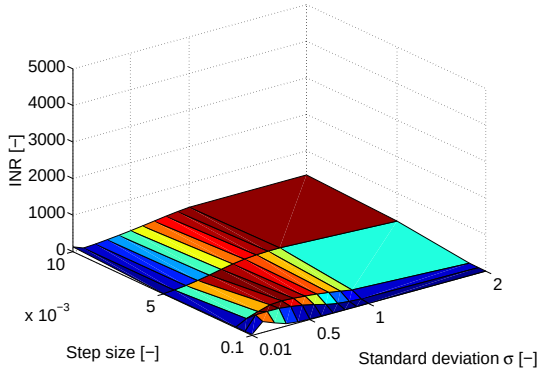
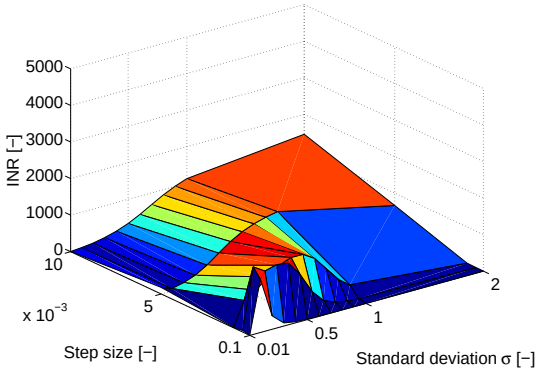


Figure 6.21: The impulse response of the IIR in equation (6.43).

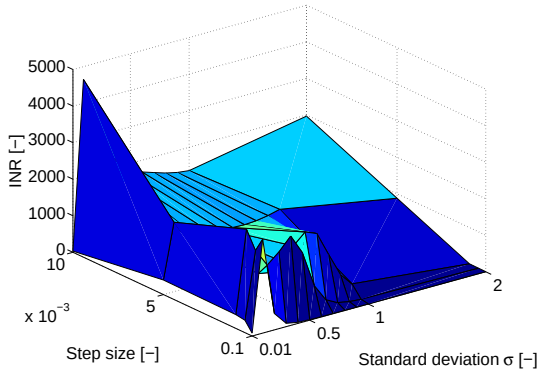
Therefore the window size is adjusted together with the standard deviation of the Gaussian kernel and the step size of the algorithm. Figure 6.22 shows the result of a simulation, where the INR was evaluated for four different window lengths as a function of the step size and the standard deviation. All simulations are made using $50 \cdot 10^3$ samples and a filter length L equal to three, which is the optimal length.



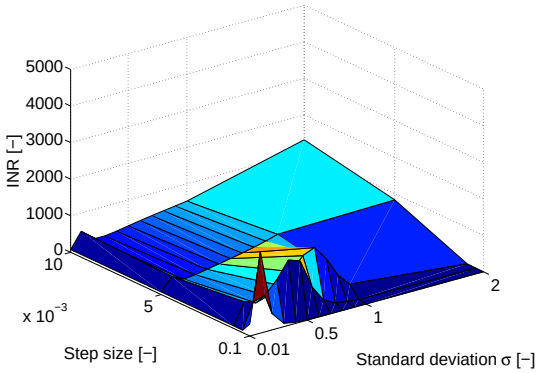
(a) Window size N = 20.



(b) Window size N = 50.



(c) Window size N = 100.



(d) Window size N = 200.

Figure 6.22: INR as a function of step size and standard deviation using the 3 tap IIR filter defined in equation (5.12).

Analyzing the plots in figure 6.22 it is seen that the largest INR, hence the best deconvolution, is obtained when the window size is set to 100, the standard deviation is 0.1, and the step size is 0.01, which is the case in subfigure 6.22(c). This result is highly unexpected because the large step size means that it will be difficult for the algorithm to adjust the filter coefficients in detail. Plotting the filter coefficients which correspond to the chosen settings, the fact that the algorithm cannot adjust the coefficients becomes evident, as illustrated in figure 6.23. The figure shows that the algorithm quickly adjusts towards the correct coefficient but that it is not able to fine tune the coefficients. The reason why such a large INR value is obtained is roughly said a coincidence, because the algorithm was fortunate to be stopped at a point where the coefficients where at a center point of their oscillating behavior. If the algorithm had been stopped at a point where the coefficients where at a minimum or maximum of their oscillation the INR value would have been much lower.

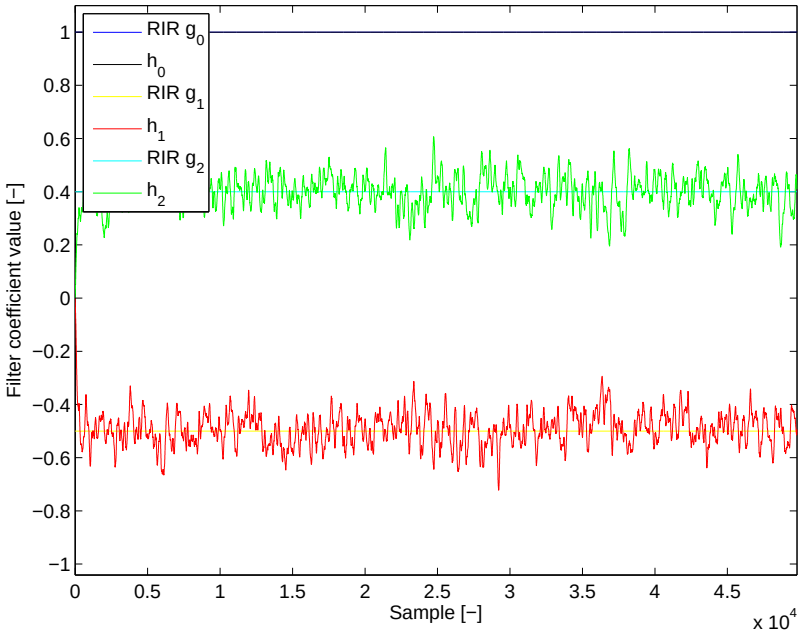


Figure 6.23: Evolution of the filter coefficients when using $N = 100$, $\sigma = 0.1$, and step size equal to 0.01.

Analyzing figure 6.22 to find the new best settings it is clear that the solution must be found in subfigure 6.22(c) once again. The second and third largest values of all the plots in figure 6.22 are also located in subfigure 6.22(c) as illustrated in figure 6.24.

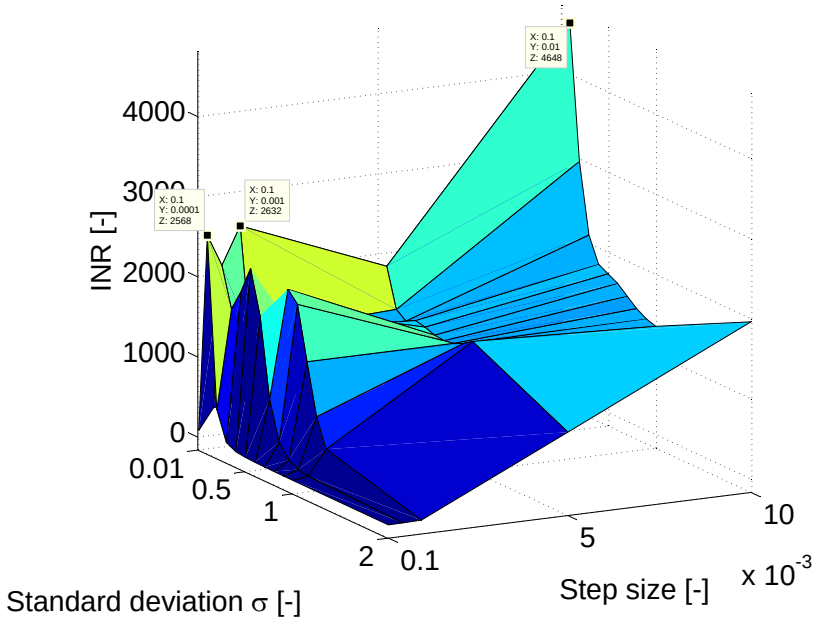


Figure 6.24: Subfigure 6.22(c) with indication of the local maxima.

Based on the INR values the next best settings are with a window size equal to 100, the standard deviation is 0.1, and the step size is 0.001. Figure 6.25 shows the evolution of the filter coefficients. The conclusion is the same as with figure 6.23, that is the algorithm has trouble making the final adjustments hence the filter coefficients oscillate around the correct

values.

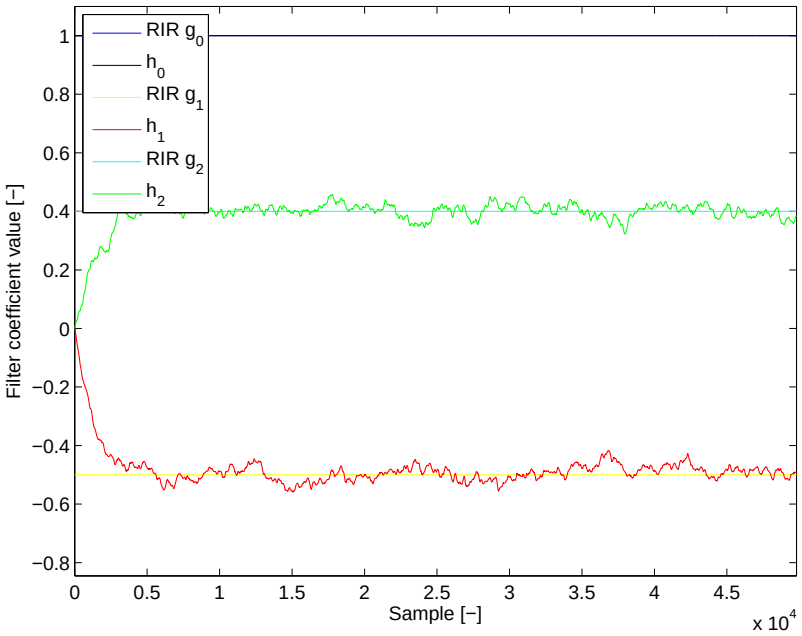


Figure 6.25: Evolution of the filter coefficients when using $N = 100$, $\sigma = 0.1$, and step size equal to 0.001.

Because of the algorithms misadjustments illustrated in figure 6.25 the third highest INR value given in figure 6.24 is used. The settings, when this value was obtained, are window size equal to 100, standard deviation equal to 0.1 and a step size equal to 0.0001. Figure 6.26 illustrates that the algorithm is able to adjust the filter coefficients towards the correct value and that the oscillating behavior has been reduced. Unfortunately the convergence rate is also lower because of the small step size.

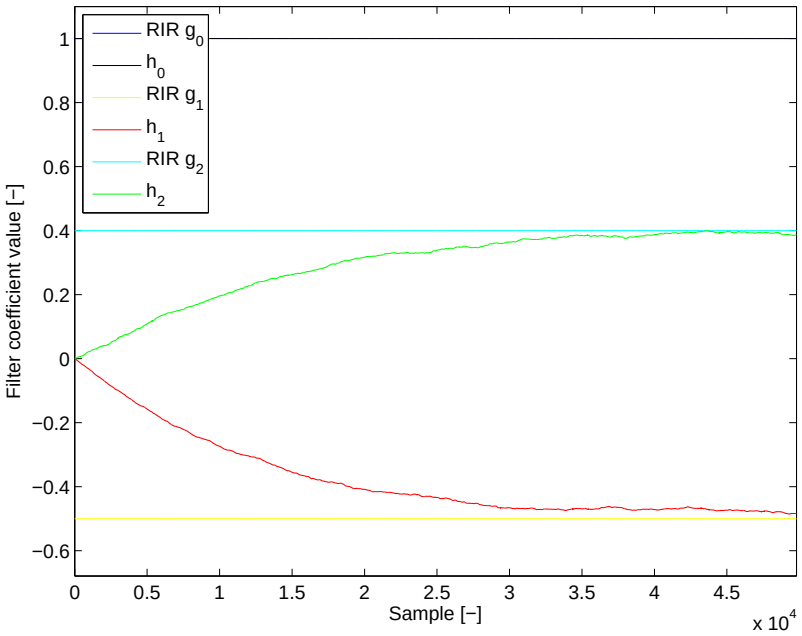


Figure 6.26: Evolution of the filter coefficients when using $N = 100$, $\sigma = 0.1$, and step size equal to 0.0001.

The latter settings are however used, when performing the rest of the tests in this section, because they reduce the oscillation so dramatically.

The performance plot in figure 6.20 is redrawn as a contour plot in figure 6.27, where the gradients calculated with equation (6.40) are plotted as well using the settings $N = 100$ and standard deviation equal to 0.1. Figure 6.27 clearly illustrates that by going in the opposite direction of the gradients the algorithm will approach minimum entropy.

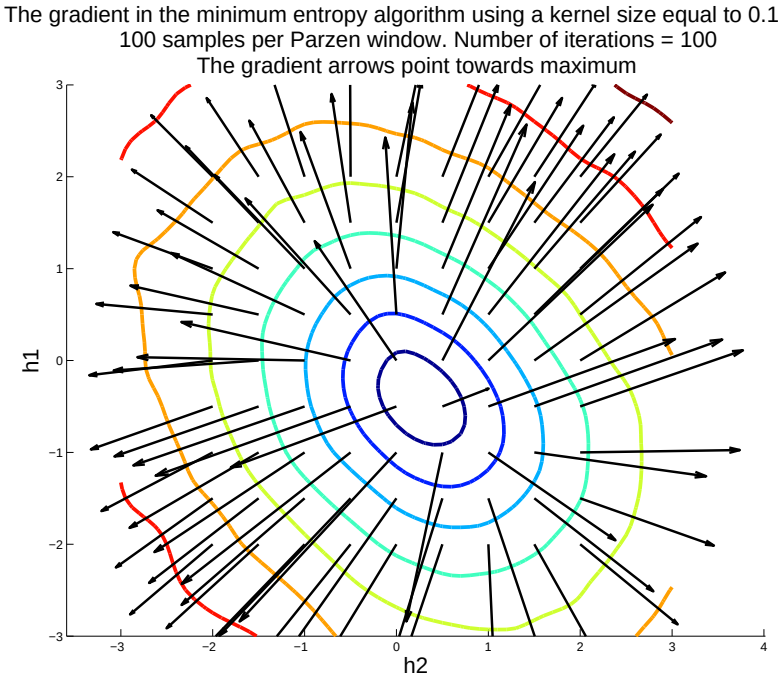


Figure 6.27: Contour plot of the minimum entropy performance surface, where minimum is indicated by the blue colours and maximum is at the red colours. The gradients, pointing towards maximimum entropy, are calculated using equation (6.40).

Now that the required algorithm parameters have been determined and it has been established that the gradient point towards maximum, the next step is to evaluate the algorithm's ability to deconvolve the RIR defined in equation (6.43). Figure 6.28 illustrates eight convolutions between the RIR and the equalizer filter, when the algorithm is applied to $50 \cdot 10^3$ samples.

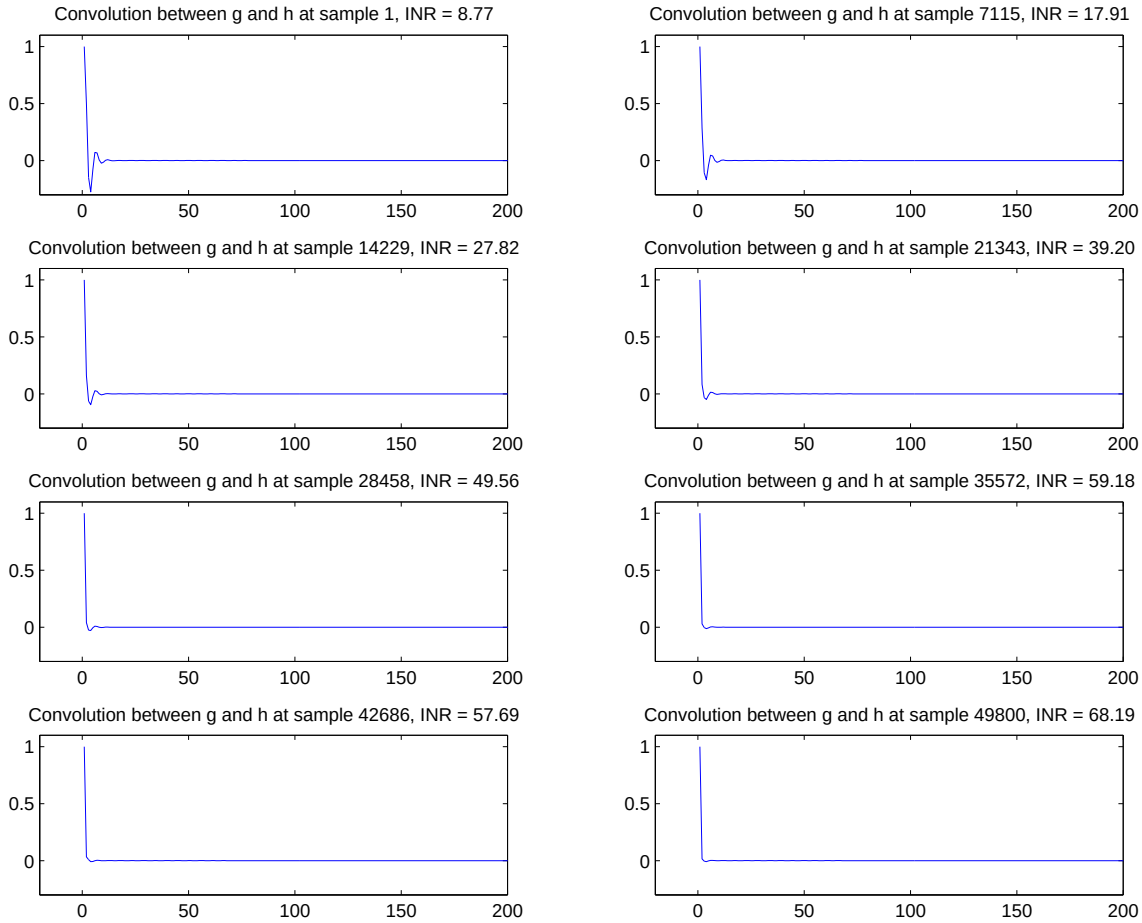


Figure 6.28: Convolutions between the RIR and the equalizer filter using the minimum entropy algorithm.

Figure 6.29 shows the final plot in figure 6.28 and figure 6.30 shows the convolution noise in the final convolution from figure 6.29.

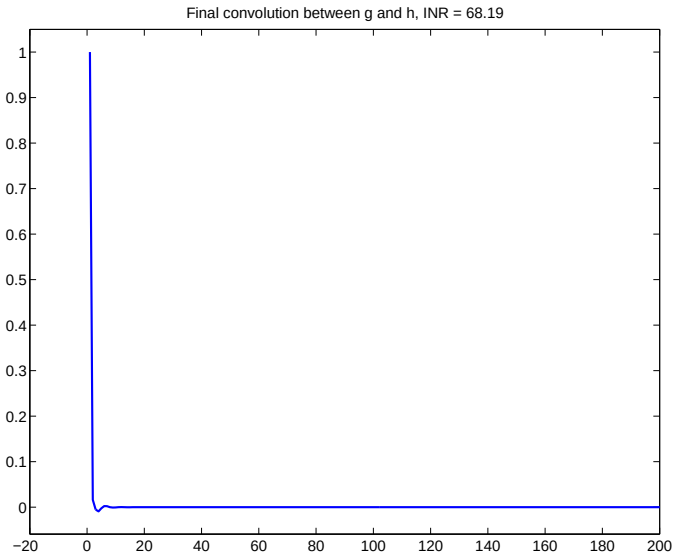


Figure 6.29: The final convolution between the RIR and the equalizer filter.

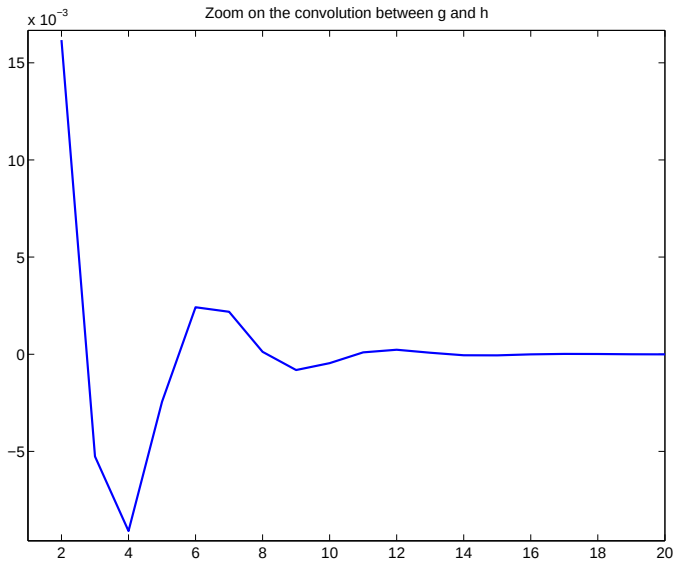


Figure 6.30: The convolution noise in the final convolution between the RIR and the equalizer filter.

The figures 6.28, 6.29, and 6.30 clearly show that the minimum entropy algorithm converges towards the correct filter. The point is finally substantiated with a contour plot of the performance surface from figure 6.20 combined with the "path" the algorithm follows during the simulation. This is illustrated in figure 6.31, which shows that algorithm updates the filter coefficients towards the correct point in $h_1 = -0.5$ and $h_2 = 0.4$.

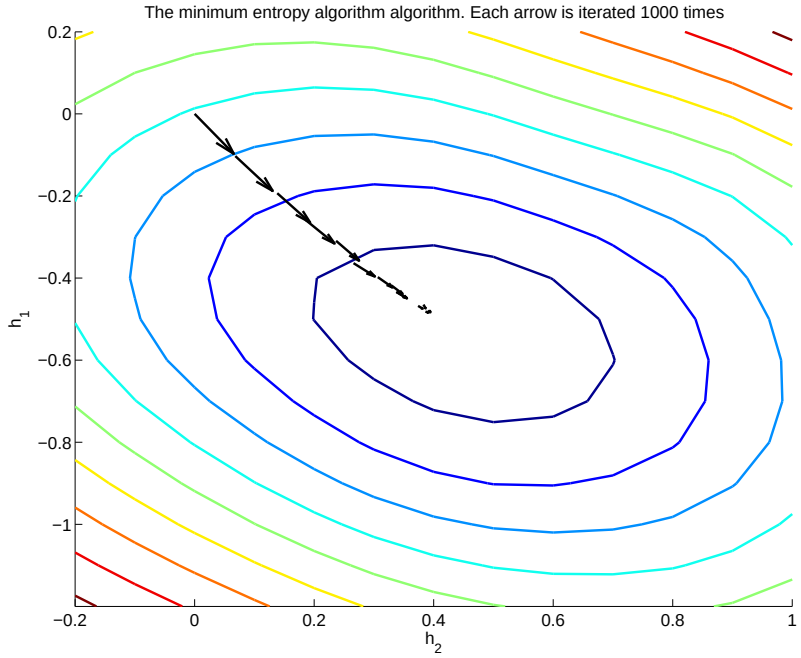


Figure 6.31: Contour plot of the performance surface together with the location of the filter coefficients and the corresponding gradients, calculated via the minimum entropy algorithm.

Note that the arrows become smaller and smaller when the global minimum is approached. The reason is that the surface is more flat near the minimum as illustrated in figure 6.20. Based on the above description it is concluded that the minimum entropy algorithm is able to

deconvolve the 3 tap IIR filter. A test using 10^5 samples has been made and it confirms that the algorithm is able to converge very closely to the correct equalizer filter leading to an INR value equal to 93.94 dB.
To further test the algorithm it is now applied to a signal, which has been reverberated using a 16 tap IIR filter defined in equation (5.13). Using a 16 tap IIR as the RIR means that the correct equalizer filter length will be a 16 tap FIR filter. Keeping the window size, standard deviation and step size constant the effect of different filter lengths is plotted in figure 6.32.

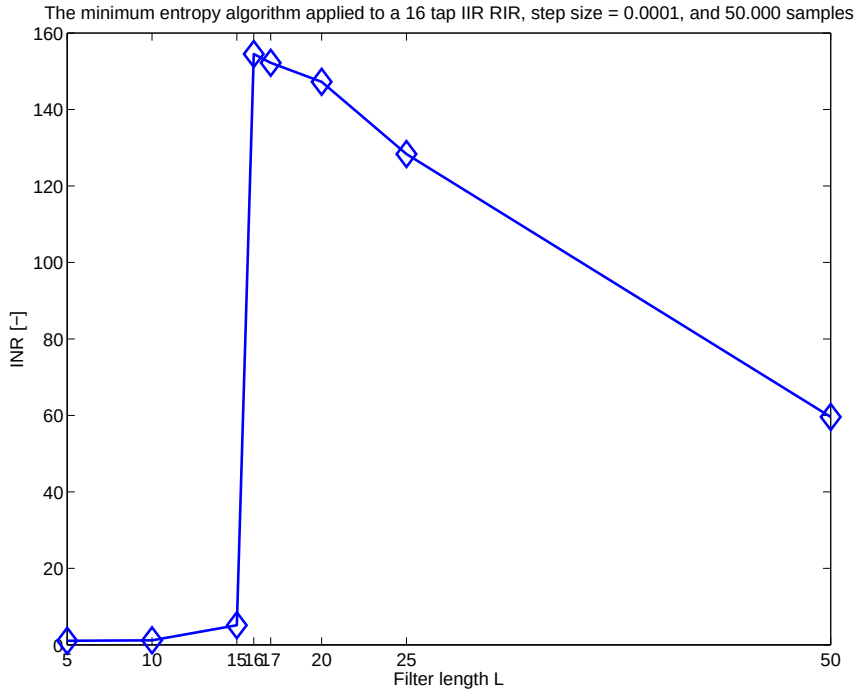


Figure 6.32: The INR value of the convolution between a 16 tap IIR RIR and the equalizer FIR, as a function of the equalizer filter length.

As expected the optimal length is equal to 16. The figure also shows that the algorithm is able to deconvolve the RIR with reasonable good results using larger filters, but that filter lengths smaller than 16 taps are insufficient.
Next the minimum entropy algorithm is tested on a 16 tap all-zero FIR RIR. The filter is defined in equation 5.14. Because the filter is of the FIR type the perfect inverse filter is an IIR filter. The minimum entropy algorithm is only capable of adjusting FIR filters and therefore the IIR equalizer is approximated with a FIR filter. Because of the approximation the optimal filter length is unknown and the INR value of the deconvolution is calculated as a function of the filter length. The result is shown in figure 6.33. Again the window size, standard deviation and step size was constant.

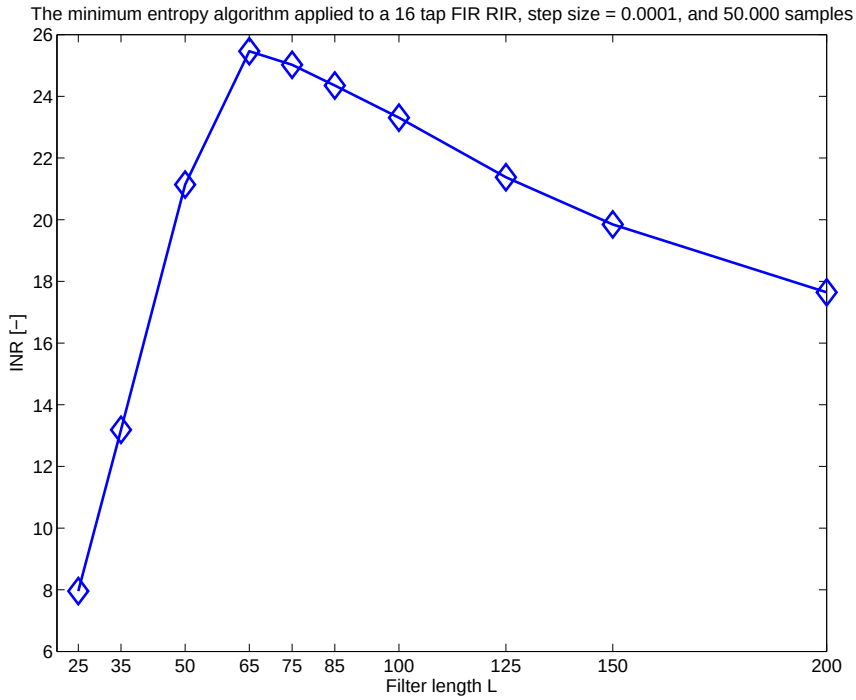


Figure 6.33: The INR value of the convolution between a 16 tap FIR RIR and the equalizer IIR, approximated with a FIR, as a function of the equalizer filter length.

Figure 6.33 shows that the best deconvolution is achieved using a filter length equal to 65. This concludes the validation tests of the minimum entropy algorithm. The conclusion is that the algorithm works, that is it is able to deconvolve the 3 tap IIR, 16 tap IIR and 16 tap FIR test filters using the input, with the known pdf calculated in section 6.1.

6.4 Summary

In this section the minimum entropy approach to blind speech deconvolution was presented and analysed. Using a steepest descent formula and a cost function based on the minimum entropy approach the algorithm was successfully implemented in Matlab and tested on three types of RIR filters: a 3 tap IIR, a 16 tap IIR and a 16 tap FIR. The maximum INR value for the 3 tap IIR is 93.9 dB using $100 \cdot 10^3$ input samples. The algorithm utilizes a Parzen window to estimate the pdf of the input signal, and therefore the validity of this estimate was also examined. Using a window size equal to 100 and a Gaussian kernel with a standard deviation equal to 0.1 results in good estimates. The pdf is input to Renyi's entropy, which is to be minimized to obtain the minimum entropy, and by plotting the entropy as a function of the filter coefficients in a 3 tap RIR it was determined that the matching performance surface has a global minima as desired. Renyi's entropy was set to be of order two.
To avoid the trivial solution, where all filter coefficients are adjusted to zero and Renyi's entropy becomes $-\infty$, the project group suggested a normalization, where the first filter coefficient is set to one after each iteration. In the article by Erdogmus et al. [10] it was suggested that the trivial solution could be avoided by using another type of scaling, but this means that the performance surface is affected such that the minimum entropy no longer corresponds to the correct inverse filter. Therefore the project groups own method was used in stead.

Compared to the kurtosis maximization algorithm the minimum entropy algorithm has the advantage that the the entropy $H_\alpha(Y)$ is the upper bound of the $H_\alpha(S)$. This property means that by minimizing $H_\alpha(Y)$ the entropy $H_\alpha(S)$ is also minimized.

This was the third and final presentation of algorithms, analysed and implemented by the project group. In the next chapter the algorithms will be compared with respect to INR and convergence speed using Monte Carlo simulations. The algorithms are also applied to more complex RIRs where the task is to remove late reverberation and equalize the RIR, which is described in appendix F. Furthermore the computational complexity of the three algorithms is examined and compared.

7

Tests and Comparison of the Approaches

The presented approaches to blind speech dereverberation have been analysed individually in chapter 5 and 6. The approaches are kurtosis maximization, maximum entropy and minimum entropy. In this chapter the methods and the pros and cons of the algorithms will be highlighted together with the modifications and suggestions made by the project group. The algorithms will be compared according to INR and convergence speed. Furthermore their ability to reduce the late reverberations will be compared. In chapter 8 the complexity of the algorithms is analysed. This chapter will start with a theoretical comparison of the three algorithms.

All three methods assume that the source signal, $\mathbf{s}$, consists of iid samples. Then according to the CLT, the room output, $\mathbf{x}$, can be considered to be Gaussian distributed, see figure 4.1.
The kurtosis algorithm weigh the equalizing filter to the RIR according to the maximum kurtosis value of the output, $\mathbf{y}$. The algorithm by Gillispie et al., [6], adjusts the filter coefficients according to the kurtosis of the LP residuals. However, assuming the samples of the source signal is iid distributed, this will bias the location of the maximum kurtosis value and move it away from the location of the correct equalizing filter. This was shown in figure 5.5. The project group therefore removed the LP analysis from the algorithm, and the algorithm operates directly on the room output. It is a necessary condition that, $K(\mathbf{y}) = K(\mathbf{s})$, such that the convolution between the RIR and the equalizer will correspond to a pure delay. However no proof states, that $K(\mathbf{y}) \geq K(\mathbf{s})$, that is there exists no proof that $K(\mathbf{y})$ is an upper bound for $K(\mathbf{s})$. This entails that there exists no guarantee that a too high kurtosis value cannot be obtained, which will lead to the wrong equalizing filter. This observation by the project group degrades the usability of the already complex performance landscape for the kurtosis approach, which has many local maxima. Furthermore the algorithm had difficulties converging to the correct equalizing filter. The project group therefore suggests that the first filter coefficient is fixed through out the iterations. Test shows this entails a much higher convergence reliability. The normalization were later applied to all the three algorithms, where it furthermore helps the algorithms avoid the zero-valued solution.

From an information point of view the problem of speech dereverberation can be viewed as redundancy reduction, because the information each source sample carries has been changed by the RIR and each room output sample is a weighted delayed version of itself, hence the redundancy has been increased. Bell et al. present in [9] and algorithm for minimizing the mutual information between samples by maximizing the entropy of the output. The main difference to the kurtosis approach is that in the maximum entropy approach, the coefficients of the equalized filter are adjusted according to all the HOS (infinitely many) and not just kurtosis. The HOS are accessed through a nonlinear function which should be matched to the cdf of the source signal. Bell et al. suggest $\tanh(\cdot)$ as the nonlinear function and express the need for a flexible nonlinear function that can be modified according to the a priori knowledge of the source signal. The project group suggests the modified $\tanh(a \cdot \mathbf{y})$ function, $a \in R$, which seems to be a good match to the cdf of the direct speech. Due to its simplicity it can easily be modified by choosing the constant a according to the knowledge of the source signal.

The minimum entropy approach does not assume any a priori knowledge of the source signal, and can thereby be viewed as a more general solution to the dereverberation problem. The algorithm operates directly on the room output, where the filter coefficients of the deconvolution filter is adjusted by minimizing the entropy of $\mathbf{y}$, knowing that the maximum entropy of a unbounded input corresponds to the Gaussian distributed signal, when the variance is constant. Because the pdf of the processed output, y, is needed and not the pdf of the source (as in maximum entropy), the algorithm does not need any a priori knowledge of the source. The pdf of $\mathbf{y}$ is then estimated by a non-parametric estimation. For the implemented algorithm, the Parzen windowing method using a Gaussian kernel was chosen. Renyi's entropy is introduced because it works better in adaptive filtering as compared to Shannon, [10] and [29]. It is important to notice that maximum entropy for a Gaussian unbounded input is only proven for Shannon's entropy and not Renyi's. The minimum entropy approach can be seen as very similar to the kurtosis approach but with one major difference. As shown in section 6.3 then $H(Y) \geq H(S)$ with equality if and only if the deconvolution is a pure delay. That is it has been shown that in contrast to the kurtosis approach $H(Y)$ is always an upper bound to $H(S)$.

The three implemented algorithms were tested individually on the same source signal, which was sampled from the pdf that matched the nonlinear function in the maximum entropy approach. The algorithms were tested to provide a proof of concept. The uncertainties in the tests, such as variance of the correct parameters and on the INR performance metric, were not taken into account. The reason is the time limitation of one academic semester. Monte Carlo simulation should have been performed for different source signals generated from an accept/reject algorithm, described in appendix B. Furthermore when reporting INR values as a function of the step size the input length will be tuned to that specific step size, which is not possible in a real life situation. However, as stated the previous tests are proof of concepts, and they show that the algorithms work.

In the following sections, the algorithms will be compared to each other with respect to INR, convergence speed, and their ability to reduce late reverberation. Monte Carlo simulations will be performed to improve statistical reliability.

7.1 Impulse-to-Noise Ratio

In table 7.1 the INR values in dB are given for the three algorithms. Throughout the chapter all INR values reported are in dB. The RIRs are the used RIRs introduced in chapter 5 and 6, that is the 3 tap IIR, the 16 tap IIR and the 16 tap FIR. Note that the number of input samples and step size are highly correlated in the sense that many samples means a smaller step size can be used, which again means that the coefficients are tuned more accurate. The input length is set to $100 \cdot 10^3$ samples such that all three algorithms are able to converge. For the minimum entropy algorithm the window size is $N = 100$ with a kernel size of $\sigma = 0.1$. The step size has been tuned for the algorithms individually for each RIR. A total of 100 Monte Carlo simulations have been made for each algorithm and each RIR and the average INR value is reported in table 7.1. Running 100 Monte Carlo simulations means that the algorithm is evaluated on 100 input signals, drawn from the same source. The result is the average of the 100 simulations.

RIR	Kurtosis	Max. entropy	Min. entropy
3 tap IIR	93.82	112.52	77.79
16 tap IIR	53.34	47.92	58.82
16 tap FIR	27.41 (65)	30.36 (75)	28.34 (65)

Table 7.1: INR for different RIRs measured in dB. For the 3 tap and 16 tap IIR the correct inverse filters are a 3 tap and 16 tap FIR filter, respectively. The $(\cdot)$ denotes the lengths of the equalizing IIR, which is approximated with a FIR.

From the tests it seems that for the simple RIR and 16 tap FIR, the maximum entropy algorithm performs the best, whereas for the 16 tap IIR, minimum entropy performs the best. However the INR values are heavily dependent on the input length and the step size. Therefore the mean values and the variances of the determined filter coefficients are examined. In table 7.2 these are listed for the simple 3 tap RIR. The variance has been calculated for $5 \cdot 10^3$ samples after the coefficient is in the range of $h_i \pm 5\%$ for $i = \{1, 2\}$ and for the next $10 \cdot 10^3$ samples

Filter coeff.	Kurtosis	Max. entropy	Min. entropy
h_1 (-0.5)	-0.485 $\pm$ 0.616$\cdot 10^{-5}$	-0.489 $\pm$ 0.494$\cdot 10^{-5}$	-0.483 $\pm$ 0.377$\cdot 10^{-5}$
h_2 (0.4)	0.389 $\pm$ 0.703$\cdot 10^{-5}$	0.392 $\pm$ 0.527$\cdot 10^{-5}$	0.389 $\pm$ 0.464$\cdot 10^{-5}$

Table 7.2: Mean and variance for the determined coefficients.

All three algorithms reach a fairly small variance. The maximum entropy achieves the best mean value of the determined filter coefficients. Only a difference of 0.002 exist between kurtosis and minimum entropy for h_1, whereas the coefficient h_2 is approximately the same for both algorithms. However the INR values report of approximately 15 dB difference. This again illustrates that the INR values are very sensitive and must be analysed carefully.
The conclusion is that the maximum entropy algorithm results in the best INR values, and because it from a statistical point of view in general converges more accurately to the correct solution the maximum entropy algorithm is concluded to have the best convergence capabilities for the 3 tap IIR filter.

7.2 Convergence speed

In table 7.1 some algorithms achieves similar INR values. However as shown in this section, the convergence speed is not the same for the algorithms. In figure 7.1 the comparison between the three algorithms has been made for the simple 3 tap RIR with 100 Monte Carlo simulations. The step size for kurtosis maximization is $1.21 \cdot 10^{-5}$, 10^{-6} for maximum entropy and 10^{-4} minimum entropy.

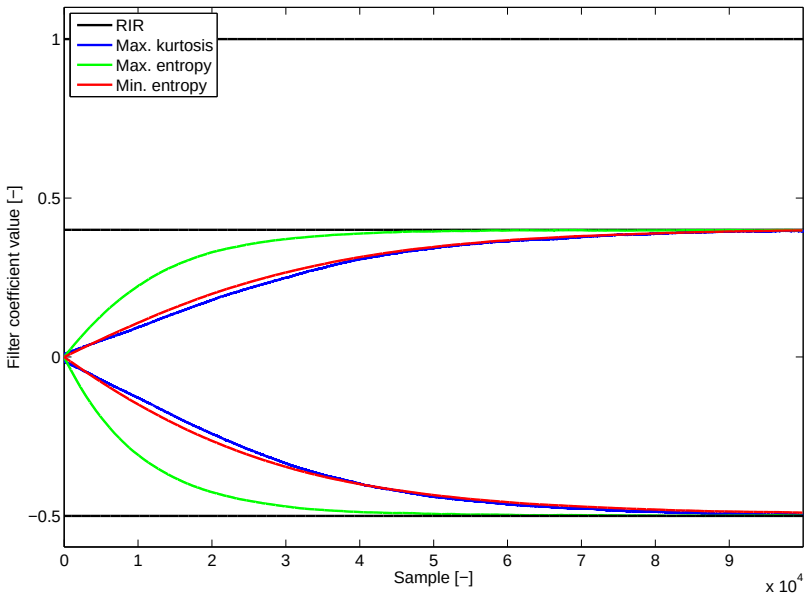


Figure 7.1: Comparison of the convergence speed for the three algorithms for the 3 tap IIR using 100 Monte Carlo simulations.

The figure shows that the maximum entropy algorithm is approximately two times faster than the other algorithms for the simple RIR. The trade of between convergence speed and step size must of course be taken into account. However the maximum entropy algorithm was tested with the smallest step size of the three algorithms. In figure 7.2 the convolution results for the 16 tap IIR are shown. The convolution results are for 100 Monte Carlo simulations, and the INR values are averaged. The upper left plot is the convolution result for the first sample and the lower right plot is the result for the last sample. The plots in between are for six samples equally spaced between the first and last sample, which is sample number $100 \cdot 10^3$.

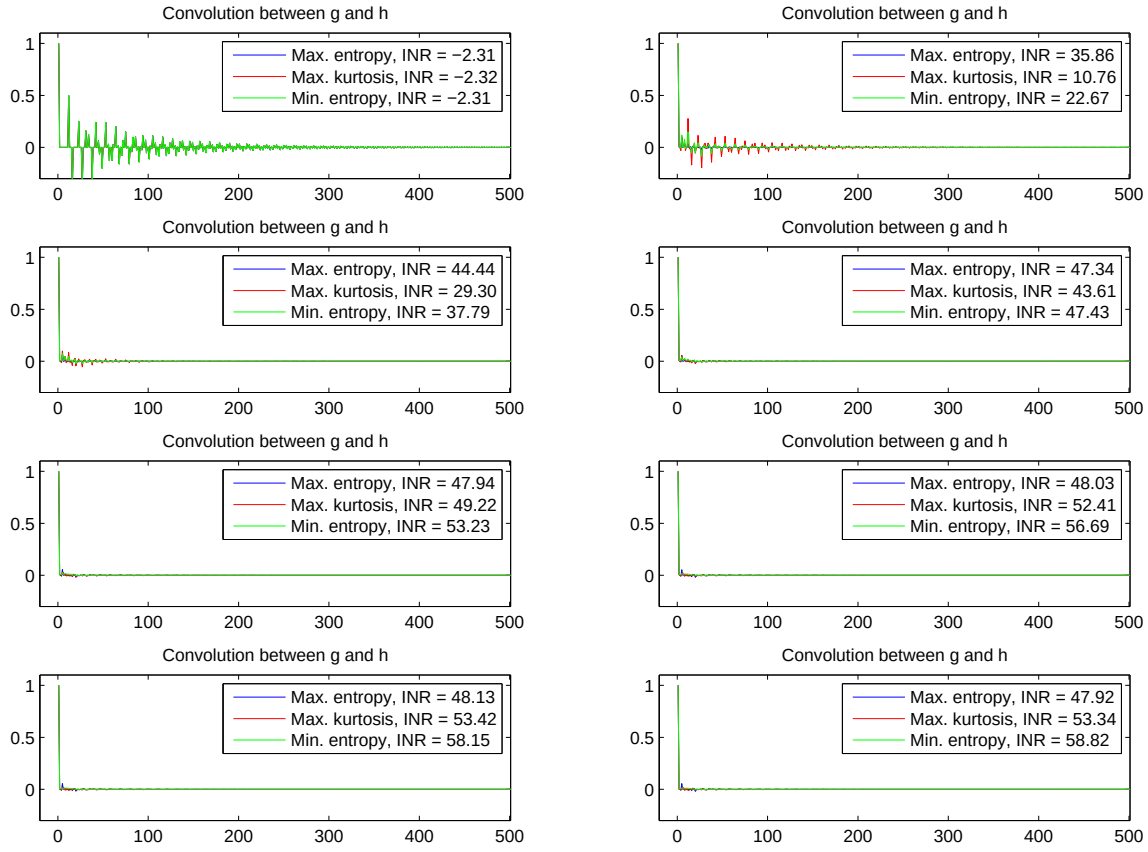


Figure 7.2: Comparison of the convergence speed for the three algorithms for the 16 tap IIR.

In figure 7.2 it can be seen that the two entropy algorithms reach convergence fairly quickly compared to the kurtosis algorithm. Again the maximum entropy obtains the fastest convergence, but the minimum entropy algorithm approaches a higher INR value. The maximum entropy algorithms achieve the best convergence speed for the more difficult 16 tap FIR RIR, approximately 2/3 faster than the kurtosis and the minimum entropy approach.

7.3 Reduction of Late Reverberation

In the previous test short length RIRs were used. This entails that the reverberation can be considered as early reverberation and the effect is a colouration of the source signal. In this section the late reverberation or echoes are considered. As defined in section 2.1 the late reflections occur 100 ms or more after the direct signal has arrived. The cost is a delay equal to the time span between the arrival of the first direct speech sample and the last late reflection sample. Hence, if it is chosen to reduce possible late reflections a delay of at least 100 ms must be accepted.
In the following the algorithms are tested upon an IIR and a FIR RIR, shown in figure 7.3. Monte Carlo simulations have not been made, because of time limitations. Figure 7.3 illustrates that the required filter length, when deconvolving the FIR RIR will be approximately 8000, because the IIR RIR, which is the inverse of the FIR RIR, has peaks around one second. Because of time limitations the algorithms were not run with this filter.

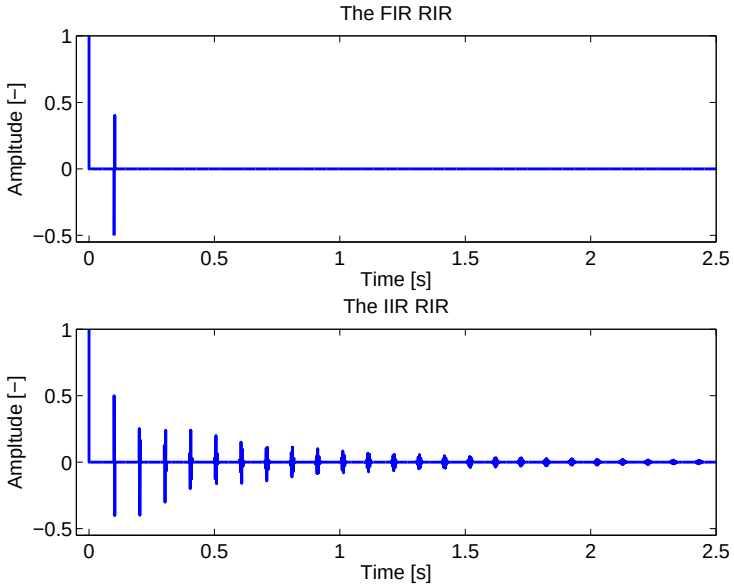


Figure 7.3: The two types of RIR used in the test for reduction of late reflections. The sampling frequency is set to 8 kHz.

The algorithms are only tested on the IIR RIR. The correct equalizing filter will therefore be a FIR. The results are shown in figure 7.4

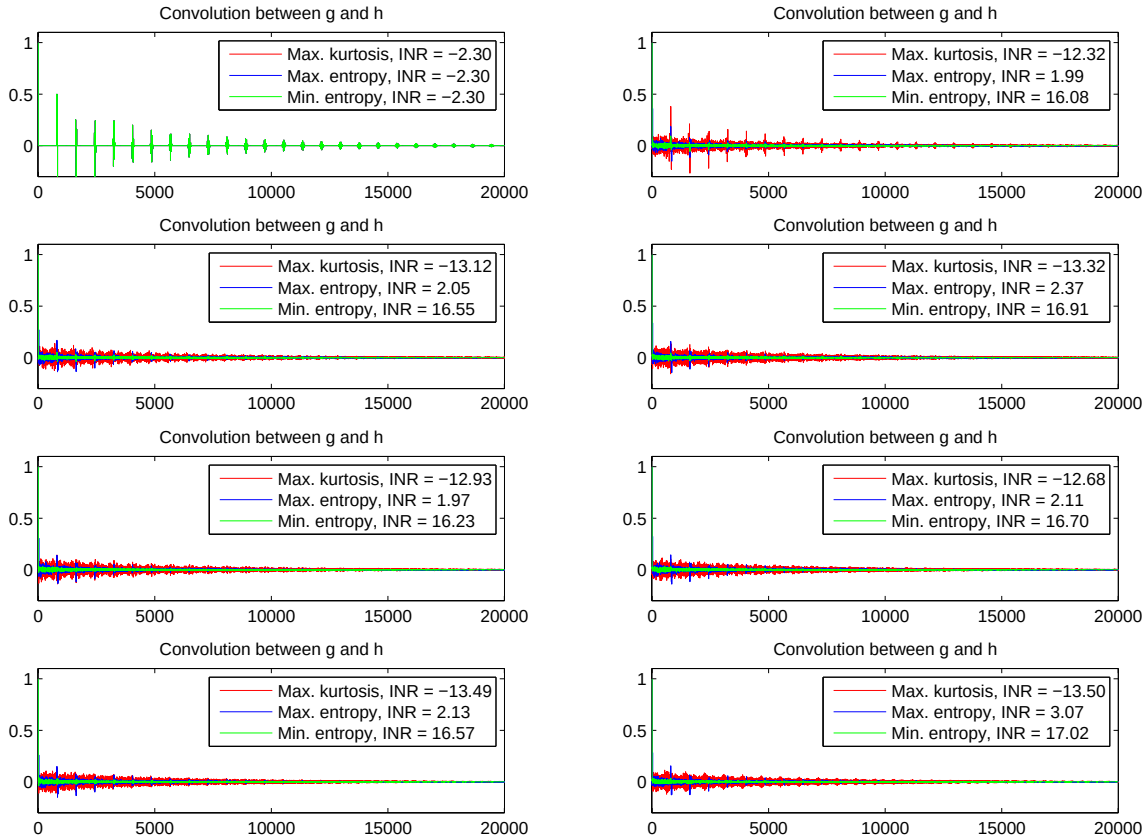


Figure 7.4: The convolution results for the three algorithms using the IIR RIR.

From the convolution results it is clear that the late reflections is a much harder task for the

algorithms as compared to the reduction of the early reflections. The kurtosis maximization algorithm cannot converge to the correct solution and the maximum entropy algorithm only performs modest improvements of a few dB. However, the minimum entropy algorithm performs well and is able to significantly reduce the late reflections and improve the INR with approximately 19 dB. In figure 7.5 a zoom of the last convolution result is shown.

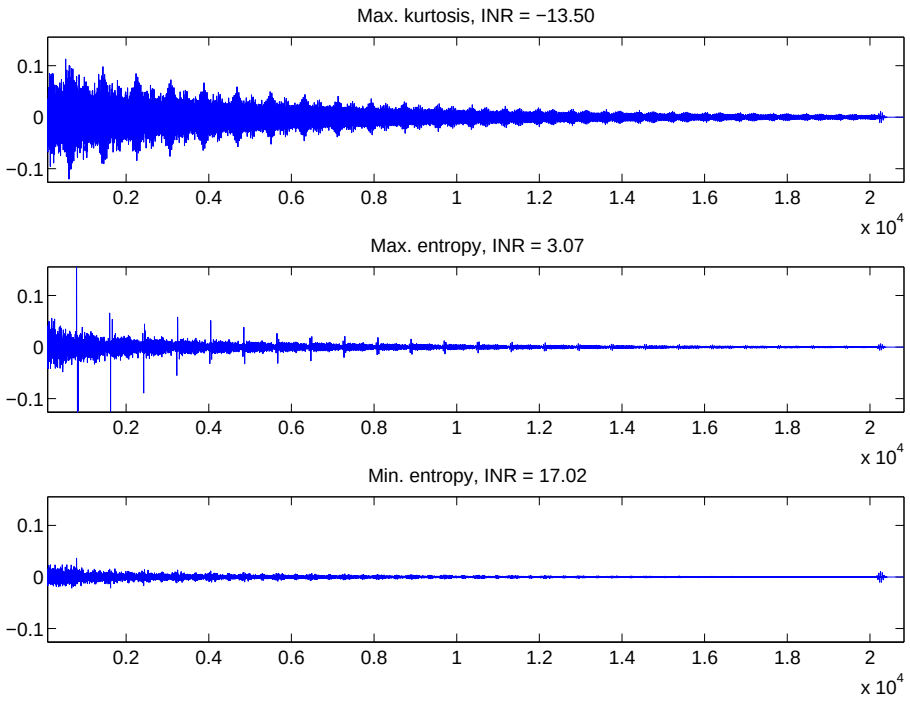


Figure 7.5: Zoom of the last convolution result for the three algorithms using the IIR RIR.

Figure 7.5 clearly shows that the convolution noise is smallest for the minimum entropy algorithm, but the figure also shows that the two other algorithms do not perform well with the IIR RIR.

7.4 Tests on measured RIR

In the previous sections the three approaches have been compared with respect to INR and convergence rate. Furthermore proof of concepts have been made in chapter 5 and 6. Based on the tests it is concluded that all three algorithms are functional and perform well under ideal conditions, except in the case of late reverberation removal, where only the minimum entropy algorithm performed well. Each algorithm has under these conditions shown to be interesting for further research. In this section the three algorithms will be tested on real life conditions, that is where a real, recorded RIR and real speech is used. The RIR is one of the RIRs measured by the project group in the facilities of the Section of Acoustics at Aalborg University, see appendix F. The source is the real speech, recorded in an anechoic chamber by [26]. The idea is that the RIR and speech reflect a real life situation, so that it can be evaluated how far the algorithms are from being able to work in a real life application such as a hearing aid. Therefore the complexity with respect to the number of multiplications needed

for each algorithm is also analysed in chapter 8. It should be mentioned that both entropy approaches have not been tested on complex RIRs in any previous work known by the project group.
In chapter 5 and 6 the algorithms were implemented using a normalization method, suggested by the project group, where the first filter coefficient h_0 is set to one. This method is also applied to the algorithms, when tests are made in this section. In [10] Erdogmus et al. suggested another scaling method for the minimum entropy algorithm, where the variance of $\mathbf{y}$ is subtracted from the differentiated cost function, as described in section 6.3. This scaling method is compared with the normalization, suggested by the project group, to determine which one is the most effective when the RIR is complex. Tests have already shown that the project group's normalization technique is superior for simple RIRs.

The RIR

The used RIR is RIR #1 in appendix F. The RIR was measured with a sampling frequency of 51.2 kHz. The speech recorded during the same measurement was also sampled with 51.2 kHz. To reduce the number of calculations the RIR and the speech were downsampled to 10.24 kHz. Hereafter an echogram, which is the logarithm of the absolute value of the RIR relative to the maximum value of the RIR, was made to find the taps that had an energy higher than -20 dB. The process can be seen in figure 7.6. The determined filter was then used for testing in the rest of this section.

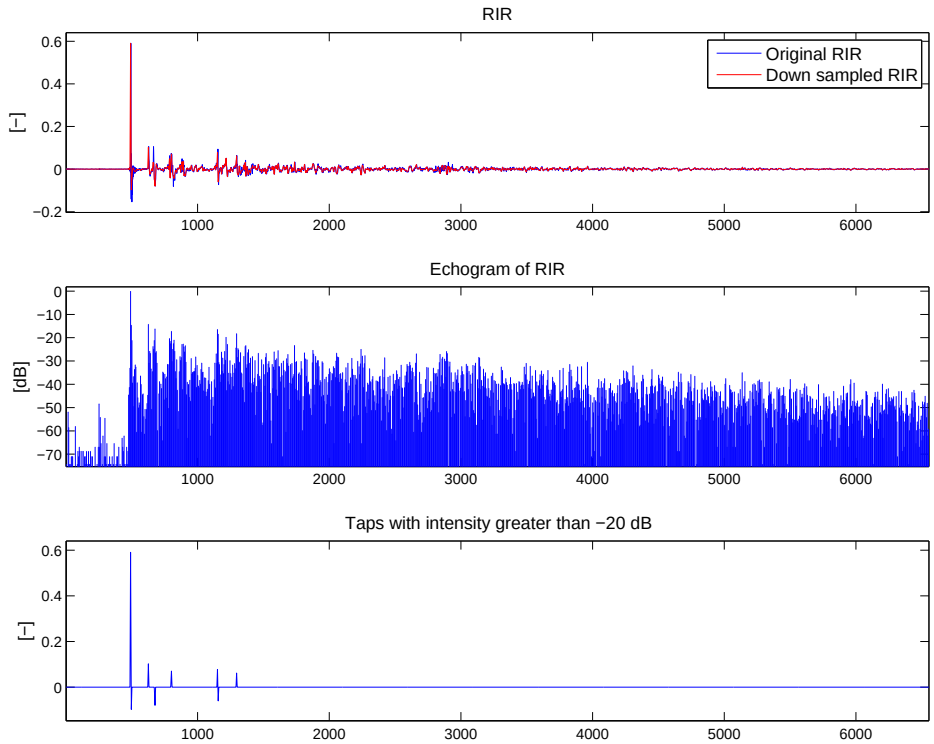


Figure 7.6: The RIR before and after downsampling to 10.24 kHz, the echogram, and the used RIR, where all taps with energy below -20 dB have been removed.

Before the algorithms are tested on the RIR in figure 7.6 and the recorded speech, various

parameters such as step size and filter length need to be determined. The selection is made based on tests, where the input signal is the iid samples drawn from the pdf based on $\cosh(\cdot)$, described in section 6.1.
First the inverse of the RIR is calculated using the FFT method described in section 3.1. The result is shown in figure 7.7 and based on the figure it is estimated that a filter length L in the range 350 to 500 will be sufficient. The remaining parameters were equal to the values, which were used when the simple tests were performed, that is $\beta = 0.99$, $N = 100$, and $\sigma = 0.1$. The M parameter in the maximum entropy has to be longer than or equal to L and it was selected to be 500.

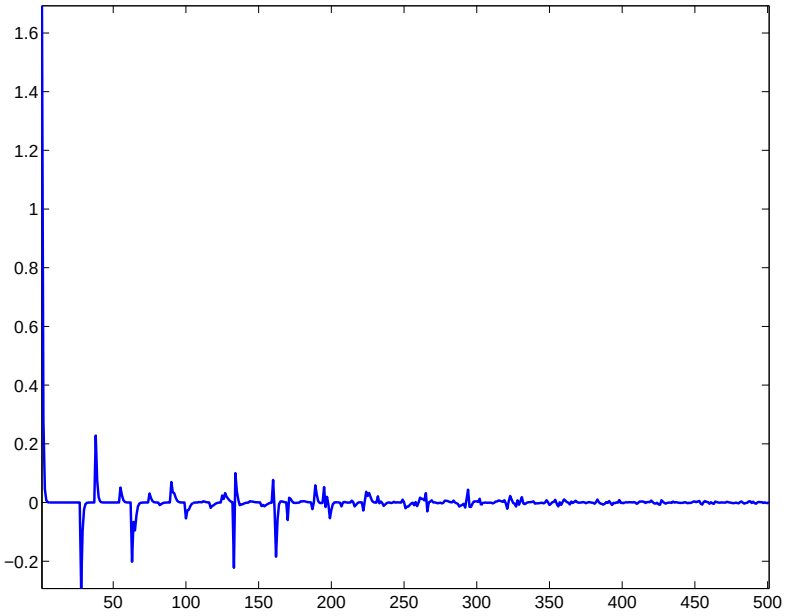


Figure 7.7: The inverse RIR.

Test on Iid Source Samples

Next the algorithms were tested using $L = 400$ and the iid input samples drawn from the pdf, which was made using the accept/reject sampling method in appendix B. The objective was to tune the step size to obtain the highest INR, and then use the selected step size values on filters with length 350, 450, and 500. The length of the input sequence when tuning the parameters was $3 \cdot Fs = 30720$ samples. The tests showed that the highest INR value is obtained for a filter length of 350 where the step size for kurtosis maximization is $2 \cdot 10^{-6}$, for maximum entropy it is $1 \cdot 10^{-6}$, and for minimum entropy it is $7 \cdot 10^{-5}$. A total of 100 Monte Carlo simulations were made with these settings. The convolutions between the RIR and the equalizer filters are shown in figure 7.8, where a sample length of $100 \cdot 10^3$ has been used.

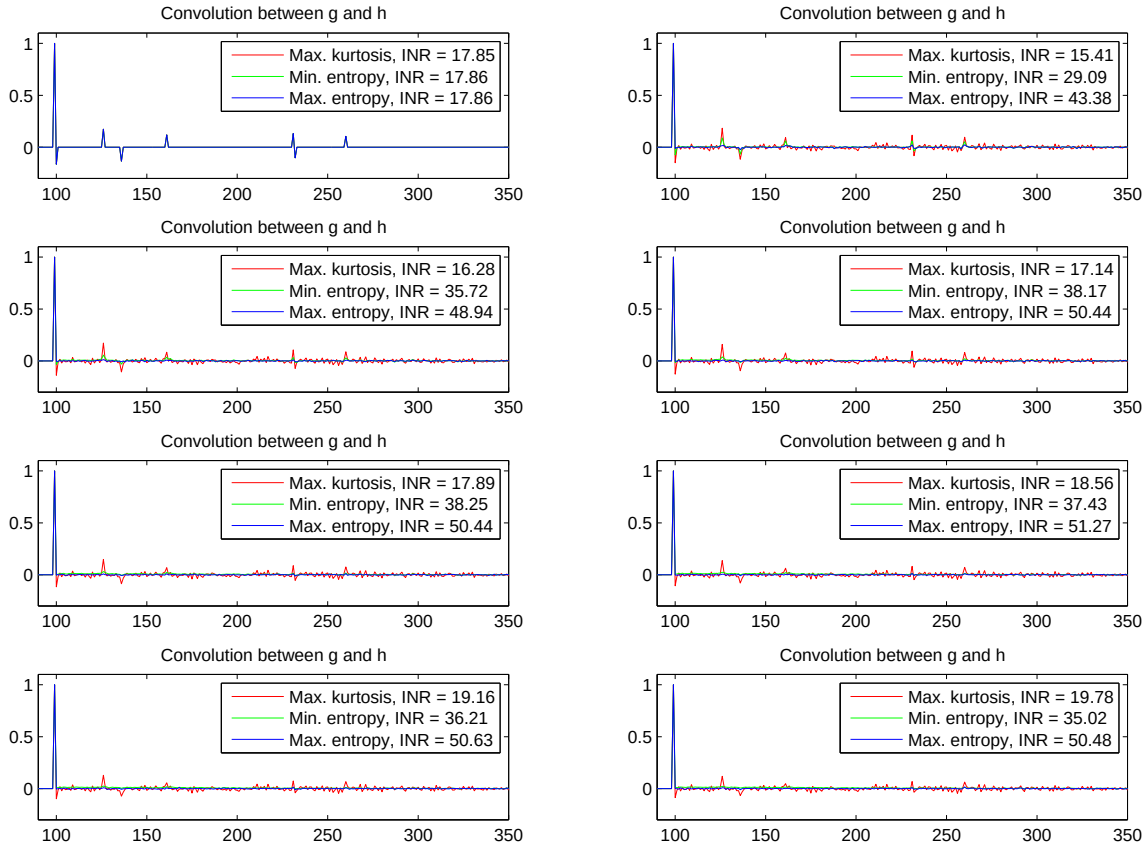
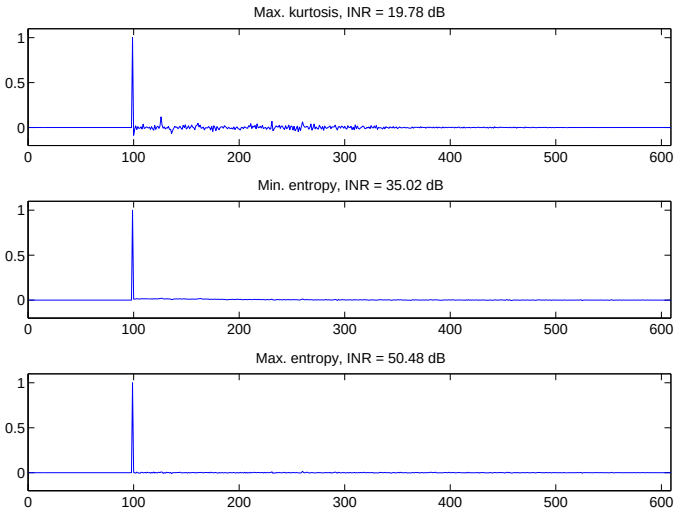
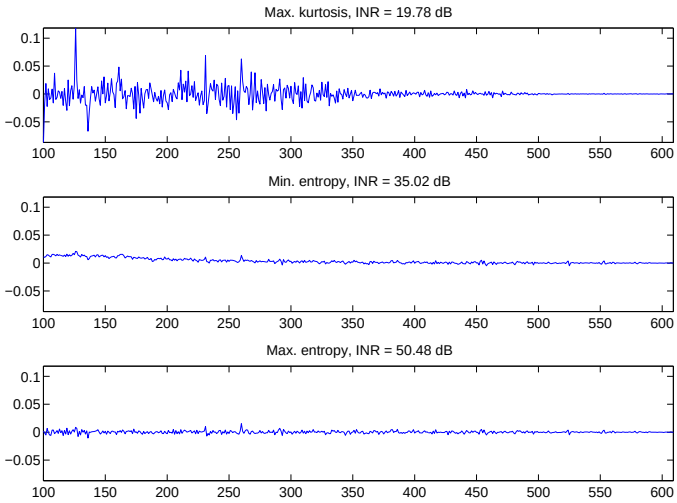


Figure 7.8: Convolution between the RIR and the equalizer for the three algorithms using $L = 350$ and the tuned step sizes. The upper left plot is calculated for the first sample and the lower right plot for the last sample. The plots in between are for six samples equally spaced between the first and last sample.

Notice that the first INR value is very high. The reason is that the filters are initialized with $\mathbf{h} = [1\,0\,0\,...\,0]$, which means that the first convolution is equal to the impulse response of the room, given in the lower plot in figure 7.6. Analysing figure 7.8 it is clear that the maximum entropy has the highest convergence rate and that it converges to the best solution. The kurtosis maximization converges much slower, which was also expected because the previous tests have shown the same relationship between the algorithms' convergence rate.
Figure 7.9(a) shows the final convolution between the RIR and the equalizer filters, and figure 7.9(b) is a zoom on the convolution noise for the same results.



(a) The final convolution.



(b) Zoom on the convolution noise in the final convolution.

Figure 7.9: The final convolution between the RIR and the equalizer.

Figure 7.9 clearly illustrates that the kurtosis maximization algorithm results in more convolution noise, hence a worse deconvolution.
The conclusion is that both entropy algorithms lead to significantly better results than the kurtosis maximization, when equalizing the measured RIR. The normalization method suggested by the project group was used in all tests.

Minimum Entropy Normalization Methods

Next the two different normalization approaches for minimum entropy are compared. The test was made on the $100 \cdot 10^3$ samples and for $L = 350$, where the effect of the h_0 normalization was evaluated. The result is shown in figure 7.10.

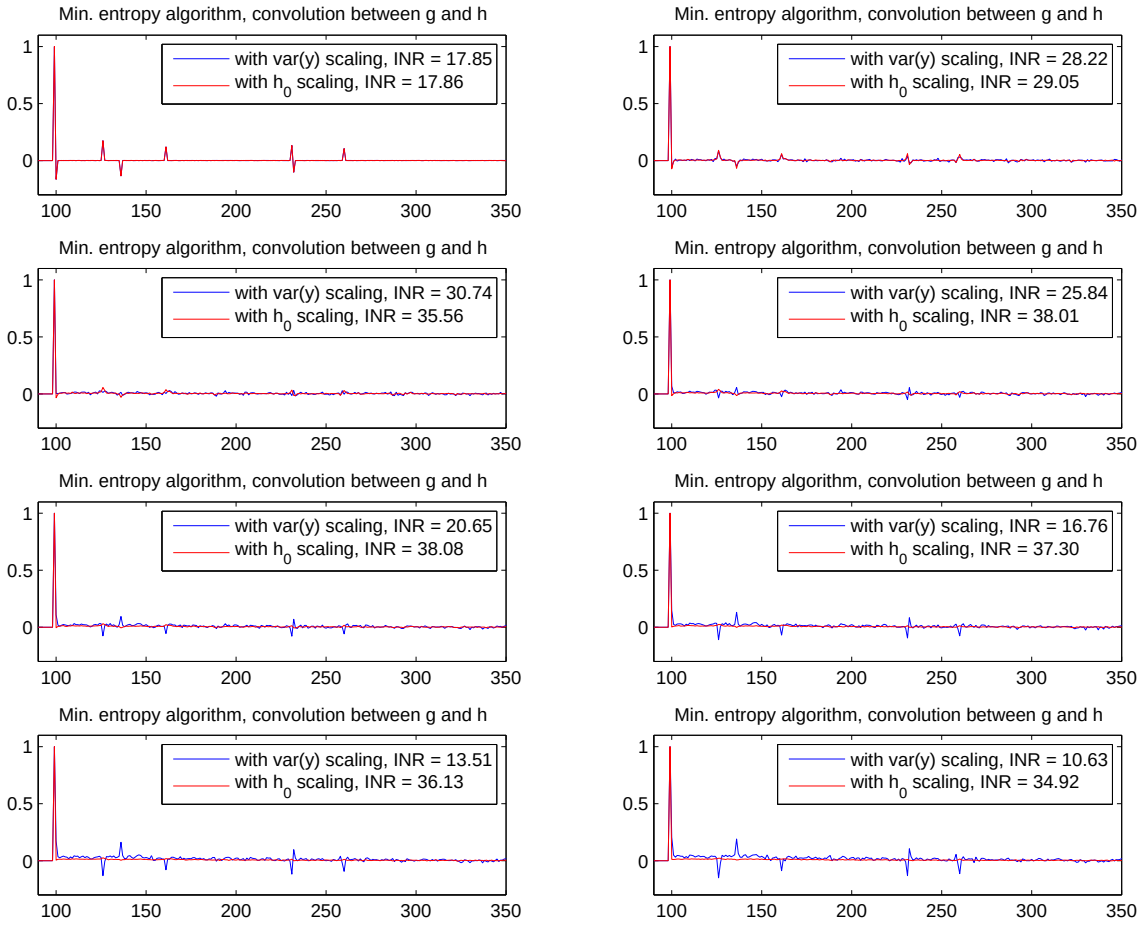


Figure 7.10: Convolution bewteeen the RIR and the equalizer, when using two different implementations of the minimum entropy algorithm.

The figure clearly illustrates that the minimum entropy algorithm performs much better, when the h_0 normalization proposed by the project group is applied. Using the variance scaling method, the minimum entropy algorithm does not converge at all. This is also shown in figure 7.11 which is a zoom on the last convolution in figure 7.10.

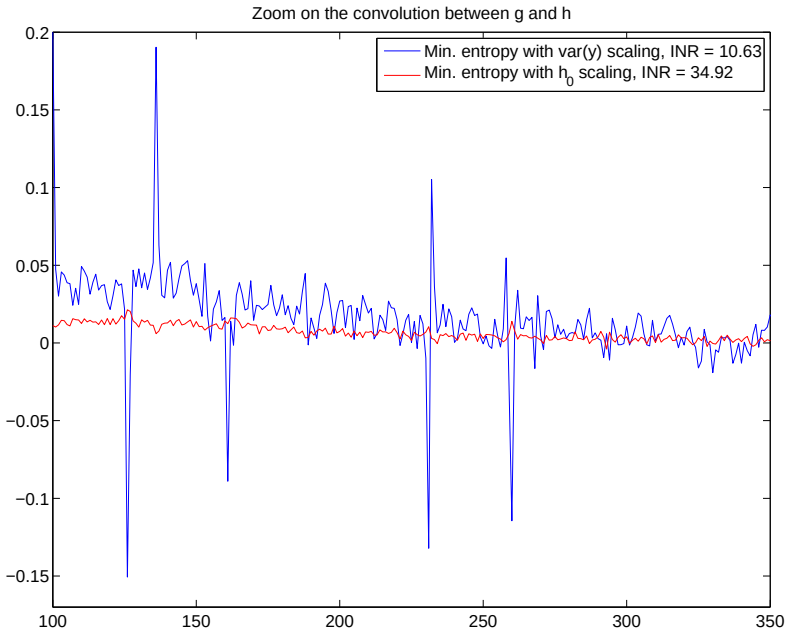


Figure 7.11: Zoom on the last convolution between the RIR and the equalizer, when using two different implementations of the minimum entropy algorithm.

Based on these results it is concluded that both entropy algorithms are able to deconvolve the real RIR. The maximum entropy has shown to be able to arrive very closely to the correct filter and furthermore it has a fast convergence rate.
The algorithms are now moved a step closer to the real life situation by using the recorded speech signal instead of the iid samples drawn from the pdf, described in section 6.1.

Real Voiced Speech

As mentioned in the introduction to this section the source speech signal was downsampled from 51.2 kHz to 10.24 kHz. After that the speech is divided into frames of 20 ms. The frame length of 20 ms entails that the speech can be considered quasi stationary. In a real time application the input to the algorithm can only be as long as the time it takes before the RIR must be updated. The project group has assessed that the RIR should be updated 25 times per second to provide a reasonably good estimate of the current RIR. This means that with a frame length of 20 ms only two frames can be processed for one RIR. A two frame sequence is clearly not enough for the algorithm to reach convergence, and the two frames are therefore reused one after another such that an update frame length of 10^5 samples is obtained for each input sequence.

The test performed on $100 \cdot 10^3$ samples using the three algorithms unfortunately showed that none of the algorithms converge. The reason why the algorithms do not work is considered to be because of the assumptions, which were that the input signal is white. This is not the case for the recorded speech, and the consequence is that the algorithms cannot distinguish between the non-white input signal and the colourisation added by the room. This means the that algorithms adjust the equalizer filters toward incorrect solutions that do not lead to deconvolution of the room.

To make the step from the iid samples drawn from the pdf to the recorded speech smaller the project group decided to test on the LP residuals of the recorded speech by using the LPC method on each 20 ms frame. The reason for using LP residuals as source is that the source is whitened, or ideally iid distributed. The pdf of the source then becomes more super Gaussian distributed as compared to the clean speech. A plot showing this is given in figure 7.12. The speech is male voice from [26].

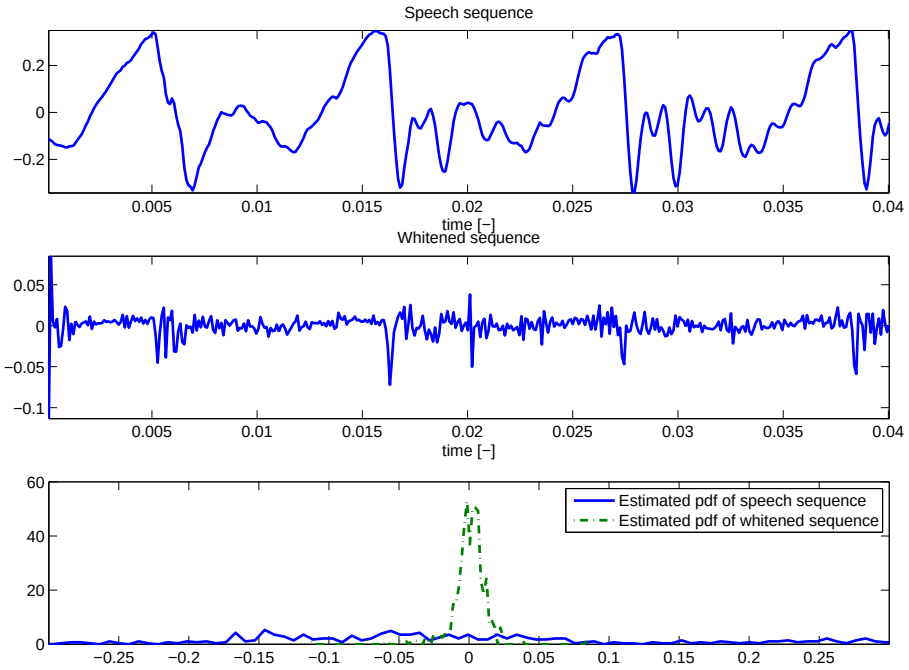


Figure 7.12: Example of whitened sequence using LPC. The sequence is two frames long.

The plot of the pdf shows that the process whitened using LPC has the desired effect on the speech signal. It should be mentioned that the whitening of the source signal is not possible in a blind signal processing approach.
Figure 7.13 shows eight convolutions between the real RIR and the equalizer, when the LP residuals are used as input.

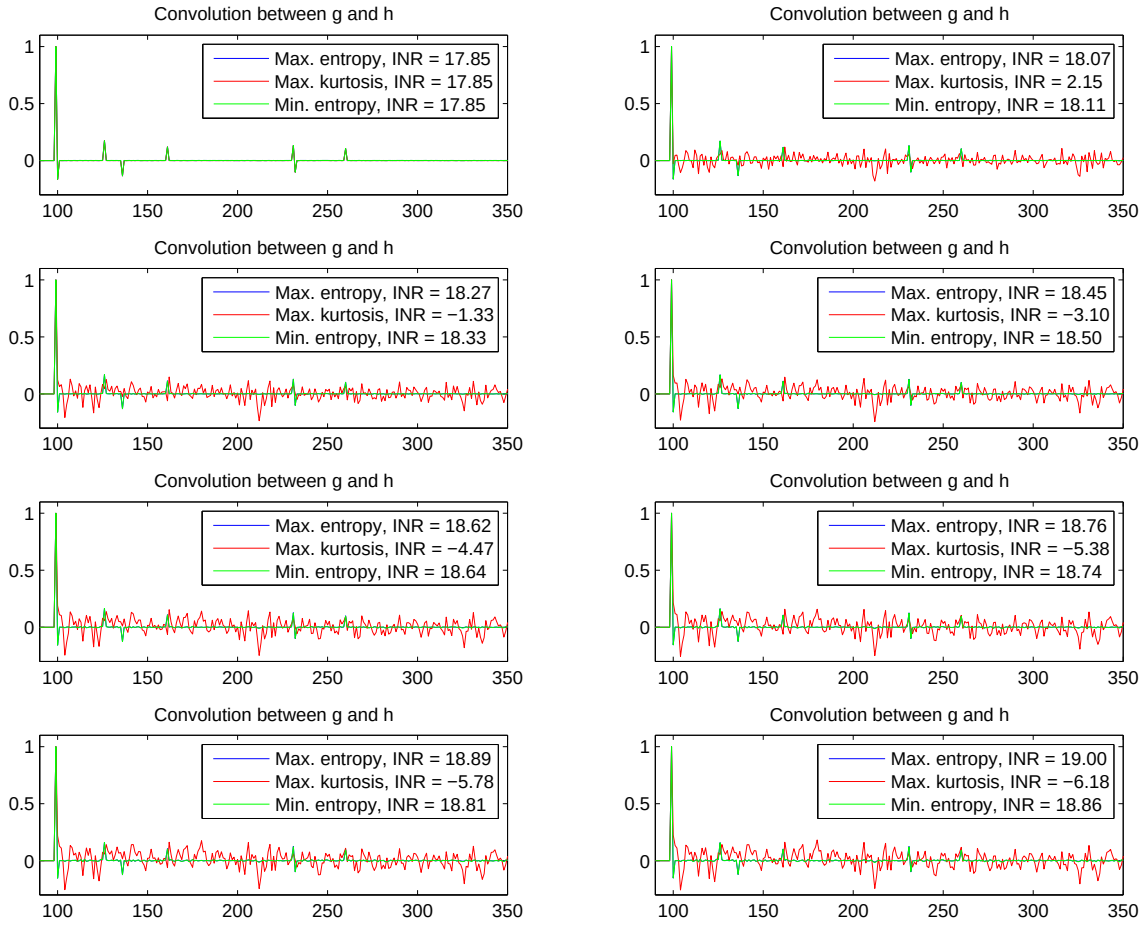


Figure 7.13: Convolution between the RIR and the equalizer when the input is LP residuals of the recorded speech.

The figure indicates that the maximum and minimum entropy algorithms only improve the INR with approximately 1 dB even though $100 \cdot 10^3$ samples have been used. The kurtosis maximization algorithm does not converge at all. The conclusion is that the algorithms require a large amount of adjustments before they can converge when the LP residuals are used. This entails that it requires even more adjustments for the algorithms to converge when the recorded speech is used as input. The reason why it is so difficult is that the assumptions regarding the source do not match the statistics of the used source sequence. This is an important topic for future work, especially the impact wrong assumptions have on the algorithms.

7.5 Summary

In this chapter the three algorithms were compared with respect to the INR performance metric and convergence rate.
First 100 Monte Carlo simulations were made with the 3 tap IIR RIR. The results showed that the maximum entropy algorithm results in the best INR, and furthermore it has the smallest bias of the three algorithms and a variance in the order of 10^{-5}. One hundred Monte Carlo simulations of the convergence rate also showed that the maximum entropy converges about twice as fast as the two other algorithms. Note however that this metric is very dependent on

the step size, but the maximum entropy was tested with the smallest step size.
In the test of the 16 tap IIR and 16 tap FIR the INR values differ less between the algorithms, and with an average INR around 53 dB and 29 dB, for the two RIRs respectively, it is concluded that the three algorithms are able to converge towards the correct equalizer filter. Therefore the algorithms were applied to more complex RIRs. First the algorithms ability to remove late reverberations, that is reflections which occur 100 ms or later after the direct speech, was tested. All previous tests focused on colourization, caused by early reflections, and therefore it was interesting to analyse a late reverberation RIR, which requires a much longer equalizer filter. The result is that the minimum entropy algorithm significantly reduces the late reflections for an IIR RIR. The kurtosis algorithm cannot converge at all, and the maximum entropy only improves the INR results slightly for the IIR RIR.
Finally the algorithms were tested on RIR #1, which is a RIR recorded by the project group, as described in appendix F. The RIR was modified using an echogram approach to simplify the task for the filters. First the iid source signal drawn from the $\cosh(\cdot)$ pdf, which has been used in all previous tests, was reverberated and the parameters, filter length and step size, were evaluated. A filter length equal to 350 leads to good results and therefore this length was used in 100 Monte Carlo simulations, which showed that the maximum entropy algorithm can deconvolve the RIR and obtain an INR $\approx$ 50 dB. The minimum entropy algorithm also converges, but not as well since the result is INR $\approx$ 35 dB. The kurtosis maximization algorithm converges very slowly and it improves the INR with less than 2 dB. All the algorithms utilized the h_0 normalization suggested by the project group. In [10] Erdogmus et al. suggested another normalization approach for the minimum entropy algorithm and it was compared to the result mentioned above. The conclusion is that the approach suggested in [10] leads to a non-converging algorithm and therefore the normalization method suggested by the project group is preferred.
To approach a real life situation the iid source signal was replaced with a real voiced speech signal, recorded by [26], in the test of the recorded RIR. However tests showed that none of the algorithms are able to converge with this source signal. The reason is that the assumptions regarding the source do not match the statistics of the voiced speech. It is therefore a topic for further work to investigate how this problem can be eliminated or how the effects on the algorithms's convergence capabilities can be reduced.

In the next chapter the complexity of the algorithms is analysed and the required number of multiplications is compared with the performance of a modern digital signal processor.

8

Complexity of the Algorithms

In this chapter the complexity of the three algorithms, kurtosis maximization, maximum entropy, and minimum entropy in section 5.1, 6.1, and 6.3 respectively, is considered.
In the project the focus is on the functionality, ie. the algorithm's ability to determine the correct inverse filter, convergence speed, bias and variance of the coefficients etc., and not on the complexity which in this report is limited to a brief analysis of the computational complexity. To further simplify the analysis the computational complexity is limited only to account for the number of multiplications each algorithm requires. The required number of multiplications is then compared with a state-of-the-art digital signal processor (DSP), in the sense that the maximum number of multiplications the selected DSP can perform is determined and compared with the number of multiplications the algorithms require to deconvolve the measured RIR, which was used on the algorithms in section 7.4. This means that additions, subtractions, all interaction with memory and converters and so forth is not included in the analysis of how well the algorithms can perform on the DSP. Furthermore specific functions in the DSP are not utilized, e.g. the multiply accumulate instruction which is very useful in filtering operations. That is the estimate is very rough but it gives an idea of how big the gap is between the current implementation of the algorithm and a real time implementation.

Before the analysis of the algorithms is made it is noted that they could be optimized. E.g. by using precalculation of often used variables, using look-up tables to store trigonometric functions, and optimizing with regards to parallelism, because all three algorithms utilize sums which can be calculated in parallel. It will however require a processor, which is able to perform the calculations in parallel. Using parallel computation will help increase the execution speed significantly. Execution speed is critical because the application is a hearing aid, where real time execution is required.

When the algorithms are to be executed they will need to update a certain number of times per second, to ensure that the estimate of the RIR is in agreement with the room, where the user of the hearing aid is located. In section 7.4 is was assessed that the equalizer filter needs to be updated 25 times per second. In the following this type update is referred to as *EQ update*. Furthermore the algorithms will need to process a certain number of samples per

EQ update to ensure that they reach convergence. This type of update is called *algorithm update*. The number of samples is dependent on the algorithm, but in this chapter the number is assumed to be the same for the minimum entropy and kurtosis maximization algorithms, and half the number for maximum entropy. This assessment is based on the test in section 7.2. The samples will be based on a set of consecutive frames, whose length is determined by the quasi stationary limitations.
After each EQ update of the filter it will be advantageous to reuse the determined filter as the initial filter in the next EQ update, that is perform an iterative update. This will entail that the number of required samples before convergence is reached, can be reduced dramatically because the starting point for the algorithm is closer to the inverse filter than $\mathbf{h} = [1\,0\,0\,...\,0]$ is. In the tests performed in this report only one EQ update has been made and therefore it was not possible to take advantage of a previously estimated filter. This means that the obtained convergence rate, hence required number of samples before convergence, approximates the worst case situation, where the initial filter is very far from the equalizer filter.

In the next three sections a multiplication count is made for each of the three algorithms. The number of multiplications required to perform an algorithm update once, are determined.

8.1 Multiplication Count for Kurtosis Maximization

The kurtosis maximization algorithm can be divided into four separate calculations defined in equation (5.8), (5.9), (5.10), and (5.11). Furthermore it is also necessary to calculate the output sample $y(n)$ of the equalizer filter once per algorithm update. In the following the number of multiplications required to compute each of the equations is determined.
The filtering resulting in $y(n)$ is defined as

$$y(n) = \mathbf{h}^T(n)\mathbf{x}(n) \tag{8.1}$$

The number of taps in the filter $\mathbf{h}(n)$ is equal to L and therefore equation (8.1) requires L multiplications to be computed. As mentioned earlier the number of required additions is not used in this simple comparison of the algorithms.
Equation (5.8) is used to update the filter and it is given as

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu f(n)\mathbf{x}(n) \tag{8.2}$$

The scaling of the feedback function $f(n)$ that is the calculation $\mu \cdot f(n)$ requires one multiplication, and multiplying this result with the input vector $\mathbf{x}(n)$ requires L multiplications because the vector is L long.
The feedback function in equation (5.9) is given by

$$f(n) = \frac{4\left(\mathrm{E}[y^2(n)]y^2(n) - \mathrm{E}[y^4(n)]\right)y(n)}{\mathrm{E}^3[y^2(n)]} \tag{8.3}$$

The squaring of $y(n)$ requires one multiplication, and because the multiplication with the constant 4 can be included into the step size μ in equation (8.2) the nominator can be calculated using three multiplications. The denominator requires two multiplications and because it is assumed that a division requires the same number of cycles as a multiplication, even though it is a rough approximation, the total number of multiplications is six for equation (8.3).
The first expectation estimate, defined in equation (5.10), is

$$\mathrm{E}[y^2(n)] = \beta\mathrm{E}[y^2(n-1)] + (1-\beta)y^2(n) \tag{8.4}$$

and it requires two multiplications, because the squaring of y has already been made in equation (8.3), and the latter estimate defined in equation (5.11) given as

$$\mathrm{E}[y^4(n)] = \beta \mathrm{E}[y^4(n-1)] + (1-\beta)y^4(n) \tag{8.5}$$

requires three multiplications, because $y^4(n) = y^2(n) \cdot y^2(n)$, where $y^2(n)$ has been precalculated.
Summing the multiplication counts for each of the five equations it is concluded that the kurtosis maximization algorithm requires

$$\mathrm{MaxKurt}_{\mathrm{mult}} = (L) + (L+1) + (6) + (2) + (3) = 2L + 12 \tag{8.6}$$

multiplications to calculate and update the filter. Each of the parentheses represents one of the equations (8.1), (8.2), (8.3), (8.4), and (8.5), respectively. Using the O-notation the complexity is $O(L)$.

8.2 Multiplication Count for Maximum Entropy

The maximum entropy algorithm is defined by the three equations (5.8), (6.19), and (6.20). This algorithm requires M values of y before it can run and using the result from the previous section it means that $M \cdot L$ multiplications are required to compute $y(k), k = \{1, 2, ..., M\}$.
The required number of multiplications for the steepest ascent update step given as

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu \Delta \mathbf{h}(n) \tag{8.7}$$

is equal to L, because the vector $\Delta \mathbf{h}(n)$ is L long. Equation (6.19) is examined next

$$\Delta h_0 = M \cdot \frac{1}{h_0} - 2a \sum_{i=1}^{M} \tanh(a \cdot y(i)) \cdot x(i) \tag{8.8}$$

$$\tag{8.9}$$

It is assumed that the hyperbolic tangent $\tanh(\cdot)$ is implemented in a lookup table (LUT) to speed up the algorithm. Furthermore the multiplication with $-2a$ can be made before the result of $\tanh(\cdot)$ is stored in the LUT and thereby even more calculations are saved.
Each part of the sum requires two multiplications, one for $a \cdot y(i)$ and one for $\tanh(a \cdot y(i)) \cdot x(i)$. The summation sums M parts and therefore the total number of multiplications is $2M + 1$, where the last multiplications origins from the $\frac{M}{h_0}$ division.
The last part of the algorithm calculates $L - 1$ filter coefficients using equation (6.20) defined as

$$\Delta h_{\mathrm{j}+1} = -2a \sum_{i=j+1}^{M-1} \tanh(a \cdot y(i)) x(i-j) \tag{8.10}$$

Again each part of the sum requires two multiplications, but this time the size of the sum depends on the filter coefficient number

$$\begin{aligned} h_1 &\Rightarrow \sum_{i=1}^{M-1} \Rightarrow 2 \cdot (M-1) \text{ multiplications} \\ h_2 &\Rightarrow \sum_{i=2}^{M-1} \Rightarrow 2 \cdot (M-2) \text{ multiplications} \\ &\vdots \\ h_{L-1} &\Rightarrow \sum_{i=L-1}^{M-1} \Rightarrow 2 \cdot (M-(L-1)) \text{ multiplications} \end{aligned}$$

The total number of multiplications tot_{mult} is

$$\begin{aligned} tot_{\text{mult}} &= 2 \cdot (M-1) + 2 \cdot (M-2) + ... + 2 \cdot (M-(L-1)) \\ &= (L-1) \cdot 2M + 2 \cdot (-1-2-...-(L-1)) = (L-1) \cdot 2M - 2 \sum_{i=1}^{L-1} i \\ &= (L-1) \cdot 2M - 2 \cdot \left(\frac{L^2 - L}{2} \right) \\ &= 2ML - 2M - L^2 + L \end{aligned} \tag{8.11}$$

where the arithmetic rule $\sum_{i=a}^{b} i = \frac{(b-a+1)(b+a)}{2}$ has been used.
The total number of multiplications required by the maximum entropy algorithm $\text{MaxEnt}_{\text{mult}}$ is

$$\begin{aligned} \text{MaxEnt}_{\text{mult}} &= (L) + (1+2M) + (2ML - 2M - L^2 + L) + (ML) \\ &= L(3M - L) + 2L + 1 \end{aligned} \tag{8.12}$$

Each of the parentheses represents one of the initial equations (8.7), (6.19), and (6.20) and the filtering operation. M is larger than or equal to L hence the complexity is $O(ML)$.

8.3 Multiplication Count for Minimum Entropy

The minimum entropy algorithm is based on two equations. The steepest ascent algorithm in equation (5.8), where the sign has been changed so the algorithm becomes a steepest descent, and the differentiated cost function in equation (6.40). The steepest descent algorithm is given as

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu \frac{\partial J(\mathbf{h}(n))}{\partial \mathbf{h}} \tag{8.13}$$

It has already been determined that the steepest ascent/descent requires L multiplications, but in the minimum entropy algorithm it is possible to multiply with the step μ during the calculation of the differentiated cost function, hence L multiplications can be saved. Before the algorithm can run it is necessary to compute N y-values, where N is the window size. That calculation requires $N \cdot L$ multiplications.

The cost function is given in equation (6.40) and repeated here for convenience

$$\frac{\partial J(h(k))}{\partial \mathbf{h}} = -\frac{\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1} -\frac{1}{\sqrt{2\pi\sigma^3}}(y_j - y_i)\exp\left(-\frac{1}{2\sigma^2}(y_j - y_i)^2\right)(\mathbf{x}_j - \mathbf{x}_i)\right)}{\sum_{j=k}^{N+k-1}\left(\sum_{i=k}^{N+k-1}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2\sigma^2}(y_j - y_i)^2\right)\right)} \tag{8.14}$$

$$= \frac{\frac{1}{\sigma^2}\sum_{j=k}^{N+k-1}\sum_{i=k}^{N+k-1}\text{ydiff}\cdot\text{expy}\cdot\text{xdiff}}{\sum_{j=k}^{N+k-1}\sum_{i=k}^{N+k-1}\text{expy}} \tag{8.15}$$

where: ydiff equals $y_j - y_i$
expy equals $\exp\left(-\frac{1}{2\sigma^2}(y_j - y_i)^2\right)$
xdiff equals $\mathbf{x}_j - \mathbf{x}_i$

The variable ydiff is simply a subtraction, i.e. it does not require any multiplications.
The exponential operator in expy is assumed to be stored in a LUT, where the constants $\frac{1}{\sigma^2}$ and μ are multiplied with the exponential operator to save multiplications. The calculation of the argument to expy requires 2 multiplications, that is the squaring of ydiff requires one and the multiplication with the constant $\frac{-1}{2\sigma^2}$ requires another.
It does not require any multiplications to calculate xdiff but multiplying xdiff with expy requires L multiplications because the vector xdiff is L long. In total this means that $L + 3$ multiplications are required to compute the nominator. The only variable in the denominator is expy, which already has been calculated in the nominator. Finally the division between nominator and denominator adds one multiplication to the total.
Each sum in (8.15) works on N parts, which means that the multiplications in the nominator have to be made N^2 times. This means that the minimum entropy algorithm requires

$$\text{MinEnt}_{\text{mult}} = (N \cdot L) + (N^2(L + 3) + 1) \tag{8.16}$$

where the multiplications in the first parentheses originates from the calculation of the output samples y and the latter is the required multiplications from equation (8.15). The complexity is $O(N^2L)$.

8.4 Comparison with a Digital Signal Processor

In this section the number of multiplications, which the three algorithms require, is compared with the capabilities of a state-of-the-art floating point DSP. The idea is to give a first impression of how well the algorithms are suited for implementation. It is however important to note that the algorithms have not been optimized in any way, because the focus in the project is on the functionality and not the complexity of the algorithms.

According to section 7.4 the algorithms will need to EQ update 25 times per second to ensure that the estimated equalizer filter fits the current RIR. Furthermore it was determined, based on 100 Monte Carlo simulations presented in section 7.2, that the minimum entropy and kurtosis maximization algorithms require in the order of $100 \cdot 10^3$ samples to converge. The maximum entropy algorithm converges faster and it therefore only requires $50 \cdot 10^3$ samples. That is the DSP will need to evaluate the algorithms on $25 \cdot 50 \cdot 10^3 = 1.25 \cdot 10^6$ and $25 \cdot 100 \cdot 10^3 = 2.5 \cdot 10^6$ samples per second for the maximum entropy and the two other algorithms, respectively. Note that when the algorithm is running in a real implementation

the required number of samples will probably be reduced, because the algorithm will be able to reuse the determined filter from the previous EQ update, as the initial filter in the new EQ update.
The project group selected a modern floating-point DSP from Texas Instruments. The processor is a member of the TMS320 C6000 floating-point DSP family, which is able to calculate 32 bit single precision values and 64 bit double precision values. The selected DSP is clocked at 350 MHz and the model name is *TMS320C6727B-350*, [33]. The clock frequency means that the DSP can execute $350 \cdot 10^6$ one cycle instructions per second, because the processors power to pipeline the calculations is not considered.
Since the complexity analysis in the previous sections focused on the multiplications, the DSP's ability to perform a single-precision multiplication is analysed. If the algorithms were to be implemented many other functions such as add/subtract, read/write, receive and decode interrupts, receive and store incoming samples, output samples to a digital to analog converter and so forth would have to be considered. That is it will be necessary for the DSP to execute a lot more than just multiplication instructions.
According to the instruction set manual [34] the *TMS320C6727B-350* can multiply two single-precision floating-point values as a four-cycle instruction using the *MPYSP* instruction. The fact that the instruction is a four-cycle one means that it can be divided into four cycles. In the first cycle the DSP will read the two values, which are to be multiplied and start the calculation. In the following two cycles the DSP continues the calculation, and in the fourth and final cycle the DSP will finish the calculation and write the results to the destination register. The four-cycle instruction means that the DSP can perform

$$N_{\text{mult}} = \frac{350 \cdot 10^6 \text{ cycles per second}}{4 \text{ cycles per multiplication}} = 87.5 \cdot 10^6 \text{ multiplications per second} \tag{8.17}$$

Comparing this with the fact that it is necessary to process $1.25 \cdot 10^6$ and $2.5 \cdot 10^6$ samples per second it means that the DSP is able to perform 35 and 70 multiplications per algorithm update, respectively.
In section 7.4 it was determined that a filter length $L = 350$, a windows size of $N = 100$, and $M = 500$ was required to deconvolve the measured RIR, which is described in F. Inserting these values in equation 8.6, and 8.12, and 8.16, table 8.1 was made.

	Kurtosis maximization	Maximum entropy	Minimum entropy
Multiplications per algorithm update	$2L + 12$	$L(3M - L) + 2L + 1$	$(N \cdot L) + N^2(L + 3) + 1$
Using determined L, N, and M	712 multiplications	403201 multiplications	3565001 multiplications
Approximate real time factor	20.3	$5.76 \cdot 10^3$	$1.02 \cdot 10^5$

Table 8.1: Multiplication count for the algorithms per algorithm update, and when deconvolving the measured RIR. The real time factor is the number of multiplications per algorithm update divided with the possible number of multiplications in the DSP. The real time factor should be equal to one or less if it the algorithms were to be executed in real time.

Examing the values in the last row of table 8.1 it is clear that the kurtosis maximization algorithm is much faster than the entropy algorithms. Even though the kurtosis maximization algorithm requires twice as many samples as the maximum entropy algorithm it is almost 300 times closer to real time. The minimum entropy is even slower, since it's real time factor is

roughly 18 times larger than the real time factor of the maximum entropy algorithm.
Based on the crude estimation of the required number of multiplications and the corresponding real time factor it is concluded that the three algorithms need optimization, before they can be executed in real time, as required by many applications such as a hearing aid. Further work could be made with regards to increase the use of inherent parallelism, which also will lead to the requirement for a parallel processor(s), and to reduce the number of required multiplications.
Note however that the required number of samples before convergence will be drastically reduced, when the algorithms are implemented in an application, where the filter is iterated, meaning that the initial filter will be based on the filter from the previous EQ update.

This was the final chapter of the algorithm analysis and implementation part of the report. In the next chapter the conclusion on the project is given and in the following chapter ideas for future work, including a novel algorithm suggested by the group, is presented.

9 Conclusion

In this project different approaches to the blind speech dereverberation problem were analysed. Reverberation in speech, caused by room reflections, is problematic especially for hearing impaired people, and it is therefore an important research area. The three dereverberation methods presented in this project are based on higher-order statistics (HOS) and information theory. The application is a single microphone setup. These methods were tested on simple and real life room impulse responses (RIRs) with self generated samples and real voiced speech.

The three methods are all based on the same assumptions. The source signal (the direct speech), $\mathbf{s}$, is assumed to consist of independent and identically distributed (iid) samples, and to be non-Gaussian distributed. The room then causes the observed signal (the reverberated speech), $\mathbf{x}$, to be corrupted by delayed and weighted versions of $\mathbf{s}$, and by the central limit theorem (CLT) the observed signal is assumed to be Gaussian distributed.

The HOS approach is based on maximizing the fourth order cumulant, called the kurtosis, of the signal. The room causes the signal to change from a non-Gaussian to a Gaussian distribution. All Gaussian signals have HOS cumulants equal to zero, and the approach is therefore to maximize the kurtosis of the observed signal, such that the room impact on the source signal will be removed. The implemented algorithm is based on the work by Gillespie et al., [6], who presented a steepest ascent algorithm for the problem. The project group experienced convergence problems with the algorithm, and based on an investigation of the performance surface it was suggested to remove the linear prediction analysis in the algorithm, such that the algorithm operates directly on the observed speech. Furthermore the project group suggested that the first coefficient of the equalizing filter is held fixed. Both suggestions improve the convergence abilities and furthermore the sensitivity to the step size is reduced. However the project group observed that the kurtosis value of the source signal, $K(\mathbf{s})$, is not an upper bound to the kurtosis value of the processed signal, $K(\mathbf{y})$. This entails that the condition $K(\mathbf{s}) = K(\mathbf{y})$ for perfect deconvolution is necessary but not sufficient. Furthermore this observation by the project group complicates the complex performance surface for the kurtosis approach which is already known to suffer from many local maxima.

The information theoretic approach to blind speech dereverberation was based on two methods: maximum entropy and minimum entropy. The room's impact on the source signal can from an information theoretic point of view be viewed as adding redundant information to the signal. The objective is therefore to reduce the statistical dependency in time between samples.
The maximum entropy method was based on the work by Bell et al., [9]. Through a nonlinear transformation of the processed signal, the higher order terms of the bounded output was weighted such that maximum entropy was obtained for the output, where it is known that maximum entropy entails minimum mutual information for bounded inputs. A steepest ascent algorithm was implemented for the maximum entropy approach. Bell et al. suggested for further research that the chosen nonlinearity function should match the cdf of the source. The project group therefore presented a flexible nonlinearity function based on the $\tanh(a \cdot u)$, where the argument u is multiplied with a constant a, such that the slope of the nonlinearity function matches the cdf of speech. This proposed cdf and matching pdf were found suitable for speech by the project group and therefore used as source signal when testing all implemented algorithms. The fact that the nonlinearity is chosen based on a priori knowledge of the source signal, reduces the applicability of the algorithm.
The implemented algorithm for the minimum entropy approach was based on [10] and [27]. A steepest descent algorithm was presented for the approach. It can be shown that for an unbounded input with a fixed variance the maximum entropy will occur for a zero-mean Gaussian signal. By the assumption that the observed signal is Gaussian, the algorithm therefore minimizes the entropy of the processed signal to make the output non-Gaussian. Renyi's entropy of order two was used because of its usability in an iterative algorithm. The pdf of the processed signal, $\mathbf{y}$, was required by the entropy calculation. Non-parametric density estimation based on a Parzen window utilizing a Gaussian kernel was therefore used to estimate the pdf of the signal. The minimum entropy approach does not require any a priori knowledge of the source as compared to the maximum entropy approach, and it makes it more widely applicable. It is necessary to constrain the algorithm from approaching the trivial zero solution, which is a zero-valued output resulting in Renyi's entropy approaching minus infinity. Erdogmus et al. suggest a cost function that is based on the variance of the processed signal. However the project group came across convergence ability problems with this solution. The project group therefore suggests that the first filter coefficient of the equalizing filter is kept fixed as in the kurtosis approach. This solution solves the convergence problem and saves the computations needed for calculating the variance.
It was shown that the entropy of the processed signal, $H(\mathbf{y})$, is an upper bound to the entropy of the source signal, $H(\mathbf{s})$, that is $H(\mathbf{y}) \geq H(\mathbf{s})$. This important proof restraints the algorithm from suffering of the same problem as the kurtosis approach. Hence the algorithm cannot approach a too small entropy. The major disadvantage of the minimum entropy approach is however that the algorithm has many parameters that are user controlled, and which the deconvolution is sensitive to.

The three implemented algorithms were tested with regards to impulse-to-noise ratio (INR) and convergence rate. The property of the INR is that it equals infinity for an equalization corresponding to a pure delay. The tests were first made upon simplified room impulse responses defined by the project group. The source speech samples were sampled from the pdf based on the nonlinearity function described in the maximum entropy approach, using the accept/reject sampling method. One hundred Monte Carlo simulations were made. The test results show that the three algorithms can reduce the reverberations and converge towards the correct equalizing filter. The maximum entropy approach performed the best on the 3 tap IIR RIR and 16 tap FIR RIR, where the minimum entropy approach performed the best for the

16 tap IIR RIR. For all tests the maximum entropy approach required the smallest amount of input samples to reach convergence, and for the 3 tap IIR RIR it was able to converge with the smallest bias and a variance in the order of 10^{-5}.

The algorithms were also tested upon rooms causing late reflections, which were defined to occur 100 ms or later after the direct speech. This test had the objective to show if the algorithms could perform well under complex but controlled conditions. Only the minimum entropy algorithm were able to significantly reduce the late reflections caused by a IIR RIR. The maximum entropy algorithm made modest improvements whereas the kurtosis algorithm did not converge and the resulting equalizing filter obtained worse results than the initial filter. As described in section 1.1 Tonelli et al. suggests in [8] that longer equalizing filters should be investigated for removing late reverberation. However this is not suggested by the project group. The test results show that the benefit of a longer filter, such that late reflections can be reduced, will not always exceed the cost of using more filter coefficients, which will add extra convolution noise.

Finally tests were made on a real life RIR, which the project group measured at the Section of Acoustics, Aalborg University, in a setup, which simulates a spartan living room. Tests were made both with real speech and the signal sampled from the proposed pdf. These test gives an useful insight in how well the algorithms perform under real life conditions. For the tests based on the proposed pdf, the three algorithms are able to reduce the reverberation. Again the maximum entropy algorithm performs the best followed by the minimum entropy algorithm. The comparison was based on 100 Monte Carlo simulations. For the minimum entropy algorithm the suggested constraining method by fixing the first filter coefficient of the equalizing filter showed to be superior to the variance constraint suggested in [10]. The algorithm based on the latter constraint could not converge at all.
The algorithms were also tested on real voiced speech, but they did not converge. The reason is that the assumptions regarding the source signal do not match the statistics of the real voiced speech.

To evaluate whether the algorithms can run in real time, the number of multiplications each algorithm requires was counted and compared with the possible number of multiplications in a state-of-the-art digital signal processor (DSP). The DSP is the floating-point TMS320C6727B-350 from Texas Instruments. Assuming that the equalizer filter needs to be updated 25 times per second it is concluded that none of the algorithms are capable of running in real time. The kurtosis maximization, maximum entropy and minimum entropy are 20.3, $5.76 \cdot 10^3$, and $1.02 \cdot 10^5$ times from real time, respectively. It should however be noted that the algorithms have not been optimized to run on the DSP, using special beneficial parts of the instruction set. However, the scope of this project is the functionality and not the complexity of the methods. Future work will obviously need to be made with regards to a real time implementation. In the next chapter the project group's suggestions for future work is presented and the focus is on a novel algorithm.

In conclusion, three interesting methods to blind speech dereverberation have been analysed in detail and implemented. Adjustments have been made by the project group based on carefully performed tests and the final result is that the three algorithms are able to significantly reduce the undesired reverberation in speech caused by a living room. The adjustments made by the project group improved the performance of the algorithms. The tests show that the entropy algorithms seem to be superior to the kurtosis maximization algorithm.

10

Future Work

The project group has made several considerations for future investigation of the blind speech dereverberation problem. These considerations and suggestions for future work will be described next.

The maximum and minimum entropy approach were superior to the kurtosis approach when testing their ability to deconvolve a RIR, but they also required the most computations as compared to the kurtosis approach. Thus the maximum entropy and minimum entropy algorithms require 566 times and 5007 more multiplications per algorithm update as compared to the kurtosis algorithm, respectively, with the parameters chosen in chapter 8. The minimum entropy algorithm was as the only algorithm able to significantly reduce the late reflections, but also required by far the most computations. The maximum entropy algorithm performed the best for the simple and real life RIR, but has the disadvantage that it requires some a priori knowledge of the source signal. However, it requires 9 times less computations than the minimum entropy approach as described in section 8.4. It will therefore be desirable to utilize the strength of each entropy algorithm in a combination.
The project group therefore suggest a new algorithm based on both algorithms, where the computations has been lowered and no a priori knowledge is required. The idea is to exchange the pdf estimation based on Parzen windowing in the minimum entropy algorithm with another pdf estimation based on the Kullback-Leibler (KL) divergence. Baram et al. present this latter pdf estimation in [24]. Equation (6.11) relates to the KL divergence by

$$-H(\mathbf{z}) = -H(\mathbf{x}) - E[\log|\det\mathbf{J}(\mathbf{x})|] = D(f_X(x)\,||\,\det\mathbf{J}(\mathbf{x})) \tag{10.1}$$

Minimizing the KL divergence entails that, the pdf estimation of $\mathbf{x}$ is given by

$$\hat{f}_X(x) = \det\mathbf{J}(\mathbf{x}) = h_0^M \cdot \prod_i^M \frac{\partial z(i)}{\partial y(i)} \tag{10.2}$$

where the last expression is from equation (6.13), and $\mathbf{z} = q(\mathbf{y}) = q(\mathbf{h}^T\mathbf{x})$, where $q(\cdot)$ is the chosen nonlinear function. This means that it is possible to estimate the pdf using the maximum entropy approach, hence by maximizing $H(\mathbf{z})$. As in chapter 6, $H(\mathbf{z})$ is maximized

by maximizing $\log|\det\mathbf{J}(\mathbf{x})|$. This pdf estimate is then used in the minimum entropy approach. The block diagram in figure 10.1 shows the proposed algorithm

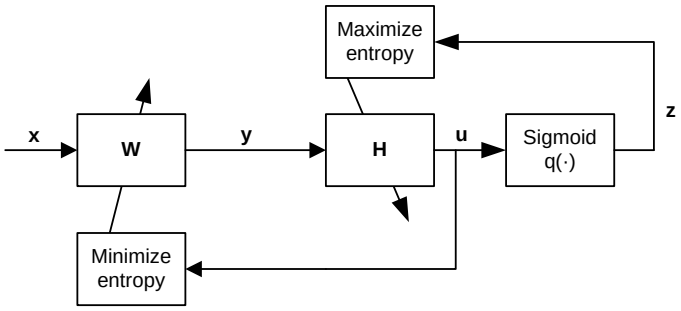


Figure 10.1: The proposed algorithm for blind speech dereverberation.

The update of $\mathbf{w}$ in the proposed steepest descent algorithm is then

$$\Delta w_1 = 2\sum_{i=1}^{M}\tanh(u(i))\cdot f(i) \tag{10.3}$$

$$\Delta w_{j+1} = 2\sum_{i=j+1}^{M}\tanh(u(i))\cdot f(i-j) \tag{10.4}$$

where $\mathbf{f}=\mathbf{H}\mathbf{x}$. The nonlinear function is based on $\tanh(\cdot)$ and is given by $\mathbf{z}=0.5(1+\tanh(\mathbf{u}))$. Note that there is no need to match it to the pdf of the source signal. The filter coefficients for $\mathbf{h}$ are calculated according to equation (6.15) and (6.16).
This proposed algorithm requires less computations as compared to the minimum entropy algorithm, if the pdf estimation based on the KL-divergence is cheaper than the method based upon Parzen windowing. Furthermore no a priori knowledge of the source is required. The algorithm will be further analysed after the deadline of this project. Interesting analysis areas are among others the validity of this new type of pdf estimate, and the robustness of the estimate with respect to parameter sensitivity. The test of the Parzen window, presented in appendix D, showed that the non-parametric pdf estimation is sensitive to the parameters controlling the window and its kernel, and therefore it is interesting to investigate if the new estimate in equation (10.2) is more robust.

Besides the novel approach the project group suggests the following subjects for future work: When the recorded speech was used as input to the recorded RIR, described in section 7.4, it was determined that the statistical assumptions regarding the input signal do not match the statistics of the speech very well, meaning the algorithms have trouble converging to the correct solution. It is therefore suggested to investigate how the statistical properties of the speech differ from the assumptions and how this knowledge can be used to solve the problem. Another problem which is evident, is that the algorithms require tuning of parameters such as the step size, and that the optimal values for the parameters depend on the input and the length of the input. An investigation of the optimal values for a general speech sequence would be very useful in the further development of the algorithms.
The validity of the Parzen pdf estimate when using the recorded speech should be examined. In the test where the recorded speech was used as input, described in section 7.4, it was assumed, that the window size and kernel standard deviation, selected in section 6.3, is also applicable to the speech sequence.
Finally, future work should focus on the complexity of the algorithms, which are far from real time execution, as calculated in section 8.4.

Bibliography

[1] ESN, *Studieordning for uddannelserne: Civilingeniør i Informatik, Civilingeniør i Kommunikationsnetværk, Civilingeniør i Proceskontrol, Civilingeniør i Signalbehandling, Civilingeniør i Telekommunikation, 1.-4. semester*. The Study board for Electronics and Information Technology, 2008. http://www.esn.aau.dk/fileadmin/esn/Studieordning/Cand_SO_ed_aalborg_sep08.pdf.

[2] M. Wu and D. Wang, "A two-stage algorithm for enhancement of reverberant speech," *Acoustics, Speech, and Signal Processing. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 1, pp. 1085–1088, 18-23, 2005.

[3] S. T. Neely and J. B. Allen, "Invertibility of a room impulse response," *The Journal of the Acoustical Society of America*, vol. 66, no. 1, pp. 165–169, 1979.

[4] B. Yegnanarayana and P. Murthy, "Enhancement of reverberant speech using lp residual signal," *Speech and Audio Processing, IEEE Transactions on*, vol. 8, pp. 267–281, May 2000.

[5] C. Nikias and J. Mendel, "Signal processing with higher-order spectra," *Signal Processing Magazine, IEEE*, vol. 10, pp. 10–37, Jul 1993.

[6] B. W. Gillespie, H. S. Malvar, and D. A. F. Florencio, "Speech dereverberation via maximum-kurtosis subband adaptive filtering," *Acoustics, Speech, and Signal Processing. Proceedings. (ICASSP '01). IEEE International Conference on*, vol. 6, pp. 3701–3704, 2001.

[7] N. D. Gaubitch, P. A. Naylor, and D. B. Ward, "Multi-microphone speech dereverberation using spatio-temporal averaging," *in Proc. European Signal Processing Conf. (EUSIPCO), Vienna, Austria*, pp. 809–812, September 2004.

[8] M. Tonelli and N. Mitianoudis, "A maximum likelihood approach to blind audio dereverberation," *Proc. of the 7th Int. Conference on Digital Audio Effects (DAFx'04)*, pp. 254–261, October 2004.

[9] A. J. Bell and S. T. J., "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, pp. 1129–1159, 1995.

[10] D. Erdogmus, K. Hild, J. Principe, M. Lazaro, and I. Santamaria, "Adaptive blind deconvolution of linear channels using renyi's entropy with parzen window estimation," *Signal Processing, IEEE Transactions on*, vol. 52, pp. 1489–1498, June 2004.

[11] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, July,October 1948.

[12] M. Miyoshi and Y. Kaneda, "Inverse filtering of room acoustics," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 36, pp. 145–152, Feb 1988.

[13] J.-H. Lee and S.-Y. Lee, "Blind dereverberation of speech signals using independence transform matrix," *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 2, pp. 1453–1457 vol.2, July 2003.

[14] D. Jurafsky and J. H. Martin, *Speech and Language Processing - An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall, 1st ed., 2000.

[15] A. V. Oppenheim and R. W. Schafer, *Discrete-time signal processing.* Prentice-Hall, Inc, second ed., 1999.

[16] S. M. Ross, *Introduction to probability and statistics for engineers and scientists.* Academic Press, third ed., 1999.

[17] K. S. Shanmugan, *Digital and analog communication systems.* Wiley & Sons Inc., 1 ed., 1979.

[18] C. L. Nikias and A. P. Petropulu, *Higher-Order Spectra Analysis.* Prentice-Hall, Inc., first ed., 1993.

[19] P. Paajarvi and J. LeBlanc, "Skewness maximization for impulsive sources in blind deconvolution," *Signal Processing Symposium. NORSIG 2004. Proceedings of the 6th Nordic*, pp. 304–307, 2004.

[20] B. Jelonnek and K. D. Kammeyer, "A closed-form solution to blind equalization," *Signal Process.*, vol. 36, no. 3, pp. 251–259, 1994.

[21] O. Shalvi and E. Weinstein, "New criteria for blind deconvolution of nonminimum phase systems (channels)," *Information Theory, IEEE Transactions on*, vol. 36, pp. 312–321, March 1990.

[22] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification.* John Wiley & Sons, Inc, second ed., 2001.

[23] L. Laughlin, "A simple coding procedure enhances a neuron's information capacity," *Z. Naturforsch*, vol. 36, pp. 910–912, June 1981.

[24] Y. Baram and Z. Roth, "Forecasting by density shaping using neural networks," *Computational Intelligence for Financial Engineering,Proceedings of the IEEE/IAFE*, pp. 57–71, Apr 1995.

[25] A. Papoulis, *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, Inc., 3rd ed., 1991.

[26] Various_artists, *Music for Archimedes.* Bang & Olufsen, 1st ed., 1992.

[27] D. Erdogmus, J. Principe, and L. Vielva, "Blind deconvolution with minimum renyi's entropy," *EUSIPCO*, vol. 2, pp. 71–74, 2002.

[28] J. F. Bercher and C. Vignat, "A renyi entropy convolution inequality with application," *in Proc. EUSIPCO, Tolouse, France*, 2002.

[29] S. Haykin, *Unsupervised Adaptive Filtering, Volume 1: Blind Source Separation.* John Wiley and Sons,Inc., 1st ed., 2000.

[30] A. Renyi, "On measures of entropy and information," *Proc. Fourth Berkeley Symp. Math. Stat. and Probability*, vol. 1, pp. 547–561, 1961.

[31] E. Parzen, "On estimation of a probability density function and mode," *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1065–1076, 1961.

[32] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[33] Texas_Instruments, *TMS320C6727B, TMS320C6726B, TMS320C6722B, TMS320C6720 Floating-Point DSPs*. TI Inc., e ed., 2008.

[34] Texas_Instruments, *TMS320C67x/C67x+ DSP CPU and Instruction Set Reference Guide*. TI Inc., a ed., 2006.

[35] M. I. Jordan, *An introduction to probabilistic graphical models*. University of California, Berkeley, not published yet, 2003.

[36] J. H. Reed, *Software Radio: A Modern Approach to Radio Engineering*. Prentice Hall PTR, 2nd ed., 2002.

[37] Videbæk_Højttalerfabrik, "Data sheet for m10md-39-08 $4\frac{1}{2}$" midrange driver," *Available on http://doc.es.aau.dk/fileadmin/doc/labs _facillities/acoustics/man/spk/Vifa _M10MD-39-08.pdf*, 1988.

[38] Brüel and Kjær, "Calibration chart for type 4010 microphone," *Available on http http://acoustics.aau.dk/localaccess/labfac/equip/pdf/bk_4010_AAU8349_ser1276881_a.pdf*, 1986.

A

Autocovariance Calculations

In this appendix the relationship between the autocovariances of the in- and output in figure 4.3 in section 4.3 are determined. Note that the processes have zero mean that is the autocovariance is equal to the autocorrelation.
The autocovariance C_2^y of the output y is defined as

$$
\begin{aligned}
C_2^y(\tau) &= E\left[y(n)\cdot y(n+\tau)\right] - \left(E\left[y(n)\right]\right)^2 && \text{(A.1)}\\
&= E\left[\sum_{i=-\infty}^{\infty} h(i)x(n-i)\cdot \sum_{j=-\infty}^{\infty} h(j)x(n+\tau-j)\right]\\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} h(i)h(j)E\left[x(n-i)x(n+\tau-j)\right]\\
&= \sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} h(i)h(j)C_2^x(n+\tau-j-(n-i))\\
&\overset{j=m+i}{=} \sum_{i=-\infty}^{\infty}\sum_{m=-\infty}^{\infty} h(i)h(i+m)C_2^x(\tau-m)\\
&= \sum_{m=-\infty}^{\infty} C_2^h(m)C_2^x(\tau-m)\\
&= C_2^h(\tau) * C_2^x(\tau) && \text{(A.2)}
\end{aligned}
$$

where:	h is a stable and causal filter	[-]
	x is a input signal, which is white and stationary	[-]
	$C_2^x(n)$ is the autocorrelation of x at time index n	[-]
	$C_2^h(n)$ is the autocorrelation of h at time index n	[-]

Fourier transforming the autocovariance yields the power spectrum $S_2^y(f)$

$$\begin{aligned} S_2^y(f) &= \mathcal{F}\left\{C_2^y(\tau)\right\} \qquad \text{(A.3)} \\ &= \mathcal{F}\left\{C_2^h(\tau) * C_2^x(\tau)\right\} \\ &= \mathcal{F}\left\{h(\tau) * h(-\tau)\right\} \cdot \mathcal{F}\left\{C_2^x(\tau)\right\} \\ &= H(f)H(-f)S_2^x(f) \\ &= |H(f)|^2 S_2^x(f) \qquad \text{(A.4)} \end{aligned}$$

where:	$\mathcal{F}\{\cdot\}$ is the Fourier transform	
	$H(f)$ is the frequency response of the filter h	[-]
	$S_2^x(f)$ is the power spectrum of x	[-]

The interpretation of this result is given in section 4.3.

B

Accept/Reject Sampling

In the maximum entropy approach to blind speech dereverberation, for which an algorithm was analysed and implemented in section 6.1, it is required to choose a sigmoid that from a priori knowledge approximates the cdf of the desired speech $\mathbf{s}$. It is therefore convenient to analyse the algorithms with a known source, $\mathbf{s}$, which has a known pdf, such that the chosen sigmoid equals the cdf of $\mathbf{s}$.

The hyperbolic tangent function, $\tanh(\cdot)$, was claimed to be a suitable cdf to the desired speech signal, [9]. In section 6.1 the project group suggested a modified $\tanh(\cdot)$, in contrast to [9], where the slope around origo could be adjusted by a constant a. The suggested cdf, y, and the corresponding pdf $p(x)$ is given by

$$y = 0.5(1 + \tanh(ax)) \tag{B.1}$$

$$p(x) = \frac{a}{2\cosh^2(ax)} \tag{B.2}$$

Sampling from the pdf, $p(x)$, must therefore be performed to generate test signals. Various algorithms exist for this purpose. The accept/reject sampling algorithm is simple and reliable and therefore chosen. The following description of the algorithm is written with inspiration from [35]. Let $p(x)$ be given by

$$p(x) = \frac{\tilde{p}(x)}{\int \tilde{p}(x)dx} \tag{B.3}$$

where: $\tilde{p}(x)$ is a distribution that it is easy to sample from.

then choose a distribution $q(x)$ that is simple to sample from, and where $kq(x) \geq \tilde{p}(x)$ for all x, $k \in R$. The algorithm works as follows:

1. Generate x_0 from $q(x)$.
2. Generate u_0 from the uniform distribution $U \sim [0, kq(x_0)]$.

3. Reject x_0 if $u_0 > \tilde{p}(x_0)$ otherwise accept x_0.

That this is a valid approach can be seen from the fact that the probability for accepting x_0 is $\frac{\tilde{p}(x)}{kq(x)}$. The distribution of x is then given by the normalization

$$\frac{[\tilde{p}(x)/(kq(x))]\, q(x)}{\int [\tilde{p}(x)/(kq(x))]\, q(x)dx} = \frac{\tilde{p}(x)}{\int \tilde{p}(x)dx} = p(x) \tag{B.4}$$

Let $q(x) \sim N(0,1)$ and $\tilde{p}(x) = \frac{a}{2\cosh^2(ax)}$. Knowing that both distributions are concave and obtain their maximum in $x = 0$, it is possible to determine k such that $kq(x) \geq \tilde{p}(x)$, which was the initial requirement. In figure B.1 the accept/reject sampling is illustrated for this choice of $q(x)$.

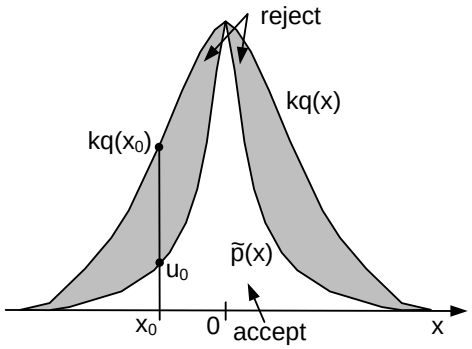


Figure B.1: The accept/reject sampling.

The constant k is then given by

$$kq(x_0 = 0) \geq \tilde{p}(x_0 = 0) \quad \Leftrightarrow \quad k\frac{1}{\sqrt{2\pi}} \geq \frac{a}{2} \quad \Leftrightarrow \quad k \geq \frac{a}{2} \cdot \sqrt{2\pi} \tag{B.5}$$

By choosing $k = \frac{a}{2} \cdot \sqrt{2\pi}$ the smallest rejection rate is obtained. In listing B.1 the implemented accept/reject sampling algorithm is given.

```
function p=acccpetreject(alpha,nosamples)

k=(a/2)*sqrt(2*pi); % to ensure that kq(x) >= p(x)
i=1; % used for indexing distribution p
const=1/sqrt(2*pi);
p=zeros(nosamples,1);

while (i<nosamples)
    % Generate x0 from q(x), q(x)=N(0,1)
    x=randn;
    % Generate u0 from the uniform distribution U=[0,kq(x0)]
    u=k*const*exp(-0.5*x^2)*rand;
    % Accept/reject
    ptilde=a/(2*cosh(a*x).^2);
    if(u<=ptilde) % then accept x(n)
        p(i)=x;
        i=i+1;
    end
end
```

Listing B.1: Accept/reject sampling.

C

Nonparametric Density Estimation

Density estimation is needed in the minimum entropy approach to blind deconvolution described in section 6.3. The basic theory of nonparametric density estimation is described next. This appendix is inspired by [22].

Let the probability P be an averaged version of the distribution $p(\mathbf{x})$ over the region R and let P be the probability that a vector $\mathbf{x}$ lies within this region

$$P = \int_R p(x')dx' \tag{C.1}$$

Then let $x_1, \ldots, x_n$ be n iid samples of $p(x)$. According to the binomial distribution, the probability that k of these samples will fall in R is given by

$$P_k = \left(\begin{array}{c} n \\ k \end{array} \right) P^k (1-P)^{n-k} \tag{C.2}$$

where the expected value is

$$E[k] = nP \tag{C.3}$$

Furthermore let V be the volume enclosed by R and let R be small such that the change in $p(\mathbf{x})$ is small, then

$$P = \int_R p(x')dx' \approx p(x)V \tag{C.4}$$

Combining equation (C.3) and (C.4) the following is obtained

$$p(x) \approx \frac{k}{nV} \tag{C.5}$$

An averaged estimate of the distribution $p(x)$ has been obtained. In order to converge to the desired $p(x)$ then $V \to 0$. However the latter will entail that k eventually equals zero which makes $p(x) = 0$. Therefore to ensure convergence three conditions must be satisfied

- $\lim_{n\to\infty} V = 0$
- $\lim_{n\to\infty} k = \infty$
- $\lim_{n\to\infty} k/n = 0$

A way to meet these requirements is to use the Parzen window method, where the region depends on the chosen window type. In section 6.3 the Gaussian kernel is used. The Gaussian kernel is the Gaussian or normal probability density function defined as

$$\kappa_\sigma(y_k - y_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(y_k - y_i)^2}{2\sigma^2}\right) \tag{C.6}$$

where: σ is the standard deviation of the Gaussian pdf

It is not required that the kernel of the Parzen window is a pdf. This means that the scaling factor $\frac{1}{\sqrt{2\pi}\sigma}$ can be omitted from equation (C.6).

As mentioned above the kernel is used in the Parzen window, which was defined in equation (6.35) and repeated here for convenience

$$\hat{f}_Y(y_k) = \frac{1}{L} \sum_{i=k}^{L+k-1} \kappa_\sigma(y_k - y_i) \tag{C.7}$$

where: $\hat{f}_Y(y_k)$ is the estimated pdf for the sample y_k
L is the window size

The estimated pdf $\hat{f}_Y$ for the sample y_k is actually not a pdf on its own, but the estimated probability for the sample y_k. The Parzen window together with the Gaussian kernel utilizes the value of the sample y_k as the mean of the Gaussian pdf. The window size then determines how many samples that are to be weighted using the Gaussian pdf, with the given mean and standard deviation. In short this means that for every input value y_k in the Parzen window a Gaussian pdf is generated, with mean equal to the input, and the samples that fall within this pdf add to the total probability. This is illustrated in figure C.1.

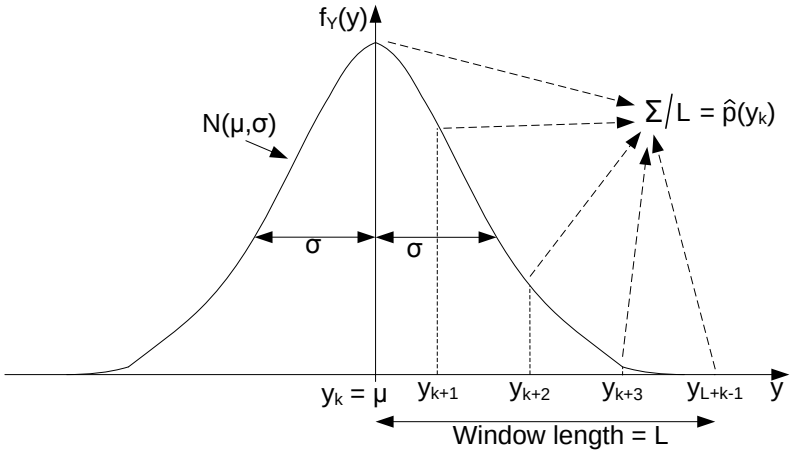


Figure C.1: The estimated probability for the sample y_k using a Gaussian kernel and a Parzen window of length $L = 5$.

By combining the estimated probabilities of all the input samples the estimated pdf can be generated. This is illustrated in figure C.2 for a Gaussian kernel.

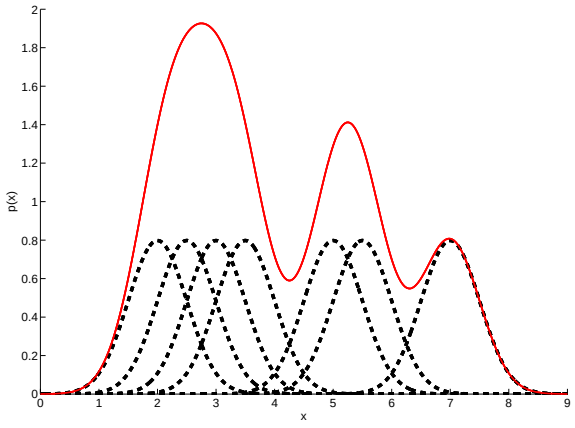


Figure C.2: A pdf estimated using the Parzen window and Gaussian Kernels. The pdf (solid curve) is the sum of seven Gaussian pdfs (dashed black curves) using means equal to the input samples.

This completes the short description of nonparametric density estimation and the Gaussian kernel.

D

Pdf Estimation Test

This appendix contains the results of the pdf estimation test. The approach was to estimate a pdf using the Parzen window with a given window size N and standard deviation σ in the Gaussian kernel. When the pdf estimation was finished the result was compared with the original pdf, that is the pdf the input samples were drawn from, using the Kullback-Leibler divergence (KL divergence). The divergence metric and its useful property is described in section 6.3.
The test was performed with three different Parzen windows, a window using samples backwards in time as compared to the current sample, a window centered around the current sample, and a window using samples forwards in time as compared to the current sample. The results for combinations of N and σ are shown in table D.1, D.2, and D.3 respectively.

$\sigma \downarrow \quad N \rightarrow$	20	30	50	100	200	500	1000	2000
0.0100	0.128	0.116	0.092	0.060	0.035	0.015	0.008	0.004
0.1000	0.033	0.025	0.018	0.013	0.010	0.008	0.008	0.008
0.1500	0.036	0.031	0.027	0.024	0.022	0.021	0.021	0.021
1.0000	0.118	0.118	0.118	0.118	0.118	0.118	0.118	0.118
2.0000	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
10.0000	0.127	0.127	0.127	0.127	0.128	0.128	0.128	0.128

Table D.1: KL divergence results using a backward window.

$\sigma \downarrow \quad N \rightarrow$	20	30	50	100	200	500	1000	2000
0.0100	0.128	0.115	0.093	0.060	0.035	0.015	0.008	0.004
0.1000	0.033	0.025	0.018	0.013	0.010	0.008	0.008	0.008
0.1500	0.036	0.031	0.027	0.024	0.022	0.021	0.021	0.021
1.0000	0.118	0.118	0.118	0.118	0.118	0.118	0.118	0.118
2.0000	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
10.0000	0.127	0.127	0.127	0.127	0.128	0.128	0.128	0.128

Table D.2: KL divergence results using a centered window.

$\sigma \downarrow \quad N \rightarrow$	20	30	50	100	200	500	1000	2000
0.0100	0.128	0.115	0.092	0.060	0.035	0.016	0.008	0.004
0.1000	0.033	0.025	0.018	0.013	0.010	0.008	0.008	0.008
0.1500	0.036	0.031	0.027	0.024	0.022	0.021	0.021	0.021
1.0000	0.118	0.118	0.118	0.118	0.118	0.118	0.118	0.118
2.0000	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
10.0000	0.127	0.127	0.127	0.127	0.128	0.128	0.128	0.128

Table D.3: KL divergence results using a forward window.

The first observation, regarding the results in the three tables, is that the placement of the window has a negligible effect on the estimated pdf. This means that the choice of window placement can be made solely from an implementation point of view without considering possible errors in the pdf estimate. The preferred window type will therefore be a backward window, because it only requires samples, that are older than the current one, in the calculations.
The second observation is that a large window size ($N = 1000$ and $N = 2000$) combined with a small standard deviation ($\sigma = 0.01$ and $\sigma = 0.1$) result in the smallest divergence. That is a large window and a small standard deviation lead to the best pdf estimate. The fact that a large window size leads to the best results is not surprising, because for iid samples a window with infinite length would be the optimal.
The standard deviation $\sigma = 0.1$ will in general give the best estimate no matter what window size is used. A large window size means many extra calculations as compared to one that is e.g. ten times smaller and therefore it is desirable to choose a smaller window, from an implementation point of view.

E

Matlab Code for the Algorithms

In the following listings the Matlab code for the implemented algorithms to the speech dereverberation problem are given. To see all implemented Matlab code, for tests etc., please refer to the enclosed CD.

The algorithm for kurtosis maximization:

```
function h = KurtMax(input,hstart,L,step)
% Inputs:
%   input = the input signal
%   hstart = the initial filter coefficient
%   L = length of the FIR filter
%   step = step size in the update equation
% Output:
%   h = the final filter coefficients
inputLength = length(input);
h = hstart;
beta = 0.99;
sim = 1:(inputLength-L+1); % The simulation steps
y = h'*flipud(input(1:L));
exp2_old = (1-beta)*y^2;
exp4_old = (1-beta)*y^4;

for n = sim
    k = n+L-1;
    y = h'*input(k:-1:n); % processed signal
    exp2 = beta*exp2_old+(1-beta)*y^2;
    exp4 = beta*exp4_old+(1-beta)*y^4;
    feedback = 4*(exp2*y^2-exp4)*y/exp2^3; % feedback function
    exp2_old = exp2;exp4_old=exp4;
    h = h + step*feedback*input(k:-1:n); % update h
    h(1) = 1; % normalization
end
```

Listing E.1: Kurtosis maximization algorithm.

The algorithm for maximum entropy:

```
function h = InfoMax(input, M, L, step, a, hstart)
% Inputs:
%   input = the input signal
%   M = length of input in each iteration
%   L = length of the FIR filter
%   step = step size in the update equation
%   a = the slope coefficient in tanh(a*y)
%   hstart = the initial filter coefficient
% Output:
%   h = the final filter coefficients

h = hstart;
inputLength = length(input);
sim = 1:(inputLength-M+1); % The simulation steps
f = zeros(L,1); % Preallocation of the differentiated cost function
count = 1;

for n=1:sim
    k=n+M-1;
    Hmatrix = convmtx(h,M);
    Hmatrix = Hmatrix(1:M,:); % update Hmatrix
    y = Hmatrix*input(n:k); % processed signal
    zc = tanh(a*y); % Sigmoid
    % cost function
    f(1) = M/h(1)-2*a*zc'*input(n:k);
    for j = 1:(L-1)
        f(j+1) = -2*a*zc(j+1:M)'*input(n:k-j);
    end
    h = h+step*f(:,1); % update h
    h(1) = 1; % normalization
end
```

Listing E.2: Maximum entropy algorithm.

The algorithm for minimum entropy:

```
function h = MinEnt(input,std_dev,N,step,L,h)
% Inputs:
%   input = the input signal
%       std_dev = The standard deviation when using a Gaussian Kernel
%       N = Number of samples in the Parzen window.
%       step = Step size in gradient descend algorithm
%       L = Filter length
% Output:
%   h = the final filter coefficients

var1 = 1/std_dev^2;
var2 = -1/(2*std_dev^2);
len = length(input);
T = len-L+1;
sim = 1:T-N+1; % The simulation steps
%%% make reverb matrix %%%%
ReverbMatrix = zeros(L,T);
for k=1:T
    ReverbMatrix(:,k)=input(k:k+L-1);
end
ReverbMatrix=flipud(ReverbMatrix);

for startP=sim
    denomSum = 0; % initialization of the denominator summing variable
    nomSum = zeros(L,1); % the nominator summing variable
    for jj = startP:startP+N-1
        ReverbDiff = repmat(ReverbMatrix(:,jj),1,N) - ReverbMatrix(:,startP
            :startP+N-1);
        ydiff = ReverbDiff'*h;
        exp1 = exp(var2*ydiff.^2);
        denomSum = denomSum+sum(exp1);
        nom =  ReverbDiff*(exp1.*ydiff);
        nomSum = nomSum+nom;
    end
    Jdiff = var1*nomSum/denomSum; % the differentiated cost function
    h = h - step*Jdiff; % update h
    h(1) = 1; % normalization
end
```

Listing E.3: Minimum entropy algorithm.

F

Measurement of a RIR

In this appendix the measurement of a real life RIR is described. The measurement was carried out in the listening room of the Section of Acoustics at the Department of Electronic Systems at Aalborg University. The project group was assisted by electronics engineer Claus Vestergaard Skipper who is the technician in the Section of Acoustics.

The measurement report is organized as follows. First the purpose of the measurement is explained. Then the measurement procedure is described including theoretical considerations. Next the list of used equipment is presented and then the measurement data are described. Then the results are given and a conclusion is made based on the results. Finally the uncertainties regarding the measurement are discussed.

Purpose

In chapter 5 and 6 several algorithms were presented, implemented and tested. The tests were performed with simple FIR and IIR filters used as RIRs and the input to these room filters were randomly generated data with a known pdf, as described in section 6.1. The tests are useful when comparing the algorithms and when the goal is to comprehend the functionality of the algorithms. However the tests do not indicate how the algorithms will perform when applied to a hearing aid, which is the application of this project, since the tests do not reflect a ”real life” situation. The project group therefore decided to measure the RIR of a room, which models a normal living room. Because of the application being a hearing aid it is also interesting to measure how the RIR changes when the microphone (the hearing aid) is moved just a few centimetres as would be the case if a person wearing a hearing aid moved his or her head.
The performed measurements are divided into two parts. First the RIR is measured when the microphone is moved around in the room and then different kinds of previously recorded speech and music is played in the room and the reverberated sounds are obtained. The latter measurements are made since they will provide the project group with interesting and realistic

test signals. The exact procedure for these two measurements are described in the next section.

Measurement Procedure

As mentioned in the previous section the measurements where divided into two separate parts. The measurement of the RIR and the recording of the reverberated speech and music are however carried out in very similar fashions. In both measurements the key idea is to set up a loudspeaker and a microphone inside the room and then play a sequence on the loudspeaker and record the received and reverberated signal with the microphone. It is important that the utilized equipment has a flat frequency response so that the transfer function of the equipment does not affect the RIR.
The used listening room is built together with a control room so that cables in the listening room can be connected directly to equipment in the control room. Using this control room and sealing up the listening room means that there is no interference and change of the setup in the listening room during all the recordings.
A sketch of the used listening room is shown in figure F.1

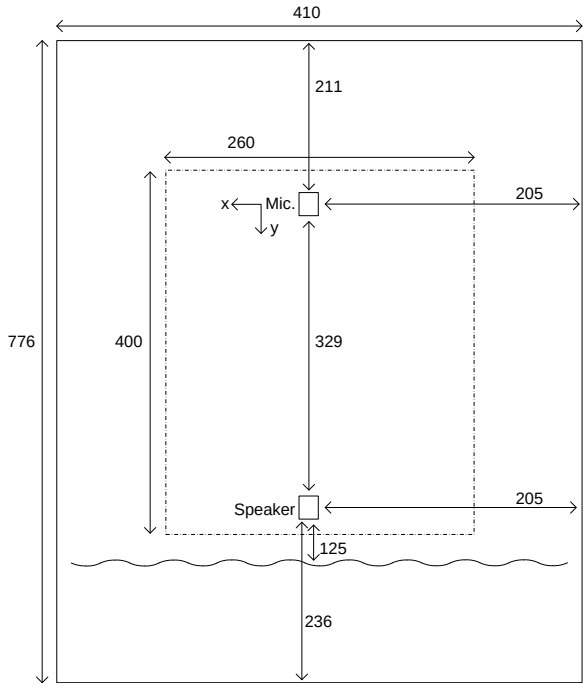


Figure F.1: The listening room. All measures are in centimeters and the speaker and the microphone are directed towards each other. The wavy line in the lower part of the sketch indicates a curtain in the room, which spanned from the floor to the ceiling. Note that the distance between the microphone and the speaker is measured between the two tripods holding the devices. The direct distance between the microphone head and the speaker is therefore 11 centimeters shorter.

When performing the RIR measurements the loudspeaker was connected to a power amplifier which again was connected to a data acquisition device (DAS). The DAS was then connected to a PC where a program matching the DAS was executed. The program generated the test signals and recorded the signal from the microphone which also was connected to the DAS. The setup is illustrated in figure F.2.

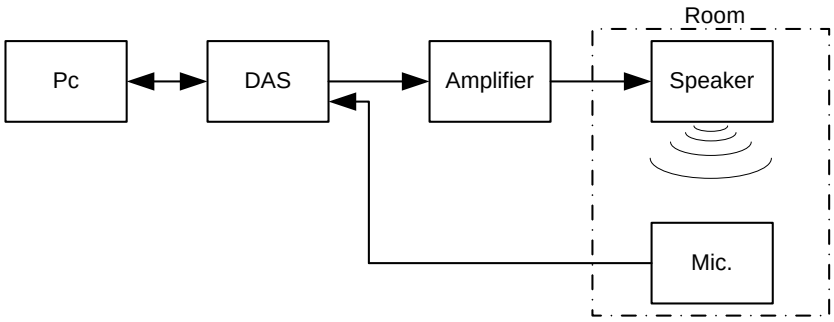


Figure F.2: The used setup when measuring the RIR. The equipment framed by the dotted line was located in the listening room whereas the equipment outside the dotted box was located in the control room.

To determine the RIR a test signal known as a maximal length sequence (MLS) was used. A MLS sequence is a special type of pseudo random binary sequence because it has a maximum period hence its name. The MLS sequence is generated via a length m shift register and among its many properties is the fact that if the period of the sequence is $N = 2^m - 1$ then the autocorrelation of the sequence $m_2(\tau)$ is one for zero lag and zero for all other lags that is [36]

$$m_2(\tau) = \delta(\tau) \tag{F.1}$$

where: τ is the lag
$\delta(\tau)$ is the Kronecker delta

This is a very useful property when measuring a RIR because the RIR ($g(n)$ in figure 4.1) can be obtained directly by determining the cross-correlation between the MLS input $s(n)$ and the recorded output $x(n)$

$$\begin{aligned} x(n) &= g(n) * s(n) \\ m_2^{xs}(n) &= g(n) * m_2^s(n) \\ m_2^{xs}(n) &= g(n) \end{aligned} \tag{F.2}$$

The program dBFA32, which belongs to the used DAS, was run on the PC and it is capable of performing the cross-correlation calculations so that the RIR was obtained directly.
In total 40 recordings where made, that is the RIR was measured at 40 different positions in the room. At each position the MLS sequence was played 16 times and the average RIR was determined. As illustrated in figure F.1 a coordinate system with a x and a y axis was made in the room. The tripod holding the microphone was then positioned in the coordinate system, which was marked on the floor. Before each measurement the position of the microphone was compared to origo of the coordinate system. Figure F.3 shows the different positions of the microphone and furthermore each position (hence each measurement) has been given a number indicated by the #.

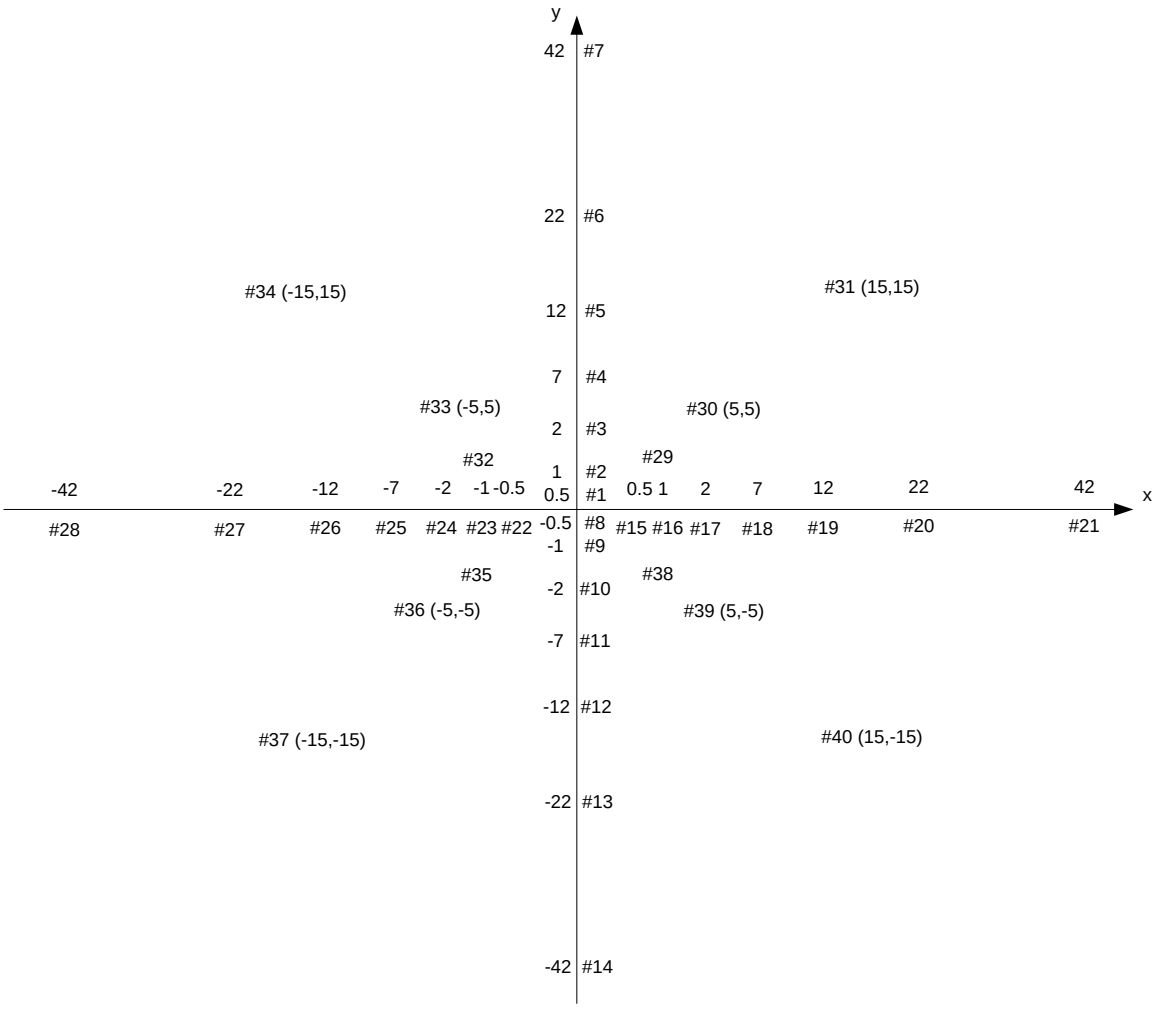


Figure F.3: The coordinate system used when measuring the RIR. The values with a # in front indicates the measurement number. The values without the # is the distance from the point to origo in centimetres. The positive directions of the x and y axes are given in figure F.1.

In the second measurement the idea was to play speech and music signals in the listening room and therefore the DAS was not used to generate the MLS sequence any more. In stead a CD containing recordings of speech and music in an anechoic chamber was played on the PC and through the PC sound card passed directly to the power amplifier. The setup is illustrated in figure F.4. The reason why the special recordings from the anechoic chamber were used is that it is important to use signals that are not already affected by a RIR.

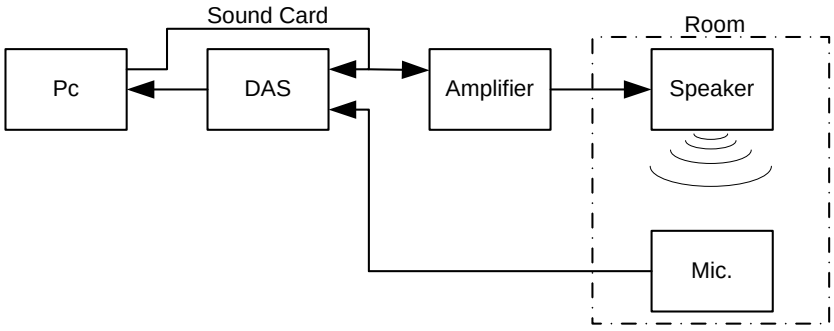


Figure F.4: The used setup when recording the reverberated speech and music signals.

As illustrated in figure F.4 the output from the PC sound card is not only connected to the power amplifier but also to the DAS. This connection is made so that it is easy to synchronize the reverberated signal with the original one. When the reverberated signal was recorded on the PC the original signal was simply recorded as well through a second channel in the DAS. During all the measurements the microphone was placed at the origo of the coordinate system illustrated in figure F.1 and F.3.

The measurement procedure for both measurements is given below

1. If necessary position microphone according to the loudspeaker
2. Playback MLS sequence or recorded speech/music
3. Record and save signal received at the microphone
4. Repeat item 1 to 3

List of Used Equipment

Table F.1 contains a list of the used equipment.

Name	Model	AAU ID	Notes
Power amplifier	Pioneer stereo amplifier A-616	08340	Set to 10 dB gain
Data acquisition system	01dB Symphonie	33694	
Microphone	Brüel & Kjær type 4010	8349	No. 1276881-A, omnidirectional
Microphone power supply	ART-cessories Phantom II	2157-56	Set to 48 V, using a battery
Loudspeaker	Nokalon Trawlnet Ball. Designed by the Section of Acoustics using a Vifa M10MD3908 driver	02017-22	Reg.no. 34758.
PC			Running dBFA32

Table F.1: Used equipment. The AAU ID is the internal identification number for all equipment available at the Department of Electronic Systems.

The dBFA32 program was calibrated in the following way when the RIR was measured

- Type = MLS acquisition
- order = 15
- Number of averages = 16
- Frequency range = 20 kHz
- Mode = single channel
- Acquisition duration = 10 s and 240 ms
- Response duration = 640 ms

When the reverberated speech and music was recorded the default settings were used in the dBFA32 program.
The speech and music samples were obtained from the Bang & Olufsen Archimedes CD [26]. The used samples are 10 second clips of male and female speech, a cello theme and a guitar play.

Data

Because of the large amount of data gathered in the measurement the project group has chosen to place all data on the attached CD in the directory `RecordedRIRs`. Some interesting results are presented in the Results section.

All data was saved in *.wav* format using the dBFA32 program and sampled with 51.2 kHz. In total 41 measurements of the RIR was made using the names given in figure F.3 that is measurement #1 is called *RIR1.wav*, #2 *RIR2.wav* and so forth.
The data obtained in the second measurement is also saved in *.wav* format. The files are named *speechX.wav*, *guitarX.wav*, and *celloX.wav*. If the $X\, modulus 2$ in the filename is even the file contains the recording from the channel receiving the reverberated version and if $X\, modulus 2$ is odd the file contains the original recording from the anechoic chamber. E.g. *speech2.wav* is a recording of some type of reverberated speech and *speech3.wav* is the corresponding original recording of the same speech.

Results

In this section a few of the RIR measurement results are presented. Figure F.5 shows six RIRs measured at different positions in the listening room. Based on the figure it can be concluded that moving the microphone slightly closer to the speaker (#1 in the figure) does not affect the RIR, but moving it towards one of the sides of the room causes a change of the RIR (#17 and #29). The figure also confirms that moving the microphone far away from origio affects the RIR. Again it is seen that movement on the x axis (#28) has a greater effect than moving it on the y axis (#7).
The conclusion is that moving the microphone, that is the hearing aid, changes the RIR, but the changes are in general small because the main reflections remain the same.

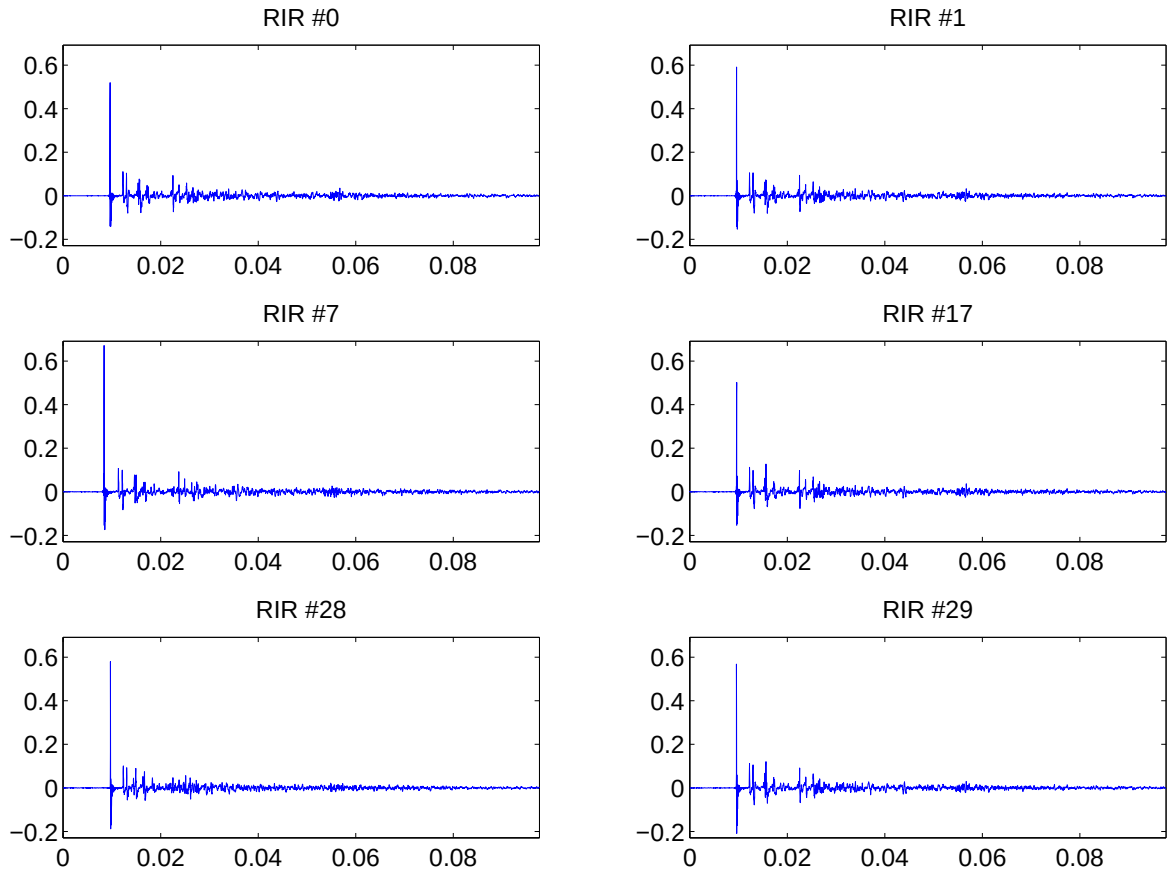


Figure F.5: Six RIRs. The title of each subplot is the measurement number given in figure F.3. The unit of the x axis is seconds.

Uncertainties

The measurements contain several uncertainties.
The most serious problem is the frequency response of the equipment, which will affect the measurement of the RIR and the recorded signals. The setups illustrated in figure F.2 and F.4 can be viewed as a cascade of transfer functions, where each piece of equipment represents a transfer function. The only transfer function of interest is the room itself and therefore it is important that the utilized equipment has a flat frequency response in the frequency band of interest. To minimize this undesired effect on the measurements the project group selected a microphone and a speaker, which according to the data sheets [37] and [38], have a flat frequency response. It is therefore assumed that the effect is negligible.
The error imposed by the frequency response of the equipment will change during time because the temperature in the equipment changes. The biggest problem is in the speaker, and in the documentation of the speaker it is mentioned that if the speaker is used for more than 30 seconds with a high-level input the driver will cause internal heating in the speaker and as a result the frequency response will change. During the measurements the maximum length of the input signal was 10.24 seconds and furthermore there was a break in between measurements of at least 3 minutes because the microphone had to be repositioned. This uncertainty is therefore ignored.
Another problem is the precision in the positioning of the microphone during the RIR measurements. The project group took great care in measuring the position and directing the

microphone towards the speaker, but minor errors could not be avoided in the simple setup. Overall the data show the expected RIR and therefore it is assumed that misadjustments of 1-2 mm have not seriously affected the measurement.
A minor problem is the omnidirectional microphone. If the microphone was perfect it would have the same gain in all directions, but this is not possible. This means that the incoming reflections from the room will be weighted differently in the microphone. The difference in gain is assumed to be small and therefore this uncertainty is disregarded.
The final uncertainty is the fact that the MLS sequence is only an approximation. This means that the autocorrelation, which is assumed to be a Kronecker delta is actually only close to being a delta function. The approximation is however very good for long periods and therefore the project group used a 640 ms test signal which was played 16 times after which the result was averaged. It is therefore assumed that the RIR measurement is valid.