



NTNU – Trondheim
Norwegian University of
Science and Technology

Real-Time Simulation of Reduced Frequency Selectivity and Loudness Recruitment Using Level Dependent Gammachirp Filters

Gaute Bertheussen

Master of Science in Communication Technology

Submission date: June 2012

Supervisor: Odd Kr. Pettersen, IET

Co-supervisor: Tron Vedul Tronstad, SINTEF

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

PROBLEM DESCRIPTION

A real-time hearing loss simulator is to be implemented using the C programming language. Reduced frequency selectivity and loudness recruitment should be the focus of the simulator. The simulator will make use of the level dependent gammachirp filter bank to divide the signal into subbands. Each subband is then to be processed individually to create effects of the two impairments.

PREFACE

This thesis marks the end of my M.Sc. degree in Communication Technology - Signal Processing and Communication. Its topic, simulation of hearing loss, was suggested by Professor II Odd. Kr. Ø. Pettersen as a specialization project in Audio and Image Processing. The project was carried out in the fall semester of 2011. This thesis is a continuation of that project.

I would like to thank my supervisor Tron Vedul Tronstad for his insight through our Skype meetings. I would also like to thank my girlfriend Elise, my mother Kirsti and my father Svein for their support.

ABSTRACT

A real-time system for simulating reduced frequency selectivity and loudness recruitment was implemented in the C programming language. The finished system is an executable program where a user can input a sound file and a list of hearing losses. As the program runs, a processed version of the input signal is played back. The processed signal includes the effects of either one or both the hearing impairments. The system, called a hearing loss simulator, is based on the dynamic compressive gammachirp filter bank. Each channel in the filter bank is signal dependent, meaning the filter characteristics are changed according to an estimate of the signal level. Reduced frequency selectivity was simulated by influencing the filter characteristics by a hearing loss value in addition to the signal level. This produced masking effects, and was able reduced the detail of spectral envelopes. Loudness recruitment was simulated by scaling each sample based on the signal level. This technique accounted for abnormal growth of loudness-level and elevated absolute thresholds. It made low sounds disappear while leaving loud sounds closer to their original level.

SAMMENDRAG

Et sanntidssystem for simulering av redusert frekvensselektivitet og "loudness recruitment" ble implementert i programmeringsspråket C. Det ferdige systemet er et kjørbart program hvor brukeren kan gi inn en lydfil og en liste med hørselstap. Mens programmet kjører spilles det av en prosessert utgave av inputsignalet. Det prosesserte signalet inneholder effektene av en eller begge hørselstap-effektene. Systemet kalles en hørselstapssimulator og er basert på den dynamisk kompressive gammachirp filterbanken. Hver kanal i filterbanken er signalavhengig. Dette medfører at egenskapene til filterene endres med et estimat av signalnivået. Redusert frekvensselektivitet simuleres ved at filteregenskapene også påvirkes av en hørselstapsverdi. Dette produserte maskeringseffekter, og reduserte detaljnivået i spektralenvelopen til et signal. "Loudness recruitment" ble simulert ved å skalere hver sample basert på signalnivået. Denne teknikken redegjorde både for unormal nivåvekst og økte høreterskler. Lave lyder forsvant, mens høye lyder ble satt til verdier nærmere originalnivået.

CONTENTS

1	Introduction	1
2	Background and Motivation	3
2.1	The Ear	3
2.2	Hearing Loss	4
2.2.1	Reduced Frequency Selectivity	6
2.2.2	Loudness Recruitment	7
2.3	Auditory Filters	9
2.3.1	Approximating the Auditory Filters	10
3	Implementation	13
3.1	Audio Input and Output	13
3.2	Cyclic Buffers	14
3.3	Digital Gammatone Filters	16
3.4	Digital Asymmetric Compensation Filters	18
3.5	Dynamic Compressive Gammachirp Filter Bank	20
3.5.1	The Passive Gammachirp Analysis Filter	21
3.5.2	The Dynamic HP-AC Filter	22
3.5.3	The Passive Gammachirp Synthesis Filter	25
3.6	Simulating Loudness Recruitment	28
3.7	Simulating Reduced Frequency Selectivity	29
3.8	Reducing Computational Cost	30
3.8.1	Resampling	31
3.8.2	Lookup Tables	32
3.8.3	Avoiding Trigonometric Functions	32
3.9	The Finished System	34
3.9.1	The C Program	36

4	Results	39
4.1	Results of RFS Simulation	40
4.2	Results of LR Simulation	44
4.3	Results of RFS and LR Simulation	47
5	Discussion and Future Work	51
6	Conclusions	55
	List of Abbreviations	i
	Appendices	
A	Degrees of Hearing Loss	iii
B	Using the System	v

1

INTRODUCTION

Hearing loss is a problem for a great number of people. Depending on the severity of the losses, they can drastically reduce peoples ability to communicate. Hearing loss is in many cases noise induced. Noise exposure increases the likelihood of experiencing damaged hearing, and is particularly a problem at noisy workplaces. However, noise induced hearing loss can in many cases be avoided by using ear protection.

Hearing loss simulators have been developed for quite some time. Both analog and digital implementations have tried to mimic the effects hearing impaired people experience. Most of the simulators have been made to understand how hearing impaired people percept speech. This thesis documents the development of a hearing loss simulator.

The hearing loss simulator that is developed runs in real-time. Opposed to many other simulators, its focus is not solely on speech intelligibility research, but also to function as a tool demonstrating the perceptual effects of hearing loss. Hopefully, when people are presented with the consequences of hearing loss, they become aware of the importance of ear protection.

The purpose of having a real-time hearing loss simulator is to allow the user to immediately compare normal and damaged hearing. Presenting the hearing losses in this manner makes it a well suited approach for a demonstrational tool. The user is given control of the simulations and the perceptual changes are presented automatically and immediately.

This simulator is based on an earlier simulator implemented as a specialization project in Audio and Image Processing at NTNU [1]. The earlier simulator was implemented in Matlab. It ran offline, meaning the sound could not be heard until after the simulator had finished processing the entire signal. Both this and the earlier simulator's focus was the impairments reduced frequency selectivity and loudness recruitment.

Structurally this thesis will present the reader with the theoretical background first, and how it motivates the choices made for the simulator. The background includes how our hearing works and the hearing impairments that are simulated. Afterwards the reader will be presented a detailed technical description of the signal processing that makes the hearing loss simulator work. Subsequently the results of the simulations, discussion of the results, suggestions for improvements and a conclusion is presented.

2

BACKGROUND AND MOTIVATION

This chapter presents the background theory needed to understand the implementation of the hearing loss simulator. How the background theory motivated the choices made for the simulator is also described. The functionality of the ear and the hearing loss effects that are simulated, loudness recruitment and reduced frequency selectivity, are presented first. Lastly mathematical functions describing the auditory filters are presented, motivating the digital filters used in the implementation.

2.1 THE EAR

When a person hears a sound, air pressure is lead into the ear by the funnel that is the outer ear. The outer ear consist of the pinna, the visible part of the ear, and the ear canal. At the end of the ear canal lies the tympanic membrane, or eardrum. The eardrum is caused to vibrate by the air pressure [5, p. 7], and converts it into mechanical vibrations through three small bones in the middle ear [22]. These bones are connected to the inner ear through an opening called the oval window [5, p. 7]. The oval window sits on the cochlea, a spiral shaped cavity filled with fluid. When the bones interact with the oval window, the mechanical

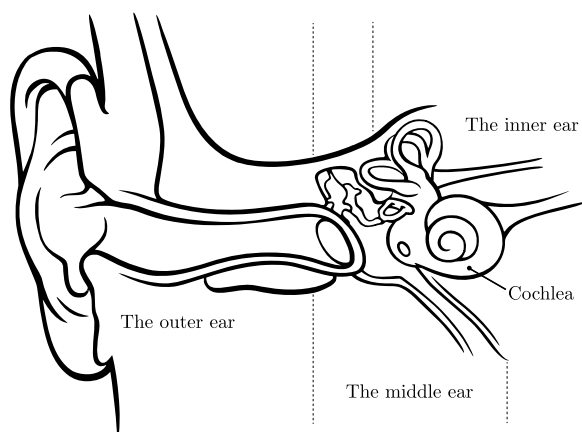


Figure 2.1: Simplified anatomic view of the auditory system. Adapted from a figure courtesy of Lars Chittka and Axel Brockmann [9].

vibrations are converted into waves in fluid inside the cochlea. The cochlea is divided by two membranes, one of which is the basilar membrane [5, p. 9]. Between the basilar membrane and a membrane called the tectorial membrane are hair cells [5, p. 10]. These hair cells are separated into two groups with different tasks: Inner hair cells (IHC) and outer hair cells (OHC). At the tip of a hair cell is what resembles hair, the stereocilia. The stereocilia of the OHCs appear to make contact with the tectorial membrane, but those of the IHCs do not [5, p. 11]. Most of the neurons that carry information from the cochlea to the brain are connected to the IHCs. Effectively this means that when the basilar membrane moves, the IHCs convert the movement into neural activity which is perceived by the brain. The OHCs have a different function. They influence the response of the basilar membrane [5, p. 11].

As the basilar membrane is excited, the position of its displacement reveals the frequency content of the input [22]. An example of this is that a high frequency causes the highest basilar membrane response near the base of the cochlea. At this position the basilar membrane is relatively stiff and narrow. Farthest away from the oval window, at the apex, the basilar membrane is less stiff, causing it to respond best to low frequencies. This means that the cochlea acts as a spectral analyzer [5, p. 10-13].

There are two mechanisms that determine the response of the basilar membrane, one is the passive mechanism. The linear passive mechanism largely depends on the mechanical properties of the basilar membrane [5, p. 13]. Another is the nonlinear active mechanism, which depends on the OHCs. The active mechanism controls the broadness of the basilar membrane displacement, making it narrower for low sound levels. Nonlinear compression is also a feature of the active mechanism. This compression makes a great number of air pressure amplitudes fall into a much smaller range of basilar membrane responses [5, p. 15-16]. For low to medium sound levels the difference between the air pressure and the basilar membrane response is nonlinear, they are amplified. However, at high levels, over 90 dB SPL, the difference is linear, and the active mechanism does not contribute [5, p.17]. This nonlinear compression is what gives the auditory system its large dynamic range, and allows us to hear quiet as well as loud sounds.

2.2 HEARING LOSS

There are two types of hearing loss: Conductive and sensorineural. Conductive losses are caused by difficulties transmitting sound from the outer ear to the cochlea. These difficulties may be a blockage of the ear canal, a damaged eardrum, or reduced func-

tionality of the bones and muscles in the inner ear. Sensorineural losses are caused by damage to the cochlea and the nerves connected to the cochlea [5, p. 28]. Cochlear damage is the major reason for sensorineural hearing loss, causing cochlear hearing loss [10, p. 61]; The focus of this thesis.

The cochlea depends on the functionality of the IHCs and OHCs. Reduced functionality of the hair cells may be caused by damage to the stereocilia, or the death of an entire cell [5, p. 29]. When IHCs are damaged, it reduces the transmission efficiency from the cochlea to the brain [5, p.43]. This means that the basilar membrane vibration must be larger than normal to produce a sound level in an impaired ear that equals the sound level in a normal ear. When this occurs, it is called elevated absolute thresholds, which can also be caused by OHC damage. An absolute threshold is the lowest sound level required for a tone, in absence of other sounds, to be audible [1; 5]. This means that for a particular frequency, if a person has an elevated absolute threshold of 40 dB, the sound level must be increased by 40 dB SPL for that person to hear it. When listening to speech this can make consonants, which are fairly weak, inaudible [5, p. 44]. The Norwegian Labour Inspection Authority has labeled hearing loss into three degrees. The audiograms, showing absolute thresholds for these losses, can be seen in Figure 2.2. As hearing degrades with age, a third degree loss is higher for a 65 year old person than for a 20 year old person. Tables describing hearing loss with age can be seen in Appendix A.

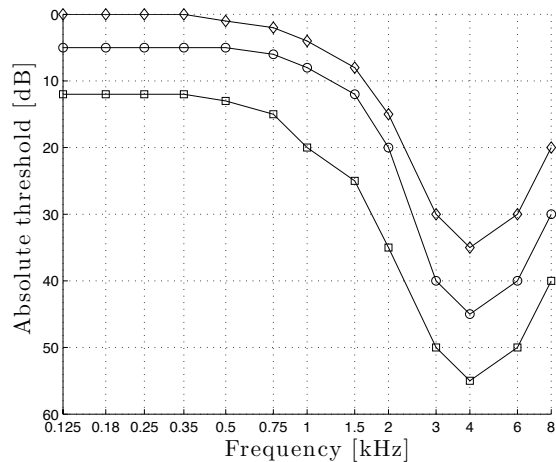


Figure 2.2: Three degrees of hearing loss. First degree, diamonds. Second degree, circles. Third degree, squares.

OHC damage can only contribute to 50 dB loss at low frequencies and 60 dB loss at high frequencies [5, p. 43]. Moore and Glasberg [11] created a model where hearing loss was the sum of the loss caused by OHC and IHC damage. They used that the maximum loss caused by OHC damage was 57.6 dB, a value also used in this thesis.

As the OHCs control the active mechanism, damaged cells mean damaged mechanism [5, p. 29]. Loudness recruitment and reduced frequency selectivity, the hearing loss

effects that are simulated, are mostly related to a damaged active mechanism.

2.2.1 REDUCED FREQUENCY SELECTIVITY

When the OHCs are damaged, and the active mechanism cease to function, a consequence may be reduced frequency selectivity (RFS). Frequency selectivity is a listeners ability to separate, or discriminate, between frequencies in a sound [5, p. 45]. Hence, RFS causes difficulties resolving frequencies. What happens is that the damaged active mechanism will no longer narrow the basilar membrane response [5, p. 29 and 45].

As it will be described later, the cochlea can be assumed to work as a filter bank of partially overlapping band-pass filters. Widening of the basilar membrane response leads to a widening of these filters [5, p. 79], and could result in RFS. How much the filters are widened, and how they are widened varies from person to person [5; 30]. However, Moore and Glasberg [30] concluded that for hearing impaired people, the low-frequency side of an auditory filter was often less steep than for normally hearing people. This indicated that the widening of the auditory filters generally occur at the low-frequency side [5, p. 81], making them pass more low-frequency components. Widening of auditory filters is the motivation for how RFS is simulated, and will be described in Section 3.7.

Baer and Moore [24] and Moore, Glasberg and Simpson [27] claimed that RFS causes several differences in an impaired ear compared to a normal ear. RFS would produce broader excitation patterns. An excitation pattern is a representation of basilar membrane activity [5, p. 50]. Broader excitation patterns would make the individual frequency components of a signal harder to resolve. The spectral envelope of complex signals, signals with several frequency components, would be blurred. Blurring the envelope could cause vowels, which are largely determined by the peaks of their spectral envelope [18, p. 39], harder to discriminate between. The output of an auditory filter would cause complex sounds to reflect the interaction between a number of components. Lastly: Noisy input would cause the widened filters to pass more noise, causing a lower signal to noise ratio [1; 24; 27].

RFS could cause several perceptual problems. One problem is susceptibility of masking. Masking is the phenomenon that occurs when one sound renders another inaudible. If a signal at one frequency is rendered inaudible by a signal at another frequency, a masker, then the two signals can not be discriminated between [5, p. 46]. An example of masking is that a low-frequency signal, such as air conditioning noise, could mask a high-frequency signal, such as an alarm [1]. As mentioned above, blurring of the spectral envelope could cause confusion between vowels. This means that another per-

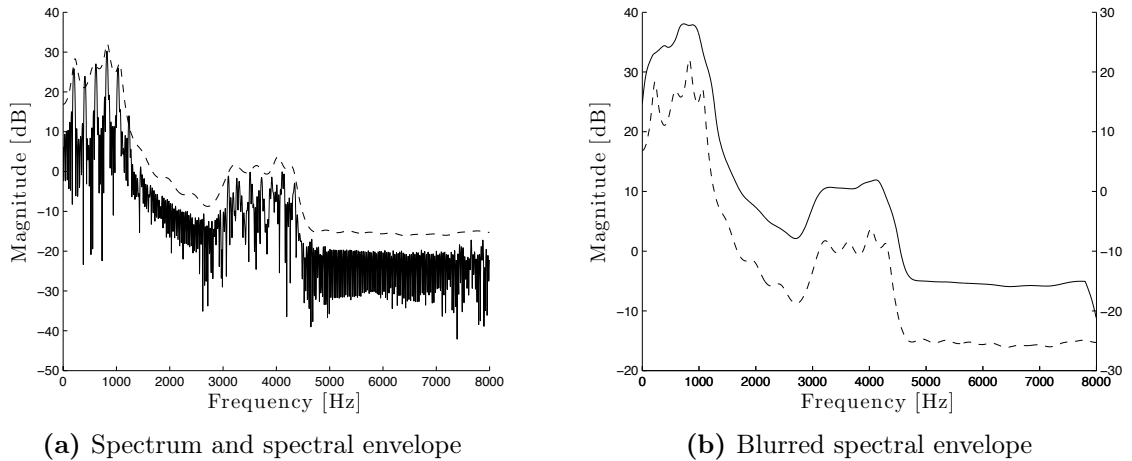


Figure 2.3: Example of spectral envelope blurring. Figure 2.3a shows the spectrum for the vowel /a/ in solid lines and its spectral envelope in dashed lines. Figure 2.3b shows the spectral envelope and a blurred spectral envelope, solid line, right axis. The spectral envelope was approximated by modeling the vowel as an autoregressive process. The blurred envelope was computed by convolving the spectral envelope with a rectangular averaging window of length 100.

ceptual consequence of RFS is difficulties understanding speech, especially in presence of interfering speech or noise [1; 5, p. 90]. The noise could act as a masker, making spectral components of the speech signal inaudible.

2.2.2 LOUDNESS RECRUITMENT

The active mechanism serves as an automatic gain control, amplifying quiet sounds [25]. This amplification is the cause of the nonlinear compression in the cochlea. People with normal hearing has a nonlinear basilar membrane input-output function. However, after 90 dB SPL the active mechanism cause no gain. No gain means that the input-output function is linear [5, p. 17]. When the active mechanism is damaged, the input-output function is nearly linear for all levels, and could cause loudness recruitment (LR).

LR is described as an unusually rapid growth of loudness-level [10; 19]. It usually occurs in combination with elevated absolute thresholds [25]. The combined effects of LR and threshold elevation can be described when comparing a normal and an impaired ear. Say that a person has a flat hearing loss of 30 dB in one ear, and no hearing loss in the other. For this person a level of 30 dB is inaudible in the impaired ear. However, a high level of 100 dB can be perceived as equally loud in both ears. If this person only had elevated thresholds, the high level would be perceived as being 70 dB. This means

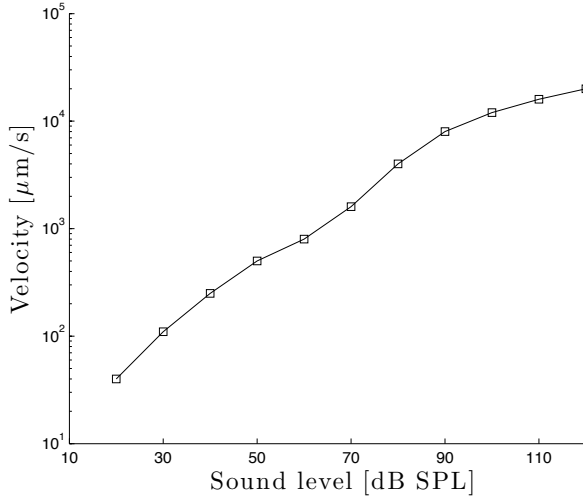


Figure 2.4: Hypothetical basilar membrane input-output function for a normal ear. The figure shows nonlinear growth of basilar membrane response until the level reaches 90 dB SPL, from that point on the response is linear. Based on a figure by Moore [5].

that quiet sounds are made quieter, while high sounds can be perceived as normally loud, or even louder. If the sound is perceived as louder it is called over-recruitment [5, p. 98].

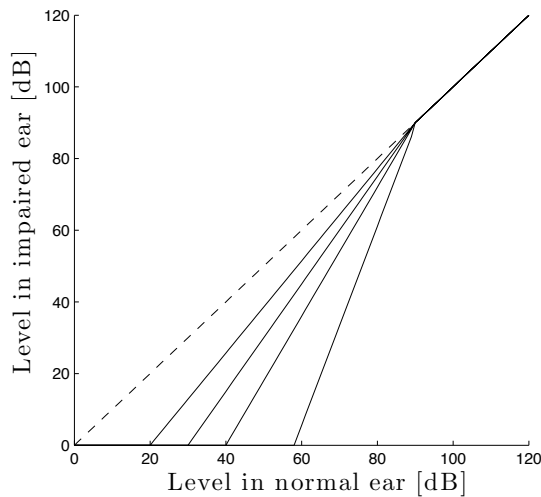


Figure 2.5: Comparison between levels in a normal ear and an impaired ear. The solid lines are according to hearing losses of 20, 30, 40 and 57.6 dB, where the steepest line is computed from the highest loss. The dashed line shows the level when both ears are normal. Figure from *Simulation of Reduced Frequency Selectivity and Loudness Recruitment* [1].

By comparing the levels of an impaired and normal ear, the growth function between these ears can be determined. Moore and Glasberg [25] reported that the resulting function was usually a straight line from the absolute threshold to the level where recruitment is complete. This level is usually above 90 dB SPL, where the active mechanism no longer affects the basilar membrane. Figure 2.5 shows the growth function for a number of hearing losses. This function is used to simulate LR, as will be described in Section 3.6.

LR can affect how speech and music is perceived. Speech can be unintelligible, not only because of elevated thresholds, but also since the difference between vowels and consonants are larger than normal [5, p. 114]. In the event of interfering speech: If

a hearing impaired person is talking to another person in a public space, the hearing impaired person may have problems hearing what the other person is saying. However, if a person in the background shouts something, the shouted utterance may be perceived as normally loud. In the case of music, LR may cause larger than normal differences between quiet and loud parts of a song, changing its intended dynamic qualities [1; 5, p. 114].

2.3 AUDITORY FILTERS

As the cochlea acts as a spectral analyzer, it is assumed that it works as a filter bank consisting of partially overlapping band-pass filters. These filters are called auditory filters, and are thought to correspond to a physical location on the basilar membrane [5, p. 46].

Band-pass filters are often defined by their bandwidth. There are several ways to define the bandwidth. For auditory filters, a common definition is the equivalent rectangular bandwidth (ERB). The ERB of a filter equals the bandwidth of a rectangular filter that transmits the same power of white noise as the desired filter, and has the same passband transmission [1; 5, p. 55]. It is assumed that an ERB corresponds to a constant distance along the basilar membrane, a distance of about 0.89 mm [5, p. 56; 29]. The definition of the ERB was given by Glasberg and Moore [29]:

$$\text{ERB}(f) = 24.7 (0.00437 f + 1). \quad (2.1)$$

Another value closely related to the ERB is the ERB-scale (ERBS) . An ERBS value describes the number of ERBs below a given frequency [3; 5, p. 56]. The ERBS is found by integrating the reciprocal of (2.1) [29], and is, for this task, useful for determining the location of the filters in the filter bank.

$$\text{ERBS}(f) = 21.4 \log_{10}(0.00437 f + 1) \quad (2.2)$$

Patterson [31] described a way of measuring the the auditory filter shapes called the notched-noise method. The resulting shape has been approximated by several mathematical functions. A common function is the one describing the gammatone filter, which has later been extended to become the gammachirp filter.

2.3.1 APPROXIMATING THE AUDITORY FILTERS

The gammatone filter was found to explain the masking data provided by Patterson’s method, and became popular as a way of simulating cochlear filtering [13]. The time-domain impulse response of the gammatone filter is seen in (2.3). The equation depends on a normalization constant β , filter order N , center frequency f_0 , the filter bandwidth $b(f_0)$ and the phase ϕ . The phase is often omitted as it does not match the phase response of the ear [15]. The bandwidth is a scaled version of the ERB. The scaling value b is given in Table 3.2, but was originally recommended to be 1.019 for a fourth order filter [20].

$$g(t) = \beta t^{N-1} \exp(-2\pi b(f_0)t) \cos(2\pi f_0 t + \phi) \quad (2.3)$$

$$b(f_0) = b \text{ERB}(f_0) \quad (2.4)$$

The gammatone is different from the auditory filters in several ways. Figure 2.6b shows the magnitude response of the gammatone filter. The magnitude is approximately symmetrical, a feature it shares with the auditory filters only at moderate input levels [5, p. 57; 13].

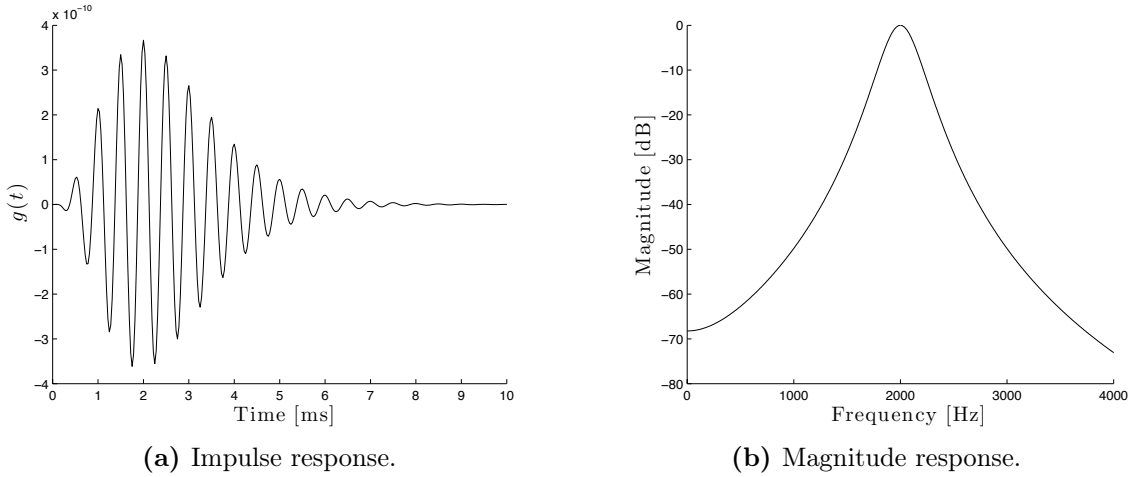


Figure 2.6: Impulse and magnitude response of a fourth order analog gammatone filter with an input frequency of 2 kHz and a bandwidth of 245.27 Hz.

For high signal levels the auditory filter is asymmetric, making the low-frequency side of the filter shallower. It is also nonlinear. The gammatone lacks both these features, motivating the development of the gammachirp filter.

Several implementations of the gammachirp has been developed. The magnitude responses in Figure 2.7a are of the compressive gammachirp filter (CGC) , computed

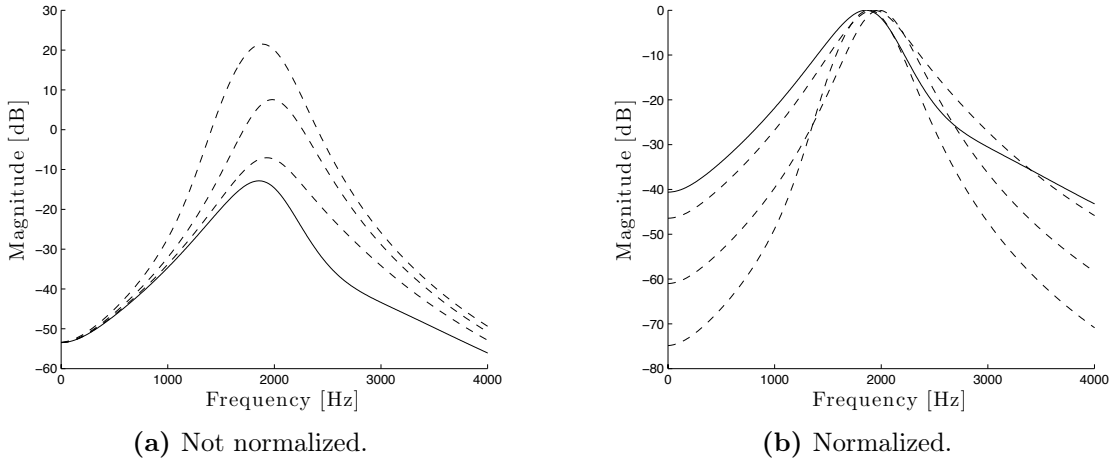


Figure 2.7: Magnitude responses of compressive gammachirp filters created with signal levels of 80, solid line, 60, 40 and 20 dB.

using a number of signal levels. The figure shows that the lowest signal level produces a filter with the most gain and the narrowest bandwidth. Conversely, the highest signal level produces a filter with the least gain and the widest bandwidth. Patterson, Irino and Unoki [7; 8; 13; 16] developed the CGC that is described here. It is designed to incorporate the cochlear nonlinear compression in the filtering process [8]. As the cochlea of normally hearing people already exhibit this compression, the gain caused by the compression is normalized in the implementation described in Chapter 3. Not normalizing the filters will cause low signal levels to sound louder, the opposite of what is expected when simulating hearing loss. The normalized filters have the characteristics shown in Figure 2.7b.

$$|H_{CGC}(f)| = |H_{GT}(f)| \cdot |H_{AF}(f, c_1, b_1)| \cdot |H_{AF}(f, c_2, b_2)| \quad (2.5)$$

$$= |H_{PGC}(f)| \cdot |H_{AF}(f, c_2, b_2)| \quad (2.6)$$

Equation (2.5) describes the magnitude response of the frequency domain CGC filter and shows that it is a product of three terms. The first is the magnitude response of a gammatone filter, the second the magnitude of a low-pass asymmetric function (LP-AF), and the third a level dependent high-pass asymmetric function (HP-AF). The combination of the gammatone and the LP-AF is called the passive gammachirp filter (PGC), and represents the passive mechanism. The product of the PGC and the HP-AF represents the active mechanism [8].

$$|H_{AF}(f, c, b)| = \exp(c \theta(f)) \quad (2.7)$$

$$\theta(f) = \arctan\left(\frac{f - f_0}{b \text{ERB}(f_0)}\right) \quad (2.8)$$

The asymmetric function, described by (2.7), is a low-pass filter if c is negative, a high-pass filter if c is positive and an all-pass filter if $c = 0$ [21]. This means that the second term of (2.5) is created when c_1 is negative, and the last with c_2 being positive. A digital approximation to the asymmetric function was developed by Irino and Unoki [21] and is described in Section 3.4.

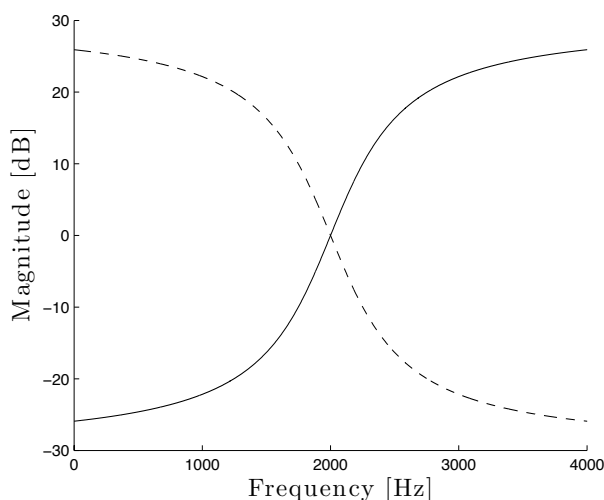


Figure 2.8: Magnitude response of an LP-AF, dashed line, and HP-AF, solid line. The functions were computed with an input frequency of 2 kHz. The absolute value of c is the same for both functions, but the signs differ.

Irino and Patterson [8] developed a dynamic version of the CGC called the dynamic compressive gammachirp filter (DCGC). The DCGC includes level estimation on a sample by sample basis. The estimated level is used to change the characteristics of the CGC. Irino and Patterson stated that the DCGC produced two-tone suppression naturally. Two-tone suppression is a nonlinear feature of the basilar membrane [5, p. 18]. It is the phenomenon that occurs when one tone reduces the basilar membrane response to another tone. When Irino and Patterson processed speech through their DCGC filter bank, they stated that the frequency components changed from being suppressors to being suppressed [8]. As it will be described later, a DCGC filter bank is implemented according to Irino and Patterson's design, with the addition of normalizing the filters. The reason for the choice of filter was twofold. One was that the level dependence of the DCGC would be realistic relative to the cochlea. Another was that by changing the filter parameters according to a hearing loss, it would possibly exaggerate the two-tone suppression. This could lead to the effects of RFS, and perhaps LR, being a result of the filtering process itself.

3

IMPLEMENTATION

This chapter presents the implementation of the hearing loss simulator. It describes how audio is read and played back and the digital approximations of the gammatone and asymmetric compensation filter. How the dynamic compressive gammachirp filter bank is composed is also presented, along with how the hearing losses are simulated. The chapter ends with how the computational complexity of the system was reduced and a short description of the finished C program.

3.1 AUDIO INPUT AND OUTPUT

A key aspect of the real-time hearing loss simulator is to provide the user the ability to listen to the simulations immediately. For a user this means that the system should respond to the user's input, and provide output accordingly within a certain amount of time. The input of this system is a number of hearing losses for different frequency bands, and the output is processed sound.

The audio output is handled by the Apple specific Audio Queue Services programming interface. Audio Queue Services provides the ability to record and playback audio on Mac OS X computers and iOS devices [2]. An advantage of Audio Queue Services is that the programmer does not have to implement functions to read sound files or connect to hardware, this is embedded in the interface.

Playback using Audio Queue Services works by setting up a queue of K buffers. Each buffer is initially filled with L samples. This implies that Audio Queue Services segments the signal, in this case a sound file, into frames of length L . When all K buffers are filled for the first time, the playback starts. After the contents of a buffer has been

played back, the Audio Queue Services invokes a callback function on that buffer. The callback function, which is created by the programmer, refills the buffer with the next L samples. At this point the buffer can be processed to simulate hearing loss before it is placed at the end of the queue. The buffers are repeatedly refilled and played back until the end of sound file is encountered. When the end is encountered the playback is finished. A detailed description of how the Audio Queue Services work, and an example on how to implement the callback function can be read in the Apple documentation for the Audio Queue Services [2]. The number of buffers used in this implementation is 3. How the frame length L is chosen is described in Section 3.5.3.

3.2 CYCLIC BUFFERS

The playback method described above segments the signal into frames of length L . Filtering a single sample depends on a number of preceding samples. This leads to a problem: At the start of each segment there are no previous samples. The lack of previous samples will lead to transition artifacts between frames. These artifacts can be avoided by storing the last samples from the previous frame in a cyclic buffer.

A cyclic buffer is a first-in-first-out (FIFO) queue of constant length M . In case of filtering with infinite impulse response (IIR) filters, M must equal the number of feedback coefficients. A cyclic buffer can be viewed as a circle, it has no end and no start. In a traditional queue a new sample is added to the end, and a sample is read from the start. This means that each sample must be moved when a new is added. With cyclic buffers, samples can be read or written from anywhere within the queue, without the need to move samples. To know which sample to read, and where a sample should be written, the read and write positions in the buffer are kept at all times.

The cyclic buffer (CB) is implemented as a structure containing: An array of M samples (CB_A), a pointer to the read position in the array (CB_R) and a pointer to the write position (CB_W).

```

1: procedure READ(CB)
2:    $x \leftarrow \text{CB}_A(\text{CB}_R)$ 
3:    $\text{CB}_R \leftarrow \text{CB}_R - 1$ 
4:   if  $\text{CB}_R < 0$  then
5:      $\text{CB}_R \leftarrow M - 1$ 
6:   end if
7:   return  $x$ 
8: end procedure

```

The procedure READ describes how a sample is read from the cyclic buffer. Its input is a reference to a cyclic buffer structure. A reference to a structure means that the structure is stored somewhere else, and CB is merely a link to that location. However, the updates made to CB in READ also applies to the structure itself. READ works by first reading the sample at the read position. Subsequently the read pointer is decreased by one such that it points to the next read position. If the pointer is below zero, it is set to point to the end of the array at position $M - 1$. The procedure returns the sample that is read.

```

1: procedure WRITE(CB, x)
2:    $\text{CB}_W \leftarrow \text{CB}_W + 1$ 
3:   if  $\text{CB}_W > M - 1$  then
4:      $\text{CB}_W \leftarrow 0$ 
5:   end if
6:    $\text{CB}_A(\text{CB}_W) \leftarrow x$ 
7:    $\text{CB}_R \leftarrow \text{CB}_W$ 
8: end procedure

```

WRITE describes how a sample is written to the cyclic buffer. As with READ, a reference to a cyclic buffer structure is passed as an input. But WRITE also takes a sample as input. The procedure works by first increasing the write pointer. If the write pointer exceeds $M - 1$, it is set to point to the beginning of the array at position 0. After the write pointer is updated, the sample is stored at the updated position. Lastly the read pointer is set to equal the write pointer. Equating the two pointers ensures that the next sample that is read is the newest.

3.3 DIGITAL GAMMATONE FILTERS

The gammatone filter is implemented according to the procedure described by Holdsworth et al. [20], which was also implemented in *Simulation of Reduced Frequency Selectivity and Loudness Recruitment* [1]. The algorithm produces an IIR filter with center frequency f_0 that approximates the continuous gammatone filter from Section 2.3.1. Figure 3.1 outlines the digital gammatone filter approximation.

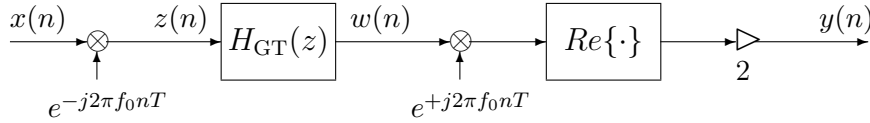


Figure 3.1: Digital approximation of the gammatone filter.

Digitally approximating the gammatone filter starts by frequency shifting the input signal $x(n)$ by $-f_0$. The resulting complex signal $z(n)$ is then passed through a filter $H_{GT}(z)$. $H_{GT}(z)$ is composed of four cascaded first order recursive filters, each with output $w_1(n)$. After filtering, the signal is frequency shifted by $+f_0$. Lastly the real part is taken. Taking the real part of the complex signal attenuates it by a factor of $1/2$. This means that to obtain 0 dB gain, the filter must be normalized by multiplying each sample by 2.

$$w_1(n) = w_1(n-1) + (1 - e^{-2\pi b(f_0)T})(z(n-1) - w_1(n-1)) \quad (3.1)$$

Equation (3.1) is the difference equation for the first order filters $H_{GT}(z)$ is composed of. The bandwidth of the filter is defined by $b(f_c)$, which is the ERB scaled by a constant b , see (2.4). The constant b will be set according to Table 3.2. $T = 1/f_s$ is the sampling rate. The transfer function for the fourth order filter, $H_{GT}(z)$, can be derived from (3.1).

$$H_{GT}(z) = \frac{(1 - e^{-2\pi b(f_0)T})^4 z^{-4}}{1 + \sum_{m=1}^4 c_m z^{-m} e^{-2m\pi b(f_0)T}} \quad \text{where } \mathbf{c} = \{-4, 6, -4, 1\} \quad (3.2)$$

The transfer function can be used to compute the difference equation (3.3) for the fourth order recursive filter. The coefficients c_m of the fourth order filter are the same as in (3.2).

$$w(n) = (1 - e^{-2\pi b(f_0)T})^4 x(n-4) - \sum_{m=1}^4 c_m e^{-2m\pi b(f_0)T} w(n-m) \quad (3.3)$$

$H_{GT}(z)$ is implemented as a direct form II structure, see Figure 3.2. Direct form II implementations are advantageous over direct form I structures in terms of memory consumption. Where a direct form I structure requires $M + N + 1$ memory locations, an equivalent direct form II structure would require $\max\{M, N\}$ [6, p. 583-584]. M is the number of feedforward coefficients, and N is the number of feedback coefficients.

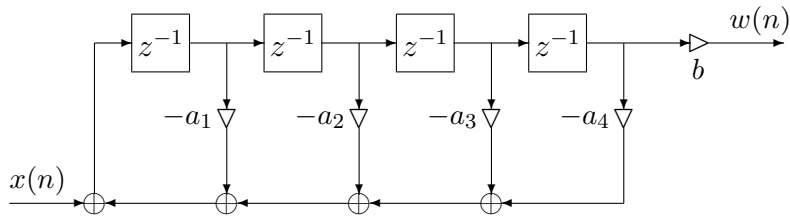


Figure 3.2: Direct form II structure of $H_{GT}(z)$. The feedback coefficients are represented by a_k , and the feedforward coefficient is represented by b .

GAMMATONEFILTERING describes how a single sample is filtered through an N th order gammatone filter, in this implementation N is four. The input is a reference to a gammatone structure and a sample. The gammatone structure contains an array of feedback coefficients (GT_A), a feedforward coefficient (GT_B), and two cyclic buffers that contain real and imaginary samples (GT_{BR} and GT_{BI}). It also contains a counter, k , and an input frequency f_0 . The counter keeps track of the number of filtered samples. This is necessary when frequency shifting the signal. Frequency shifting means that the filter $H_{GT}(z)$ needs to run on complex samples. Complex processing is implemented by multiplying the sample by the real (e_R) and imaginary (e_I) part of Euler's formula separately. The two products are then processed identically before they are combined to form the output sample.

```

1: procedure GAMMATONEFILTERING(GT, s)
2:    $e_R \leftarrow \cos(2\pi f_0 kT)$ 
3:    $e_I \leftarrow -\sin(2\pi f_0 kT)$ 
4:    $s_R \leftarrow s \cdot e_R$ 
5:    $s_I \leftarrow s \cdot e_I$ 
6:   for  $n = 0 \rightarrow N - 1$  do
7:      $t_R \leftarrow \text{READ}(\text{GT}_{BR})$ 
8:      $t_I \leftarrow \text{READ}(\text{GT}_{BI})$ 
9:      $s_R \leftarrow s_R + \text{GT}_A(n) \cdot t_R$ 
10:     $s_I \leftarrow s_I + \text{GT}_A(n) \cdot t_I$ 
11:   end for
12:    $\text{WRITE}(\text{GT}_{BR}, s_R)$ 
13:    $\text{WRITE}(\text{GT}_{BI}, s_I)$ 
14:    $k \leftarrow k + 1$ 
15:   return  $2 \text{GT}_B \cdot (t_R \cdot e_R + t_I \cdot e_I)$ 
16: end procedure

```

3.4 DIGITAL ASYMMETRIC COMPENSATION FILTERS

The asymmetric function from Section 2.3.1, composing the LP-AF and HP-AF, is what gives the CGC filter its elongated low-frequency tail and compressive abilities [8]. This section describes the digital approximation of the asymmetric function; The asymmetric compensation filter (AC). The procedure for approximating the filter was developed by Irino and Unoki [21], and consists of cascading four IIR filters.

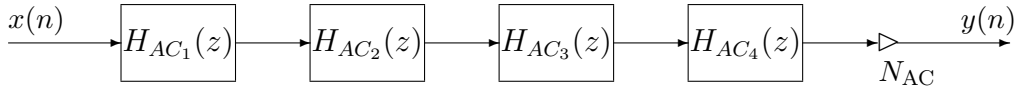


Figure 3.3: Digital approximation of the asymmetric compensation filter, $H_{AC}(z)$.

The equations (3.4) and (3.5) show that the AC filter, $H_{AC}(z)$, is a cascade of N second order IIR filters, $H_{AC_m}(z)$. The transfer function, $H_{AC_m}(z)$, can be used to derive the difference equation for each second order IIR filter, $y_m(n)$.

$$H_{AC}(z) = \prod_{m=1}^N H_{AC_m}(z) \quad (3.4)$$

$$H_{AC_m}(z) = \frac{(1 - r_m e^{j\phi_m} z^{-1})(1 - r_m e^{-j\phi_m} z^{-1})}{(1 - r_m e^{j\theta_m} z^{-1})(1 - r_m e^{-j\theta_m} z^{-1})} \quad (3.5)$$

$$y_m(n) = x(n) - 2r_m \cos(\phi_m)x(n-1) + r_m^2 x(n-2) \\ + 2r_m \cos(\theta_m)y_m(n-1) - r_m^2 y_m(n-2) \quad (3.6)$$

Equation (3.6) shows that the filter coefficients depend on ϕ_m , θ_m and r_m , variables that are different for each filter in the cascade. The variables were originally defined by Irino and Unoki [21], but were improved and redefined by Irino et al. [17].

$$r_m = \exp\{-p_1(p_0/p_3)^{m-1} \cdot 2\pi b(f_0)T\} \quad (3.7)$$

$$\phi_m = \begin{cases} 2\pi(f_0 - \Delta f_m)T & \text{if } f_0 - \Delta f_0 \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.8)$$

$$\theta_m = \begin{cases} 2\pi(f_0 + \Delta f_m)T & \text{if } f_0 + \Delta f_0 \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.9)$$

$$\Delta f_m = (p_0 p_3)^{m-1} \cdot p_2 c b(f_0) \quad (3.10)$$

The variables p_0 , p_1 , p_2 and p_3 are provided by Irino et al. [17] and are shown in Table 3.1. They are valid for a cascade of four second order IIR filter, $N = 4$, the number of filters used in this implementation.

Table 3.1: Asymmetric compensation filter variables p_0 , p_1 , p_2 and p_3 , for $N = 4$

p_0	p_1	p_2	p_3
	1.7818	0.5689	
2	$\cdot(1 - 0.0791 b)$ $\cdot(1 - 0.1655 c)$	$\cdot(1 - 0.1620 b)$ $\cdot(1 - 0.0857 c)$	1.0724

The digital AC filter has the same shape as the asymmetric function, but it is non-unitary at the input frequency f_0 . Being unitary at f_0 is desirable to avoid unintended gain, and can be obtained by normalizing the filter. By evaluating $H_{AC_m}(z)$ when $z = e^{j2\pi f_0 T}$, the normalization factor N_{AC} can be found. N_{AC} is the product of the

reciprocal magnitudes each second order filter has at f_0 .

$$N_{AC} = \prod_{m=1}^N \frac{1}{|H_{AC_m}(e^{j2\pi f_0 T})|} \quad (3.11)$$

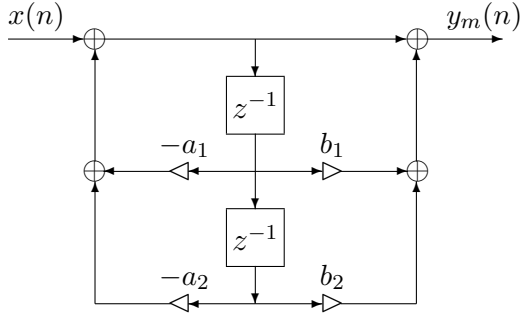


Figure 3.4: Direct form II structure of $H_{AC_m}(z)$. Where a_k represent the feedback coefficients, and b_k represent the feedforward coefficients.

ACFILTERING describes how a single sample is filtered through a second order IIR filter, $H_{AC_m}(z)$. The filter is implemented as the direct form II structure in Figure 3.4. ACFILTERING takes a reference to a second order filter structure (AC_2) and a sample as input. The structure contains a cyclic buffer (AC_{CB}), feedback coefficients (AC_A) and feedforward coefficients (AC_B). To filter the input sample through an AC filter, ACFILTERING must be run four times with different filter structures as input.

- 1: **procedure** ACFILTERING(AC_2, x)
- 2: $t_1 \leftarrow \text{READ}(AC_{CB})$
- 3: $t_2 \leftarrow \text{READ}(AC_{CB})$
- 4: $y \leftarrow x - AC_A(1) \cdot t_1 - AC_A(2) \cdot t_2$
- 5: WRITE(AC_{CB}, y)
- 6: **return** $y + AC_B(1) \cdot t_1 + AC_B(2) \cdot t_2$
- 7: **end procedure**

3.5 DYNAMIC COMPRESSIVE GAMMACHIRP FILTER BANK

The compressive gammachirp filter (CGC) is a cascade of a passive gammachirp filter (PGC) and a high-pass asymmetric compensation filter (HP-AC). If the HP-AC filter is determined by dynamic level estimation, the resulting filter is called the dynamic compressive gammachirp filter (DCGC).

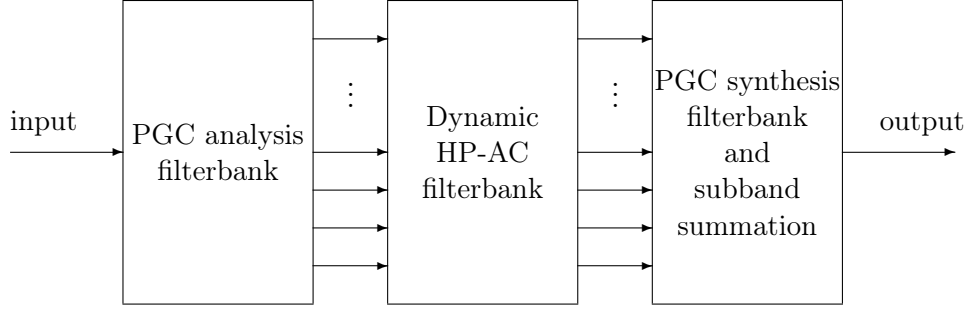


Figure 3.5: Architecture of the dynamic compressive gammachirp filterbank.

The implementation of the DCGC filter bank is described in this section. Its design follows that of Irino and Patterson [8]. The implemented DCGC filter bank is comprised of three components: An analysis PGC filter bank, a filter bank of HP-AC filters with dynamic level estimation, and a synthesis PGC filter bank.

3.5.1 THE PASSIVE GAMMACHIRP ANALYSIS FILTER

As mentioned in Section 2.3.1, the PGC filter consists of a gammatone filter cascaded with a low-pass asymmetric compensation filter (LP-AC). The procedure for filtering a frame, F , through a PGC is described in `PGCANALYSISFILTERING`.

`PGCANALYSISFILTERING` passes each sample of the frame through a gammatone filter, then through four second order IIR filters. The four IIR filters together form the LP-AC filter. `PGCANALYSISFILTERING` takes a reference to a PGC structure as input. The PGC structure contains a gammatone structure and four AC structures. The input frequency of the gammatone and the LP-AC filter is the same.

```

1: procedure PGCANALYSISFILTERING(PGC, F)
2:   for  $n = 0 \rightarrow L - 1$  do
3:      $F(n) \leftarrow \text{GAMMATONEFILTERING}(\text{PGC}_{\text{GT}}, F(n))$ 
4:      $F(n) \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC1}}, F(n))$ 
5:      $F(n) \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC2}}, F(n))$ 
6:      $F(n) \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC3}}, F(n))$ 
7:      $F(n) \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC4}}, F(n))$ 
8:      $F(n) \leftarrow F(n) \cdot N_{\text{AC}}/G_{\text{PGC}}$ 
9:   end for
10:  return F
11: end procedure

```

The LP-AC filter causes the peak frequency of the PGC to shift downwards compared to that of the gammatone. The two peak frequencies are however related through (3.12), where f_0 is the peak frequency of the gammatone, and f_p is the peak frequency of the PGC. f_p serves two purposes: One is as an input parameter to the HP-AC filter. The other is to compute the analytical peak gain of the PGC, which is used to normalize the filter. Irino and Patterson [8; 16] supplied the peak frequency, f_p , and the analytical peak gain, G_{PGC} .

$$f_p = f_0 + c \cdot b(f_0)/N \quad (3.12)$$

$$G_{\text{PGC}} = \frac{\exp\{c \cdot \arctan(c/N)\}}{[1 + (c/N)^2]^{N/2}} \quad (3.13)$$

3.5.2 THE DYNAMIC HP-AC FILTER

The dynamic HP-AC filter is the component of the DCGC that gives the filter its level dependent properties. This level dependence is specifically attributed its input frequency $\hat{f}_0(n)$, which is related to the PGC's peak frequency [8; 16]. The design of the HP-AC filter is identical to that of the LP-AC filter, although the two have differently valued coefficients c and b , see Table 3.2.

$$\hat{f}_0(n) = f_{\text{RAT}}(n) \cdot f_p = [0.466 + 0.0109 \cdot P(n)] \cdot f_p \quad (3.14)$$

The level dependence of the DCGC is caused by the term $P(n)$ in $f_{\text{RAT}}(n)$. $P(n)$ is an estimate of the signal level at sample n . By influencing the input frequency by the signal level, the overlap of the HP-AC and the PGC can be controlled. Shifting the cutoff frequency causes changes in the overlap. When $P(n)$ is small, the passbands of the two filters overlap significantly, causing positive gain of the combined filter. As $P(n)$ grows, less of the passbands overlap, causing negative gain. However, the gain is normalized, as described below. Normalizing the filters cause 0 dB gain, but different bandwidths. Low levels cause narrow filters, while high levels cause wide filters.

Irino and Patterson [8] described a filter architecture that dynamically estimate the signal level. This architecture can be seen in Figure 3.6, and is what constitutes the DCGC. The figure is a modified version of a figure by Irino and Patterson [8].

Figure 3.6 shows that the signal level $P(n)$ is estimated from two signals. One is the output of a higher channel PCG, $s_1(n)$, which is reused in the corresponding higher

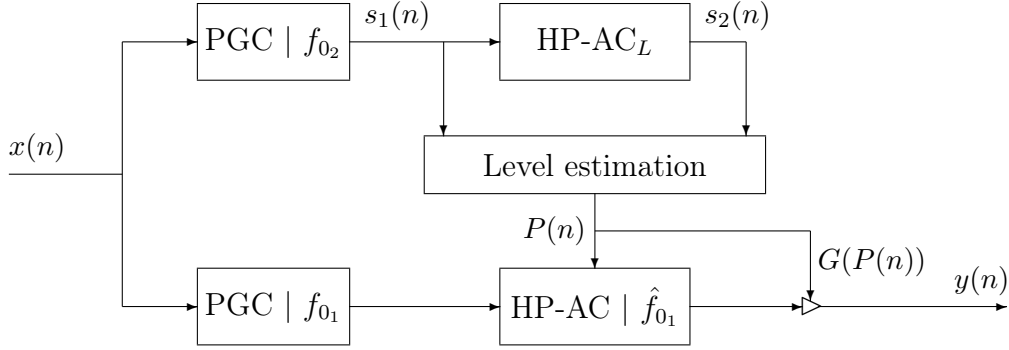


Figure 3.6: Architecture of the dynamic compressive gammachirp filter. The upper PGC is that of the next channel DCGC, that is, $f_{0_2} > f_{0_1}$.

channel DCGC. $s_2(n)$ is a high-pass filtered version of $s_1(n)$. The upper high-pass filter, HP-AC_L , is level independent. Its purpose is to account for different rates of suppression growth [8].

Table 3.2: DCGC filter coefficients.

	PCG	HP-AC	HP-AC _L
c	-2.96	2.2	2.2
b	1.81	2.17	2.17
$f_{\text{RAT}}(n)$	N/A	$0.466 + 0.0109 \cdot P(n)$	1.08

$$\bar{s}_1(n) = \max\{\bar{s}_1(n-1)e^{-(T/\lambda)\cdot\ln 2}, \max\{s_1(n), 0\}\} \quad (3.15)$$

$$\bar{s}_2(n) = \max\{\bar{s}_2(n-1)e^{-(T/\lambda)\cdot\ln 2}, \max\{s_2(n), 0\}\} \quad (3.16)$$

$P(n)$ is estimated for each sample, and depends on a number of weighting coefficients in addition to $\bar{s}_1(n)$, $\bar{s}_2(n)$. Equation (3.15) and (3.16) shows that $\bar{s}_1(n)$ and $\bar{s}_2(n)$ tracks the positive output of the filters as they grow. However, after a peak the output is no longer tracked, and the estimate decays according to the half-life λ [8]. Equation (3.17) was supplied by Irino and Patterson [8]. It is modified by adding one to the value before the logarithm is taken. This is done to ensure that the smallest value of $P(n)$ is 0 dB.

$$P(n) = 20 \log_{10} \left(w_L \cdot a_{\text{RL}} \left(\frac{\bar{s}_1(n)}{a_{\text{RL}}} \right)^{v_{L_1}} + (1 - w_L) \cdot a_{\text{RL}} \left(\frac{\bar{s}_2(n)}{a_{\text{RL}}} \right)^{v_{L_2}} + 1 \right) \quad (3.17)$$

$$a_{\text{RL}} = 10^{P_{\text{RL}}/20} \quad (3.18)$$

Irino and Patterson [8] supplied the weighting coefficients w_L , v_{L_1} , v_{L_2} , which are shown in Table 3.3. The table also shows the reference level in dB, P_{RL} , and the half-life λ .

Table 3.3: Level estimation coefficients.

P_{RL}	λ	v_{L_1}	v_{L_2}	w_L
50 dB	0.5 ms	1.5	0.5	0.5

The gain of the CGC is approximately equal for all frequencies, meaning it only depends on the estimated signal level $P(n)$. By constraining $P(n)$ to a finite number of values, the normalization coefficients can be computed ahead of processing. In this implementation, this is done by computing the coefficients in Matlab and storing them in a comma-separated values (CSV) file.

$$G(P) = \frac{1}{\max\{|\text{FFT}\{\text{CGC}(P)\}|\}} \text{ for } 0 \leq P \leq 105 \quad (3.19)$$

Equation (3.19) explains how the gain table is computed. The term $\text{CGC}(P)$ is a Matlab function that returns the impulse response of a CGC filter determined by P . P is used as a signal level to set the input frequency of the HP-AC filter.

Pre-computing a gain table like this implies putting constraints on the signal level $P(n)$. One constraint is that $P(n)$ may only take integer values. This is because its value is used as the index of the normalization coefficient. Another constraint is that if $P(n)$ must assume a finite number of values, it needs a maximum value. The input signal's datatype is 16 bit signed integer, which has a maximum value of 32768. Inserting this number into (3.17) instead of \bar{s}_1 and \bar{s}_1 produces a value of approximately 105 dB. This means that $P(n)$ is constrained to a maximum value by the datatype and (3.17).

HPACFILTERING filters a frame through a dynamic HP-AC filter. The procedure measures the signal level by filtering the output of the next PGC in the filter bank, F_N , through a HP-AC_L filter. A HP-AC structure is the procedure's input, along with two frames F and F_N . F is the output of the PGC filter. The structure contains four static AC₂ structures that together compose the HP-AC_L filter. It also contains four dynamic AC₂ structures that form the dynamic HP-AC filter. Additionally the previously computed \bar{s}_1 and \bar{s}_1 values are kept by the structure.


```

1: procedure HPACFILTERING(HPAC, F, FN)
2:   for  $n = 0 \rightarrow L - 1$  do
3:      $t \leftarrow$  ACFILTERING(HPACL1, FN( $n$ ))
4:      $t \leftarrow$  ACFILTERING(HPACL2,  $t$ )
5:      $t \leftarrow$  ACFILTERING(HPACL3,  $t$ )
6:      $t \leftarrow$  ACFILTERING(HPACL4,  $t$ )
7:      $t \leftarrow t \cdot N_{AC_L}$ 

8:   Compute  $P$  according to (3.15), (3.16) and (3.17)
9:   Update filter coefficients

10:   $F(n) \leftarrow$  ACFILTERING(HPAC1, F( $n$ ))
11:   $F(n) \leftarrow$  ACFILTERING(HPAC2, F( $n$ ))
12:   $F(n) \leftarrow$  ACFILTERING(HPAC3, F( $n$ ))
13:   $F(n) \leftarrow$  ACFILTERING(HPAC4, F( $n$ ))
14:   $F(n) \leftarrow F(n) \cdot N_{AC} \cdot G(P)$ 
15:  end for
16:  return F
17: end procedure

```

3.5.3 THE PASSIVE GAMMACHIRP SYNTHESIS FILTER

As the PGC analysis filter bank consist of IIR filters, its phase is nonlinear. This can make the temporal positions of sounds change nonlinearly relative to the positions in the unfiltered signal. Additionally it causes destructive interference when the subbands are added together. To avoid these problems, PGC synthesis filters are used for phase compensation.

A common technique for obtaining zero phase distortion of IIR filters is called forward-backwards filtering [23]. This technique works by first filtering an entire signal through a filter. The filter output is then reversed, and passed through the same filter again. Reversing the output of the second filtering results in a signal with zero phase distortion [23]. A drawback of this strategy is that it is not applicable to real-time systems, as the entire signal is required per filtering.

Synthesis filtering is implemented as a version of forward-backward filtering. It is based on a technique for realizing linear phase IIR filters developed by Powell and Chau [28]. Their technique involves filtering segments of a signal. The segment length, L , is

crucial. IIR filters can only be made to have linear phase if their impulse response is finite due to sampling, or if a scheme is developed that truncates the impulse response [28]. The idea of their technique is that at after the L th sample, a filter's impulse response decays far enough to assume values that are negligible [12]. At this point the impulse response appears to have finite length.

The technique roughly works in the following manner. Each frame is reversed and convolved with the impulse response of an IIR filter. The output of the convolution is overlap-added with the outputs of the adjacent frames, and reversed again. Lastly the resulting frame is again convolved with the impulse response [12]. This results in a linear phase filter whose magnitude response is the square of the IIR filter's magnitude response [28].

A direct implementation of Powell and Chau's technique was found to not suit this implementation sufficiently. The reason for this was a desire to use cyclic buffers, and not use the overlap-add method. However, the principle of filtering and reversing segments, and the frame length remains. In this case the frame length is determined by the PGC filter with the lowest input frequency; 100 Hz. Its impulse response was assumed to take on negligible values after approximately 2000 samples, leading to a frame length of 2048. The modified technique uses an overlap-save method, and is described by the following steps. Its input is assumed to be the output of a PGC analysis filter.

1. Create a frame of length $3L/2$. The frame consists of the reversed input frame concatenated with the reversed last half of the previous frame.
2. Pass the frame through a PGC filter.
3. Discard the first $L/2$ samples of the filter output.
4. Reverse the remaining samples.

An advantage of this technique, compared to a direct implementation of Powell and Chau's, is that the processing delay is shorter. Where Powell and Chau's technique has a delay of $3L + 1$ samples [12], this implementation has a delay of $L/2$. Whether the linear phase accuracy of this technique is comparable to that of Powell and Chau is not examined further. For this implementation it is sufficient that the technique avoids destructive interference when the subbands are added, and that the components of the signal have the same relative positions to each other as in the unfiltered signal.

PGCSYNTHESISFILTERING shows how a frame is filtered through the filtering scheme

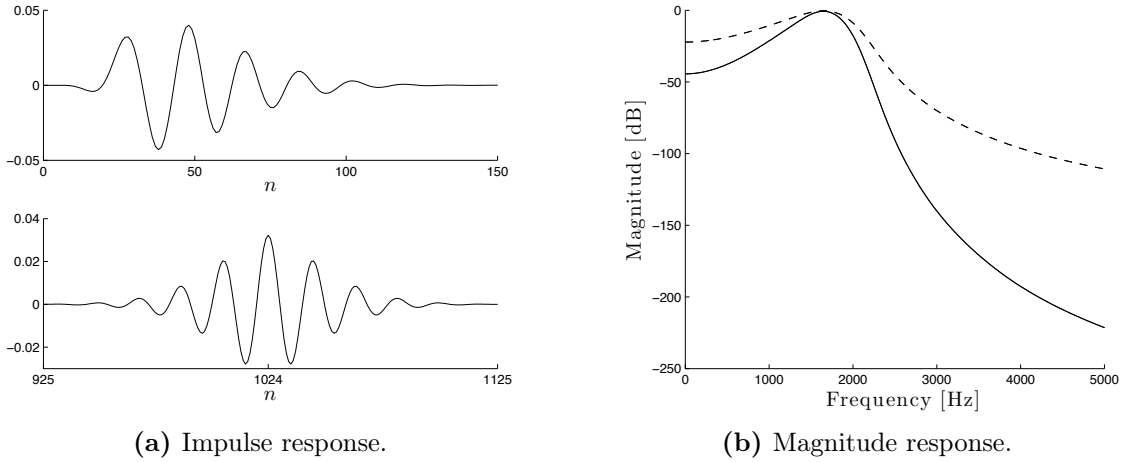


Figure 3.7: Impulse and magnitude response of a PGC analysis filter (top a). Impulse response of a PGC synthesis filter (bottom a). Magnitude response of an analysis PGC (dashed b), and the magnitude response of a synthesis PGC (solid b). The input frequency of the PGC is 2000Hz, and the frame length is 2048.

described above. It works on the same PGC structure as PGCANALYSISFILTERING. Another attribute of the structure is an array (PGC_{PR}) containing $L/2$ samples. These samples are the reversed last half of the previous frame.

```

1: procedure PGCSYNTHESISFILTERING(PGC, F)
2:   for  $n = 0 \rightarrow 3L/2 - 1$  do
3:     if  $n < L$  then
4:        $x \leftarrow F(L - 1 - n)$ 
5:     else
6:        $x \leftarrow \text{PGC}_{\text{PR}}(n - L)$ 
7:        $\text{PGC}_{\text{PR}}(n - L) \leftarrow F(2L - 1 - n)$ 
8:     end if
9:      $x \leftarrow \text{GAMMATONEFILTERING}(\text{PGC}_{\text{GT}}, x)$ 
10:     $x \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC1}}, x)$ 
11:     $x \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC2}}, x)$ 
12:     $x \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC3}}, x)$ 
13:     $x \leftarrow \text{ACFILTERING}(\text{PGC}_{\text{AC4}}, x)$ 
14:    if  $n \geq L/2$  then
15:       $F(3L/2 - 1 - n) \leftarrow x \cdot N_{\text{AC}}/G_{\text{PGC}}$ 
16:    end if
17:  end for
18:  return F
19: end procedure

```

3.6 SIMULATING LOUDNESS RECRUITMENT

As mentioned in Section 2.2.2, loudness recruitment (LR) makes quiet sounds quieter, while loud sounds are perceived as equally loud as in normal hearing. Moore and Glasberg [25] simulated LR by altering the envelope of a signal. Their method was implemented in *Simulation of Reduced Frequency Selectivity and Loudness Recruitment* [1]. The result was that the difference between high and low parts of a signal was made larger.

In this implementation, a different approach to LR simulation has been taken. As there is already an estimate of the signal level, by (3.17), LR can be simulated in relation to that level. This means that the signal level in combination with a hearing loss can be used to scale each sample.

LR is simulated during the dynamic HP-AC filtering. After the signal level has been estimated, a scaling factor $F_{LR}(n)$ is computed, and the output sample is scaled by it. The scaling factor is based on the function that produced Figure 2.5 in Section 2.2.2. Scaling the signal by $F_{LR}(n)$ results in both elevated thresholds and loudness recruitment.

$$F_{LR}(n) = \begin{cases} 1 & \text{if } P(n) \geq R_C \\ \frac{R_C}{R_C - HL_{OHC}(f_0)} \frac{P(n) - HL_{OHC}(f_0)}{P(n)} & \text{if } HL_{OHC}(f_0) \leq P(n) < R_C \\ 0 & \text{if } P(n) < HL_{OHC}(f_0) \end{cases} \quad (3.20)$$

The component $HL_{OHC}(f_0)$ describes the hearing loss at a frequency f_0 , due to OHC damage. R_C describes the point, in dB SPL, where recruitment is complete. Recruitment is complete at the same level where the active mechanism no longer contributes, often at 90-100 dB SPL [25]. $R_C = 90$ dB SPL is used in this implementation.

Figure 3.8 shows a diagram of how the dynamic HP-AC filter is implemented with LR simulation. The block named LR computes the scaling factor $F_{LR}(n)$.

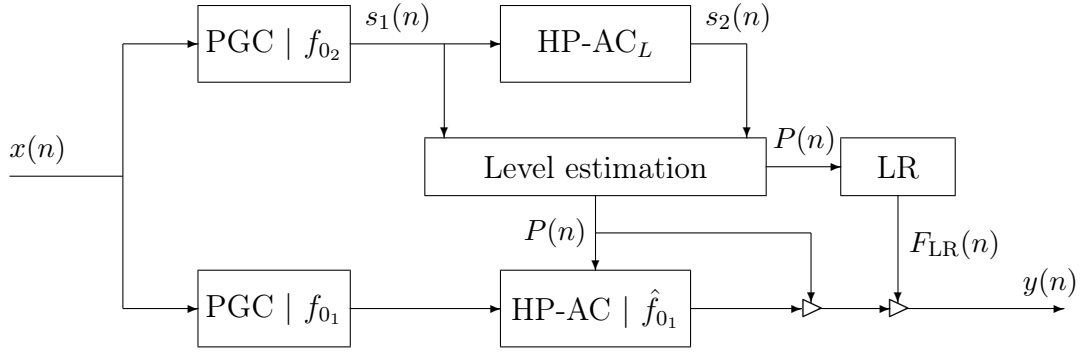


Figure 3.8: Architecture of the DCGC with LR simulation.

3.7 SIMULATING REDUCED FREQUENCY SELECTIVITY

Reduced frequency selectivity (RFS) has been simulated by various methods in the past. Baer, Glasberg, Moore and Simpson [24; 27] performed spectral smearing on a signal’s magnitude spectrum. Another method has been to smear the spectral envelope of a signal [26].

Most of these methods are computationally expensive. They require Fourier transforms and convolutions for each subband, making them unsuited for a real-time system. This implementation is based on filter widening. Glasberg and Moore [30] concluded that the auditory filters of people with hearing impairments often had longer low-frequency tails than normal, a feature mimicked in RFS simulation.

Figure 3.9 shows the magnitudes of a number of CGC filters. The filters are created by widening the bandwidth of the HP-AC filter by a factor $W(f_0)$. Moore and Glasberg [11] created a model of loudness perception where they scaled the bandwidth of the auditory filters by this factor to account for RFS.

$$W(f_0) = \begin{cases} 10^{0.01HL_{\text{OHC}}(f_0)} & \text{if } f_0 \geq 500\text{Hz} \\ 10^{0.01HL_{\text{OHC}}(f_0)(1-(f_0/1000-0.5)^2/1.23)} & \text{if } f_0 < 500\text{Hz} \end{cases} \quad (3.21)$$

The widening factor depends on hearing loss due to OHC damage, $HL_{\text{OHC}}(f_0)$, at a particular frequency. $W(f_0)$ is constrained by having the upper limit of 3.8 if $f_0 > 500$ Hz and $HL_{\text{OHC}}(f_0) > 57.6$ dB. If $f_0 < 500$ Hz and $HL_{\text{OHC}}(f_0) > 57.6$ dB, the upper limit is 2.7. These limitations are based on the assumption that after 57.6 dB the

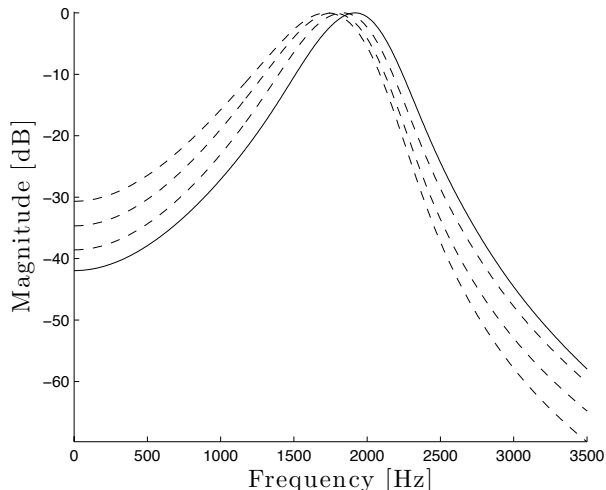


Figure 3.9: Magnitude response of a CGC filter with an input frequency of 2 kHz and $P(n) = 60$ dB. The dashed lines show the filter shapes for different values of $HL_{\text{OHC}}(f_0)$. Inspecting the figure from the left, the uppermost line is according to $HL_{\text{OHC}}(f_0) = 57$ dB. The following lines are according to losses of 40 dB and 20 dB. The solid line shows the shape without hearing loss.

response of the basilar membrane is unaffected by the OHCs [1; 11]. Widening the HP-AC filters cause the gain they produce to change. This means that the gain table from (3.19) must also depend on hearing loss, leading to a two-dimensional gain table. The gain table means that the hearing loss must also be constrained to assume a finite number of integers. Therefore $HL_{\text{OHC}} = 57$ dB is chosen as the maximum value.

$$G(P, HL_{\text{OHC}}) = \frac{1}{\max\{|\text{FFT}(\text{CGC}(P, HL_{\text{OHC}}))|\}} \quad (3.22)$$

The idea of simulating RFS in this manner is due to the properties of the DCGC. When the filters are widened, the HP-AC_L will pass more low-frequency components. This could lead to the estimated level being different than what is actually the case. Setting the coefficients of the HP-AC filters by the false level could make them pass unwanted spectral components, while wanted components are filtered out. In addition to potentially estimating false levels, the cutoff frequencies of the filters change with hearing loss. A high loss lowers the cutoff frequency, and could lead to wanted components being removed. The tail of the filter is also elevated, which could cause the filter to pass more low-frequency noise. These properties could cause the filtering process to produce the effects of RFS by increasing the amount of two-tone suppression.

3.8 REDUCING COMPUTATIONAL COST

Real-time systems have time constraints. If audio is to be processed in real-time, the processing must be quicker than the playback. This system, that process frames, must process a frame faster than a frame is played back. On a sample level this means that

each sample must be processed faster than $1/f_s$ seconds. This section describes three strategies that are implemented to reduce the computational cost, and thereby the processing time.

3.8.1 RESAMPLING

When the input signal has been run through a PGC analysis filter it is, for many channels in the filter bank, highly oversampled. The lowest filter runs on an input frequency of 100 Hz. If that filter has a stop band after 200Hz, and the sampling frequency is 32 kHz, the subband is oversampled by a factor of 80. This means that the filter can be downsampled after the analysis filter, and upsampled before the synthesis filter. By doing this, the analysis PGC works as a decimation filter, and the synthesis PGC works as an interpolation filter.

The procedure HPACFILTERING from Section 3.5.2 implies that the level-estimation HP-AC runs on the same sampling frequency as the dynamic HP-AC. This means that if these two filtering processes are to be performed within the same procedure, they must be decimated by the same factor. As the input frequency of HP-AC_L is higher than that of the dynamic HP-AC, the sampling frequency must be set according to HP-AC_L. Figure 3.9 shows that a CGC filter without hearing loss has a higher cut off frequency than filters where $HL_{\text{OHC}}(f_0)$ is non-zero. This means that the decimation factor must be determined according to the input frequency of HP-AC_L without hearing loss.

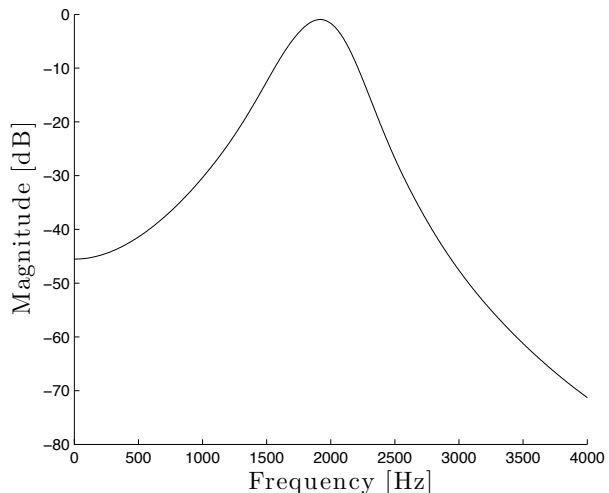


Figure 3.10: Magnitude response of the composite filter created by a PGC and HP-AC_L filter. The input frequency of the PGC is 2 kHz, causing the input frequency of the HP-AC_L to be 2.47 kHz.

The PGC and the HP-AC_L create the composite filter in Figure 3.10. By choosing a dB value where this filter is sufficiently close to zero, the decimation factor can be computed. In this implementation the value was chosen to be -50 dB. Figure

3.10 shows that the frequency where this value occurs is approximately 3 kHz. Tying this frequency to the input frequency of the HP-AC_L, \hat{f}_L , its ERB and the sampling frequency gives the decimation factor, D .

$$D = \left\lceil \frac{f_s}{\hat{f}_L + 2 \text{ERB}(\hat{f}_L)} \right\rceil \quad (3.23)$$

The denominator of (3.23) is one of the lowest frequencies where the composite filter is below -50 dB. For $\hat{f}_L = 2.47$ kHz, the denominator is 3.05 kHz, giving a decimation factor of 10.

Resampling is implemented by changing the dynamic HP-AC filtering slightly. During filtering, HPACFILTERING only processes every D th sample, setting the remaining samples to zero. This produces a result that is equal to only keeping every D th sample, filtering the signal, and inserting $D - 1$ zero-samples between every D th sample.

3.8.2 LOOKUP TABLES

When filtering the signal through the dynamic HP-AC filter, the signal level is estimated for each sample. This means that the filter coefficients need to be updated every time the signal level changes, potentially for every sample. Computing the coefficients of the HP-AC filter is an expensive operation, mostly due to the use of trigonometric functions.

Lookup tables are tables that contain precomputed values, like the gain table in (3.19). By using lookup tables, the cost of computing the filter coefficients are removed entirely from the filtering process. Since the signal level $P(n)$ is confined to assume a finite number of integer values, it can be used as an index for looking up the feedback, feedforward and normalization coefficients of the HP-AC.

The lookup tables are made from three arrays. One array contains the feedback coefficients, another the feedforward, and the last contains the normalization coefficients N_{AC} . Before starting the filtering process, the lookup tables are computed. This means that during filtering, it is only necessary to read values from the tables.

3.8.3 AVOIDING TRIGONOMETRIC FUNCTIONS

As mentioned above, trigonometric function such as sine and cosine are expensive operations. After lookup tables were implemented, sine and cosine computations still

consumed about 20% of the processing time. These computations ocured during the gammatone filtering, as it needs one sine and one cosine computation per sample.

Ma et al. [4] implemented a gammatone filter that avoided the extensive use of trigonometric functions. They exploited the fact that an exponential function can be written as a product of two exponential functions. When frequency shifting the signal, they used that:

$$e^{-j2\pi n f_c T} = e^{-j2\pi f_c T} \cdot e^{-j2\pi f_c T(n-1)}. \quad (3.24)$$

This way the first term could be computed in advance, while the last term was the result of the previous multiplication [4]. By applying this principle to trigonometric functions using Euler's formula, the result is:

$$\sin(2\pi n f_c T) = \sin(2\pi f_c T + 2\pi f_c T(n-1)) \quad (3.25)$$

$$\cos(2\pi n f_c T) = \cos(2\pi f_c T + 2\pi f_c T(n-1)). \quad (3.26)$$

Expanding these two equations using the sum and difference formulas gives:

$$\begin{aligned} \sin(2\pi f_c T + 2\pi f_c T(n-1)) &= \sin(2\pi f_c T) \cos(2\pi f_c T(n-1)) \\ &\quad + \cos(2\pi f_c T) \sin(2\pi f_c T(n-1)) \end{aligned} \quad (3.27)$$

$$\begin{aligned} \cos(2\pi f_c T + 2\pi f_c T(n-1)) &= \cos(2\pi f_c T) \cos(2\pi f_c T(n-1)) \\ &\quad - \sin(2\pi f_c T) \sin(2\pi f_c T(n-1)). \end{aligned} \quad (3.28)$$

From this it can be seen that $\sin(2\pi f_c T)$ and $\cos(2\pi f_c T)$ can be computed in advance. Leading to only two sine and cosine calls per gammatone filter. The remaining terms are produced by the previous sample. An improved procedure describing how a sample is filtered through a gammatone filter can be seen below. The gammatone structure that is passed as a reference holds the same attributes as in GAMMATONEFILTERING from Section 3.3. But it also contains the four values \sin_0 , \cos_0 , \sin_p and \cos_p . Where $\sin_0 = \sin(2\pi f_c T)$ and $\cos_0 = \cos(2\pi f_c T)$. The previous sine and cosine computations are stored in \sin_p and \cos_p , which are initially zero and one.

```

1: procedure GAMMATONEFILTERING(GT, s)
2:    $e_R \leftarrow \cos_p$ 
3:    $e_I \leftarrow -\sin_p$ 
4:    $\cos_p \leftarrow \cos_0 \cdot e_R - \sin_0 \cdot \sin_p$ 
5:    $\sin_p \leftarrow \sin_0 \cdot e_R + \cos_0 \cdot \sin_p$ 
6:    $s_R \leftarrow s \cdot e_R$ 
7:    $s_I \leftarrow s \cdot e_I$ 
8:   for  $n = 0 \rightarrow N - 1$  do
9:      $t_R \leftarrow \text{READ}(\text{GT}_{BR})$ 
10:     $t_I \leftarrow \text{READ}(\text{GT}_{BI})$ 
11:     $s_R \leftarrow s_R + \text{GT}_A(n) \cdot t_R$ 
12:     $s_I \leftarrow s_I + \text{GT}_A(n) \cdot t_I$ 
13:   end for
14:    $\text{WRITE}(\text{GT}_{BR}, s_R)$ 
15:    $\text{WRITE}(\text{GT}_{BI}, s_I)$ 
16:    $k \leftarrow k + 1$ 
17:   return  $2 \text{GT}_B \cdot (t_R \cdot e_R + t_I \cdot e_I)$ 
18: end procedure

```

3.9 THE FINISHED SYSTEM

A block diagram of the complete filter bank, with hearing impairment simulation, is shown in Figure 3.11. The number of subbands, or filters in each filter bank, is important to the result of the processing. Each filter bank, except the PGC analysis filter bank, consists of M filters. The analysis filter bank contains $M + 1$ filters, where the $M + 1$ th filter is used to compute the signal level for the M th HP-AC filter. The number of filters is determined by how flat the resulting magnitude response of the PGC analysis and synthesis filter bank is. Another factor that contributes to the number of filters is processing time. Many filters means a long processing delay. If the number of filters are too high, the processing delay may be longer than the playback time, which will lead to "glitches" in the playback. A value of $M = 25$ was found to produce a sufficiently flat spectrum, while processing the signal faster than the playback.

The position of the filters are computed as follows. Select a minimum and maximum input frequency. Compute the ERBS number of these frequencies. Place $M - 2$ equally spaced points between the two ERBS values. Apply the reciprocal of (2.2) to each point,

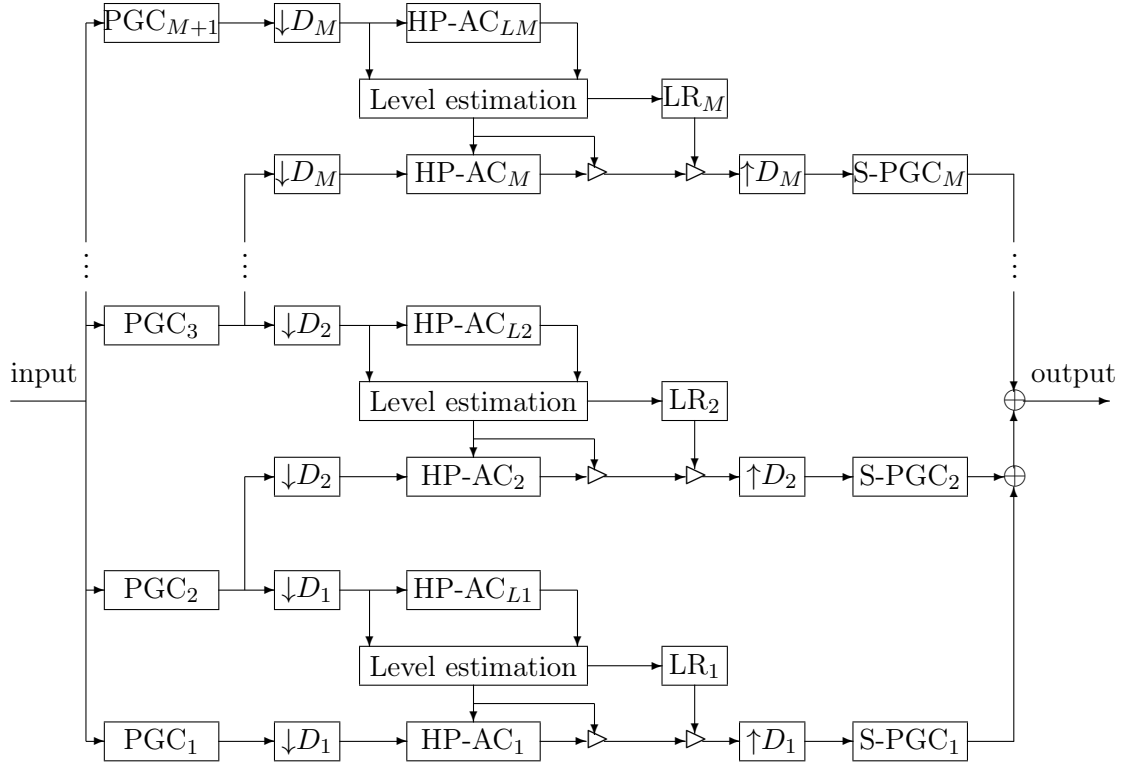


Figure 3.11: Diagram showing the finished system.

yielding the input frequencies, or positions of the filters [1]. For the PGC analysis filter bank, the $M + 1$ th filter is positioned at $ERBS(f_{\max}) + s$, where s is the spacing between the input frequencies on the ERB-scale. The minimum input frequency of the implemented DCGC filter bank is 100Hz, and the maximum is 8 kHz.

The output of the analysis and synthesis PGC filter bank can be seen in Figure 3.12. Each filter is attenuated by 7 dB to give the summed output, dashed line, 0 dB gain.

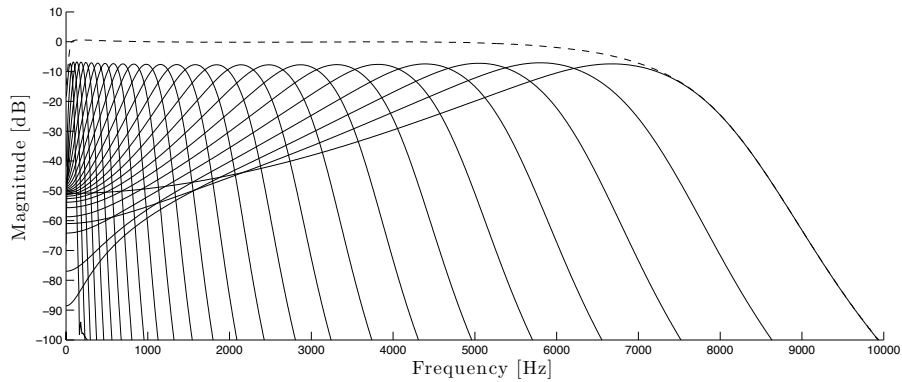


Figure 3.12: Output of analysis and synthesis PGC filter banks.

3.9.1 THE C PROGRAM

The hearing loss simulator was implemented using the C programming language in Xcode 3.2.4 IDE for Mac OS X. It was compiled using GCC 4.2. C was chosen since it is a widely used and popular language for signal processing applications. The finished executable is run as a command line tool on Mac OS X computers. For a description of how to use the program, refer to Appendix B.

The implementation contains six C structures. Each structure contains the variables needed to perform a task. An example of this is when a segment is passed through a PGC analysis filter: The function `PGCaFiltering` is called with an input frame and a PGC structure as input. The PGC structure contains filter coefficients, cyclic buffers and other variables, defining a PGC filter. A note regarding the PGC structure is that as it contains the necessities to perform gammatone and LP-AC filtering, these filters are not represented by explicit structures, as implied in Section 3.3 and 3.4. An overview of the structures, and how they are connected can be seen in Table 3.4.

Table 3.4: Overview of C structures.

Structure name	Description	Contains
CYCLICBUFFER2	A cyclic buffer of length two.	
CYCLICBUFFER4	A cyclic buffer of length four.	
PGC	Contains the coefficients and buffers needed to perform PGC analysis and synthesis filtering.	4 CYCLICBUFFER2, 2 CYCLICBUFFER4
HPAC	Contains the coefficients and buffers needed to perform HP-AC _L filtering, estimate the signal level, and perform dynamic HP-AC filtering. Enables LR and RFS simulation.	8 CYCLICBUFFER2
PROCESSOR	Acts as the DCGC filter bank. Contains a number of PGC and HPAC structures.	26 PGC, 25 HPAC
AUDIO	Enables playback of audio.	1 PROCESSOR

The program contains a number of functions, most of which takes a structure as input. Figure 3.13 shows a sequence diagram describing which functions are called when a frame is processed. The top boxes show what structures the functions run on.

The standing rectangles show the duration of a function. Calls to PGC and HPAC are

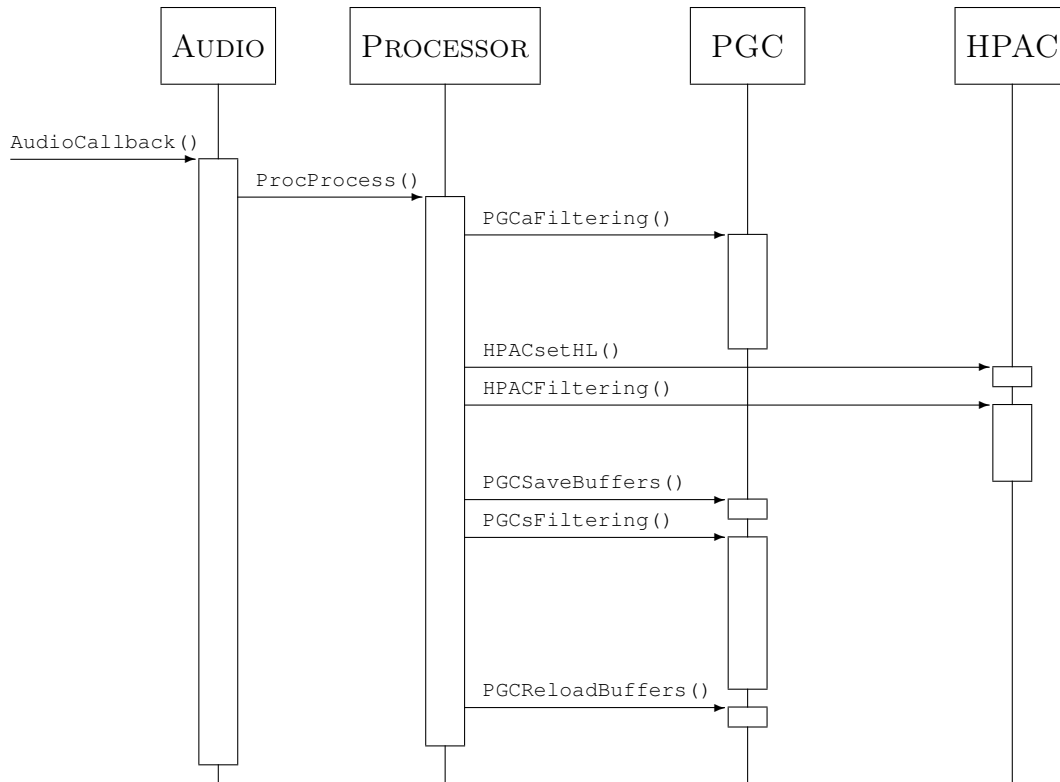


Figure 3.13: Function calls when processing a frame.

made several times per frame, where the number of calls depends on the number of filters in the filter bank. The three functions that perform filtering is `PGCaFiltering`, `PGCsFiltering` and `HPACFiltering`. These functions perform analysis PGC filtering, synthesis PGC filtering, and dynamic HP-AC filtering. The hearing losses are read from a CSV file prior to the first call to `PGCaFiltering`. When the hearing loss changes, the coefficients of the $HP-AC_L$ and $HP-AC$ structures need to be updated. `HPACsetHL` sets the variables in $HPAC$ related to hearing loss and recomputes the filter coefficients and lookup tables. The two functions `PGCSaveBuffers` and `PGCReloadBuffers` are implemented to allow both analysis and synthesis PGC filtering using the same structure. `PGCSaveBuffers` temporarily store the buffers and the trigonometric computations, `PGCReloadBuffer` reloads them when called. If not for these two functions, the analysis filtering would produce erroneous output, as values in the PGC structure are changed during synthesis filtering.

4

RESULTS

In this chapter, the results of the simulations are presented. The results interpreted from what theoretical perceptual effects hearing impaired people may experience. Interpreting the simulations to how they affect the intelligibility of speech is not done.

The point of having a real-time hearing loss simulator is to present the effects of hearing impairments within a short amount of time. Another point is to make the user able to change the hearing loss while he is listening. That the simulator is able to do this is a result in itself. Processing a frame is faster than playback, causing the user to hear a continuous stream of sound, regardless of how the system's parameters change. This does however depend on the computer the system runs on. Under normal circumstances, where the computer is fast enough, the delay from a parameter is changed until the processed frame is played back is 192 ms.

Another important result is that an unprocessed signal, a signal without LR or RFS simulation, sounds similar to the original sound file. The user must be able to compare normal hearing with impaired hearing. Figure 4.1a shows the spectrogram of the sound file used to create the results. The signal has a sampling frequency of 32 kHz and is low-pass filtered to make its highest frequency 8 kHz.

Figure 4.1a shows the spectrogram of the original sound file. Figure 4.1b shows the spectrogram after the sound file has been passed through the system without hearing loss processing. The two signals are almost the same, although the filtered signal is slightly weaker than the original. Additionally, Figure 4.1b shows that the system produces some spectral smearing, particularly noticeable above 4 kHz just before 1 second of duration.

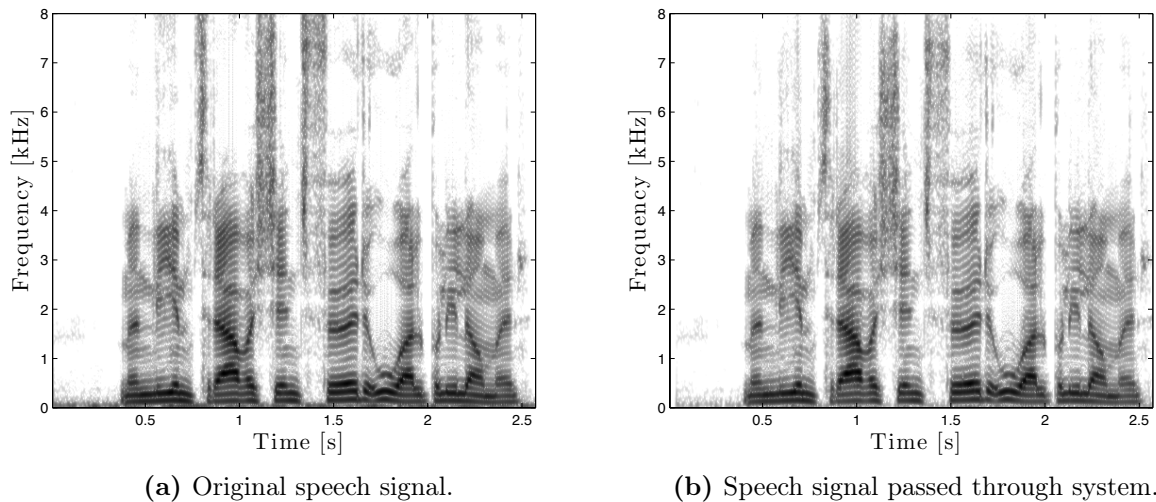


Figure 4.1: Figure 4.1a shows the spectrogram of a sound file containing the Norwegian sentence "Men limtre var kjent før OL på Lillehammer". Figure 4.1b shows the spectrogram of the signal after it has been passed through the system without hearing loss simulation.

4.1 RESULTS OF RFS SIMULATION

RFS can make speech harder to understand. This has many reasons, but affects the spectral content of a speech signal in various ways. The results of the simulations are interpreted by the spectral and perceivable degradation that can be expected by RFS.

Figure 4.2 shows the spectral envelope of the vowel /e/, pronounced as an /æ/, processed to simulate RFS. The unprocessed vowel is extracted from the speech signal in Figure 4.1b. All the spectral envelopes were estimated by modeling the signal as an autoregressive process in Matlab using the function `lpc`.

The figure shows that the largest peak is attenuated with increasing loss. Additionally the slope from 0 Hz to the peak grows less steep as the loss increases. At approximately 2.5 kHz there is another peak which almost disappears as the loss grows. These effects are consistent with the theoretical consequence of RFS where it blurs, or reduces the detail of a signal's spectral envelope.

The dashed lines show the spectral envelope when white gaussian noise is added to the signal before processing. Speech spoken in noisy environments is known to be harder to understand for both hearing impaired and normal people. By inspecting the dashed curves, it can be seen that the spectral envelopes are almost identical, but with an increasingly attenuated main peak. For the most part the attenuation increases with the loss, the exception being between the unprocessed and the second degree loss signal.

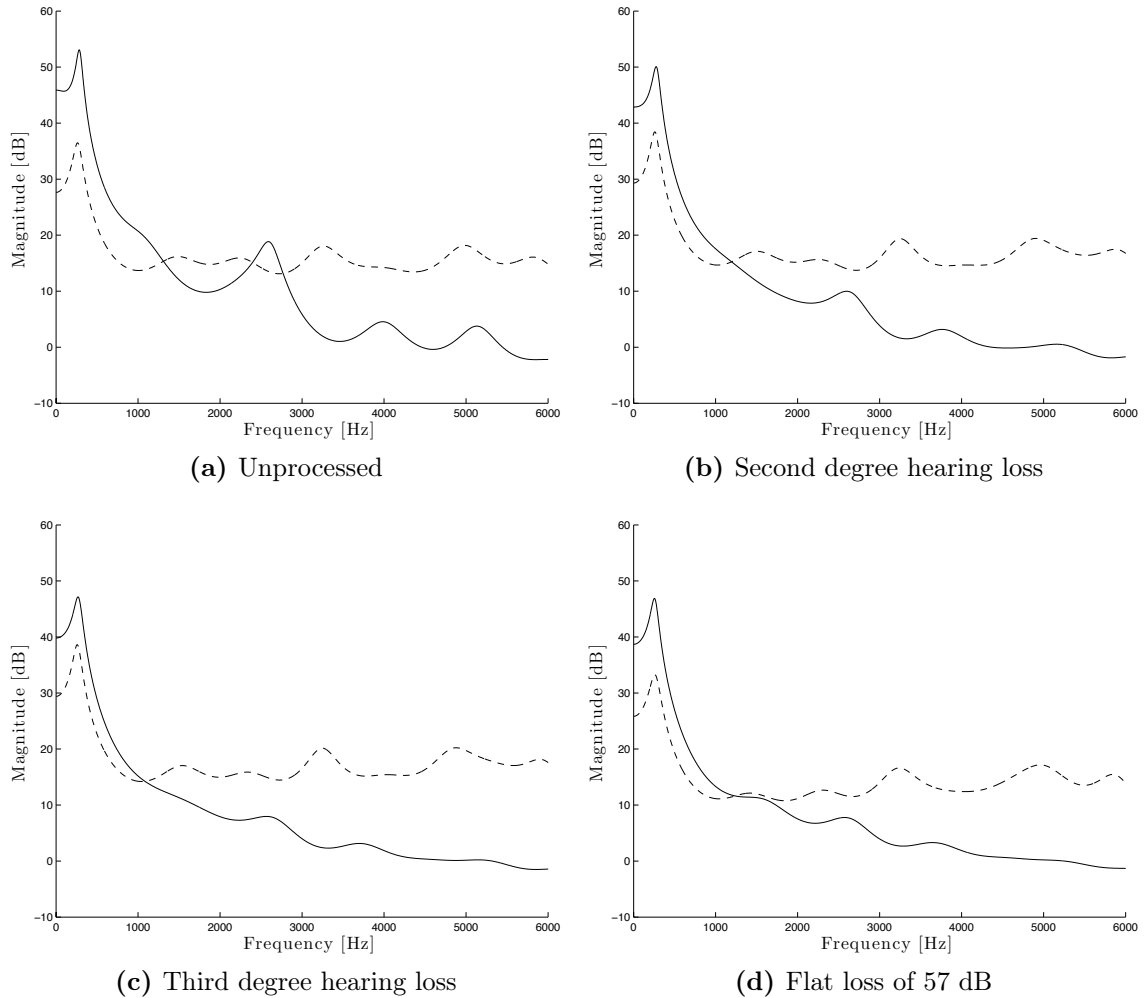


Figure 4.2: Spectral envelope of the vowel /æ/ from the signal shown in Figure 4.1b. The dashed line shows the vowel extracted from a noise contaminated version of the same signal.

Figure 4.4, 4.5 and 4.6 shows the spectrograms of the speech signal in Figure 4.1b, processed to simulate three losses: Second degree, third degree and a flat loss of 57 dB. The first degree loss is not included as it produced very small changes compared to the unprocessed signal. Left aligned figures show the speech signal in silence, while right aligned figures show the speech signal in noise.

The figures show that the signal is attenuated with increasing loss. It is particularly the weak spectral components that are attenuated, indicating that this method of simulating RFS produces some masking effects. These effects increase with hearing loss, which is a common consequence of RFS.

Hearing the difference between the processed signals is difficult, and the sentence is in every case intelligible. However, when comparing the unprocessed signal with the signals processed to simulate severe losses, such as 57 dB, certain differences can be heard. The difference is particularly an attenuation of the higher frequencies, causing

the signals to sound low-pass filtered. Clearer differences can be heard in the signal with noise, especially the noise sounds different. Inspecting the spectrograms reveal that the noise is attenuated, but the characteristics of the noise also appears to change. The noise shown in Figure 4.3b sounds like white noise, but the noise in in Figure 4.6 appears to have more high-frequency components. This can also be seen in the figure, the spectral components around 5-6 kHz are somewhat stronger than the other noise components.

An interesting result of RFS simulation is that some components are attenuated, while others are not. This makes some sounds, such as the one that occurs at approximately 1 second, normally loud or even louder. In effect this means that the DCGC filter bank, in combination with the RFS simulation technique, produces effects related to LR. This will be discussed further in Chapter 5.

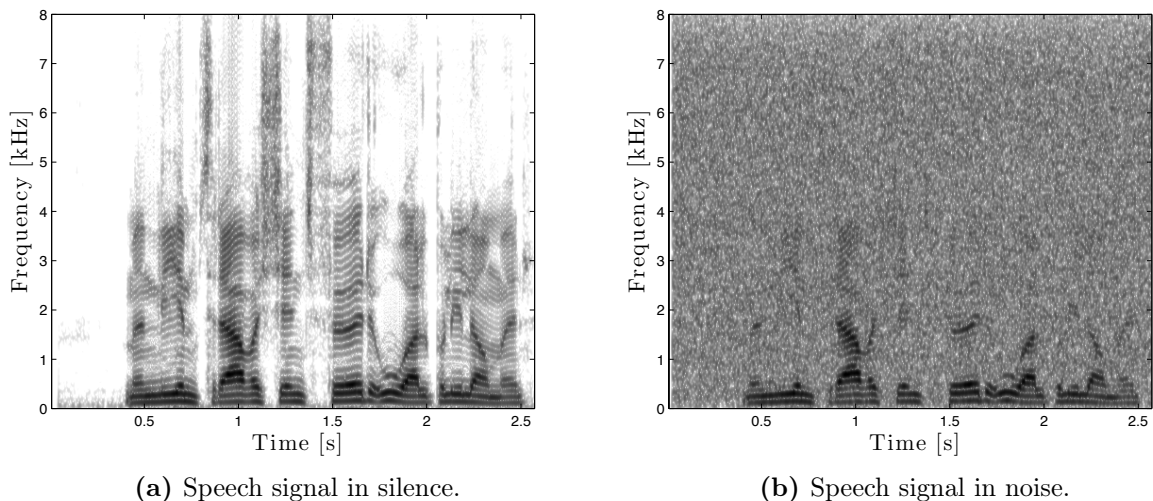


Figure 4.3: Spectrograms of the processed speech signal. Figure 4.3a is the same as 4.1b. The noise which is added to the signal in Figure 4.3b has the same level as the speech signal.

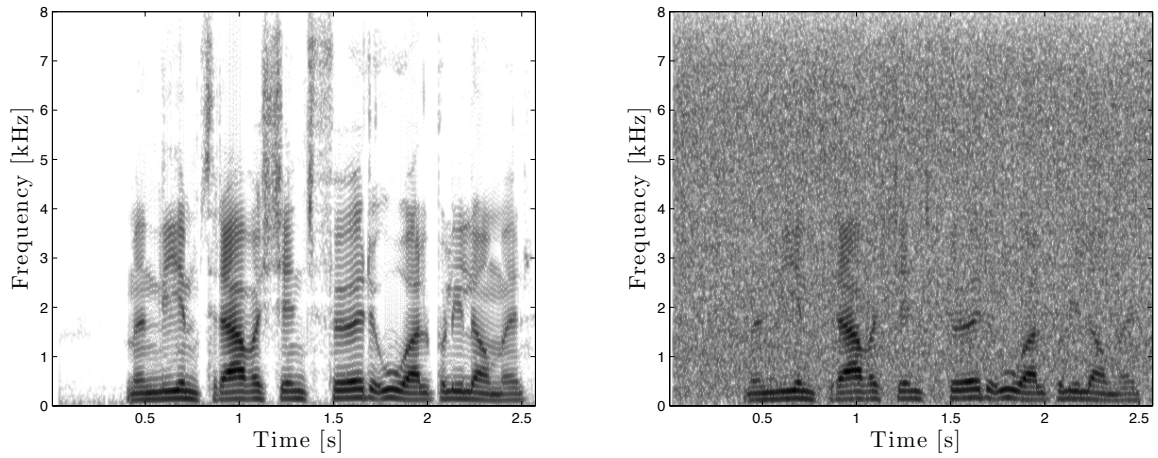


Figure 4.4: RFS simulation according to a second degree hearing loss.

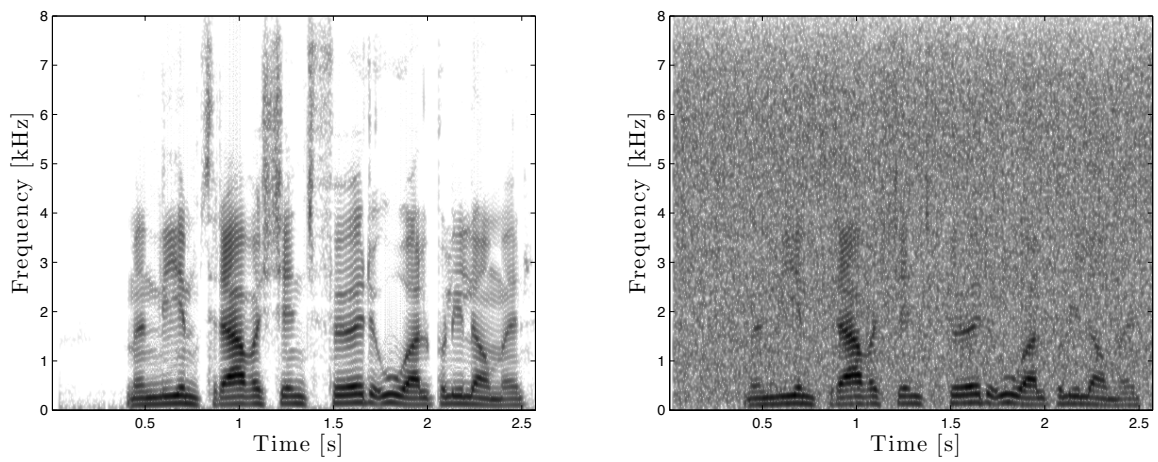


Figure 4.5: RFS simulation according to a third degree hearing loss.

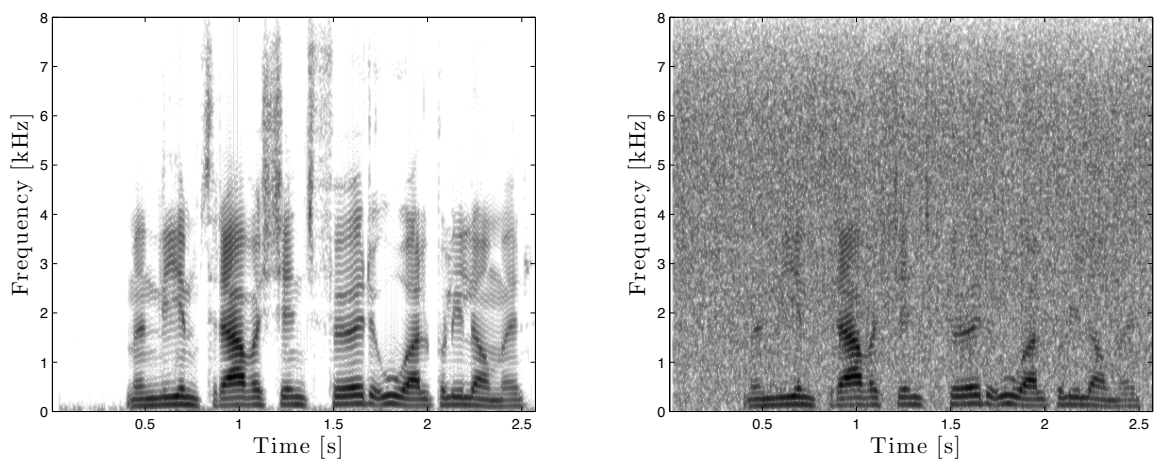


Figure 4.6: RFS simulation according to a flat hearing loss defined by $HL_{OHC} = 57$ dB.

4.2 RESULTS OF LR SIMULATION

Figure 4.8 - 4.12 shows the spectrograms and waveforms of the input signal, processed to simulate LR. The simulations are according to first, second and third degree losses, in addition to a flat loss of 30 dB. Simulating a flat loss of 57 dB causes the signal to almost completely disappear. The few remaining samples belong to the strongest vowels of the signal.

Simulating LR by the method described in Section 3.6 also accounts for elevated absolute thresholds. Perceptually this means that the sounds below the hearing loss, or threshold, is made inaudible by setting the samples to zero. However, sounds above the threshold are set to have levels that are closer to the levels of the original sounds. All spectrograms show that quiet sounds are attenuated more than loud ones. The time plots show that the entire signal is attenuated due to threshold elevation. Figure 4.12 shows the spectrogram of the signal after it has been processed to simulate a flat loss of 30 dB. As the loss is flat, weak sounds are equally attenuated for all frequencies. Comparing Figure 4.12 with 4.10a shows that the spectral component around the vowel /æ/, that occurs after 1 second, is attenuated. This causes the vowel to appear more abruptly. Combining this with attenuation of weak sounds causes the signal to fluctuate more than the unprocessed signal. The waveform also shows this, the sound climbs faster from 0 to the vowel's peak.

The simulations of non-flat losses sounds low-pass filtered, and they actually are. As the low frequencies have smaller losses, the slope of the function in Figure 2.5 is less steep. The gradual slopes cause less attenuation, leaving more samples close to their original value. As consonants contain more high frequency content than vowels, a high non-flat loss makes them prone to attenuation. Inspecting Figure 4.11 reveals that the consonant /t/ has almost vanished, while the vowel /æ/ at approximately 1 second is still relatively strong. The /t/ consists of the high-frequency stripe of energy at about 1.3 seconds. Both flat and non-flat losses attenuate this sound, but the following sound, an /f/ consisting of the noise-like energy between 1.3 and 1.5 seconds, has disappeared in both the flat and the third degree loss. This, and what was described above are common perceptual consequences of LR. Consonants are heavily attenuated or removed, while vowels are audible. Processing music leads to the same effects as described here. However, music often has more and stronger high frequency content than speech. This causes music processed with non-flat losses to fluctuate in a similar manner as speech does with flat losses.

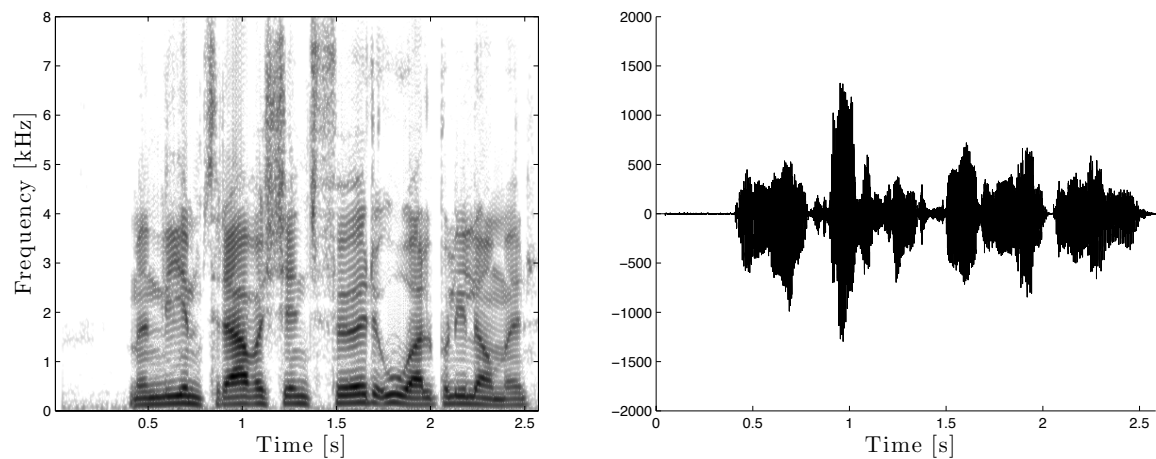


Figure 4.7: Unprocessed speech signal.

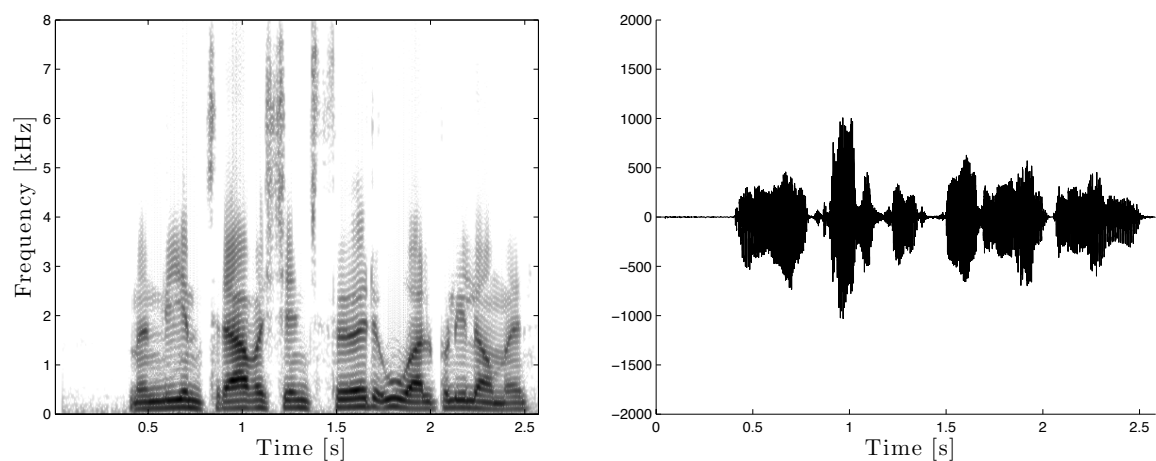


Figure 4.8: LR simulation according to a first degree hearing loss.

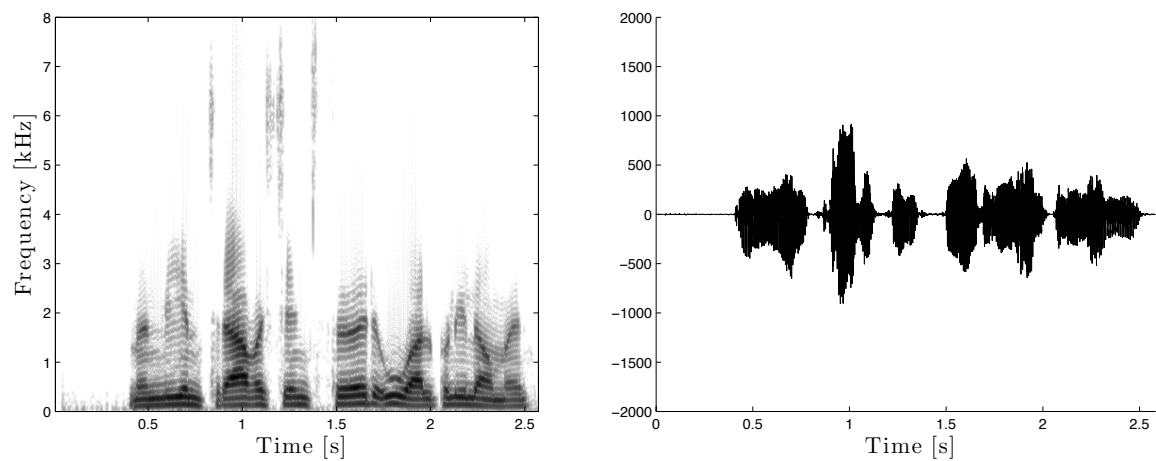


Figure 4.9: LR simulation according to a second degree hearing loss.

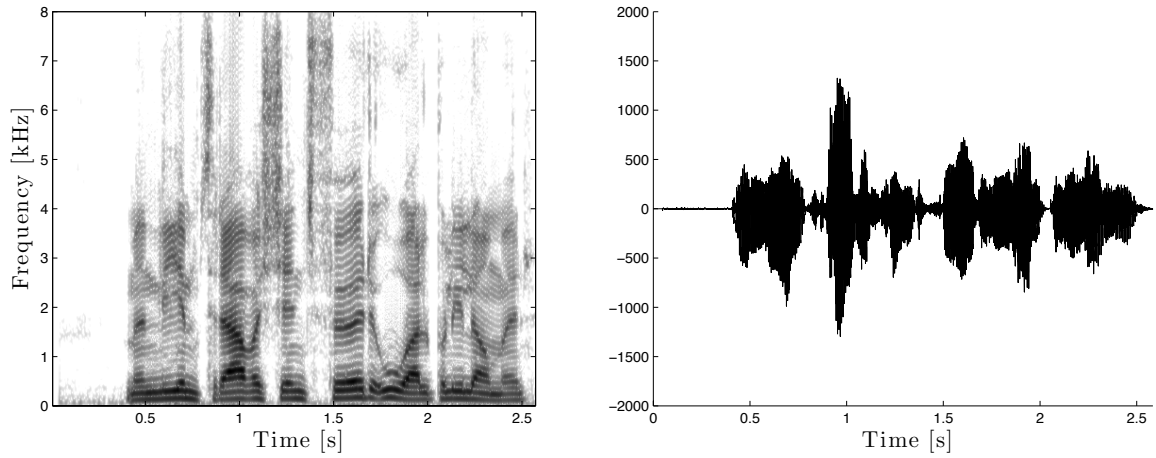


Figure 4.10: Unprocessed speech signal. Same figure as 4.7, inserted for comparison.

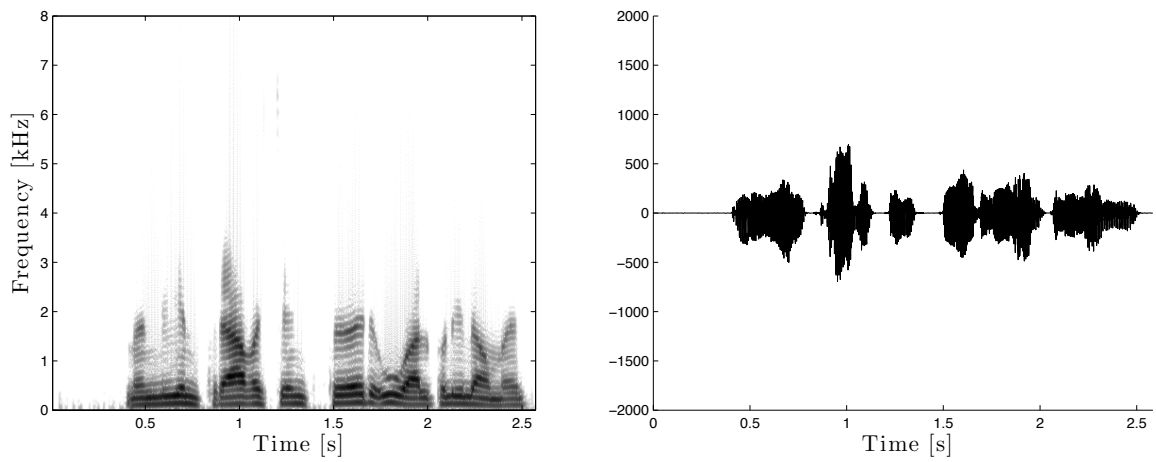


Figure 4.11: LR simulation according to a third degree hearing loss.

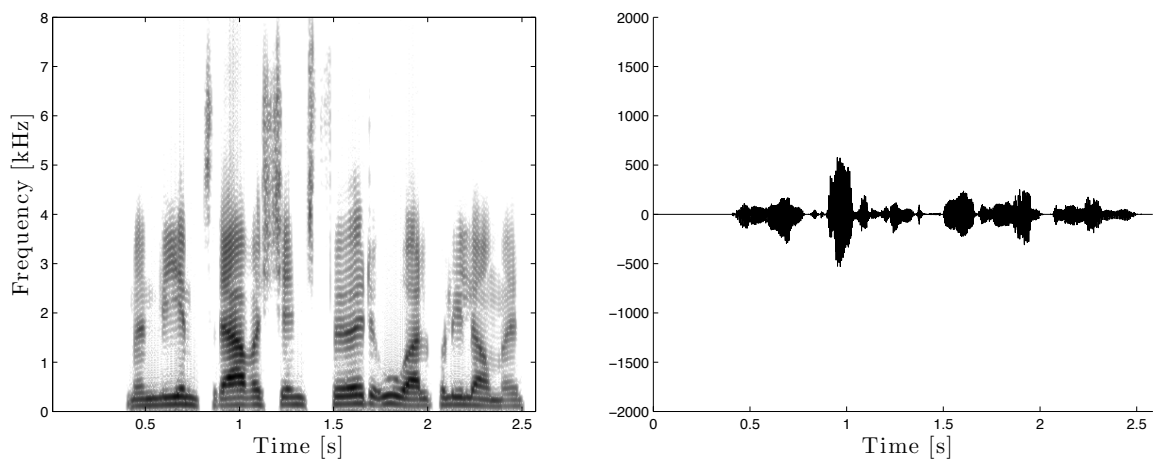


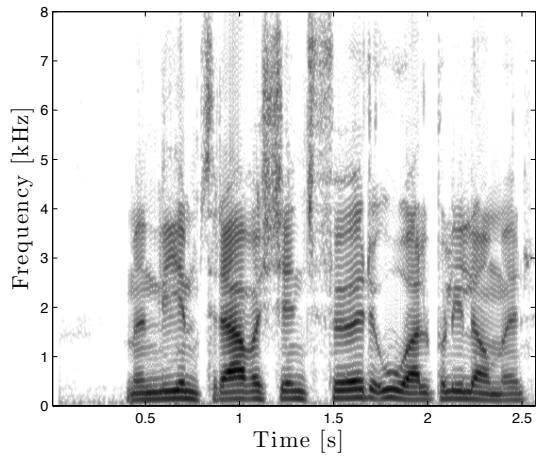
Figure 4.12: LR simulation according to a flat hearing loss 30 dB.

4.3 RESULTS OF RFS AND LR SIMULATION

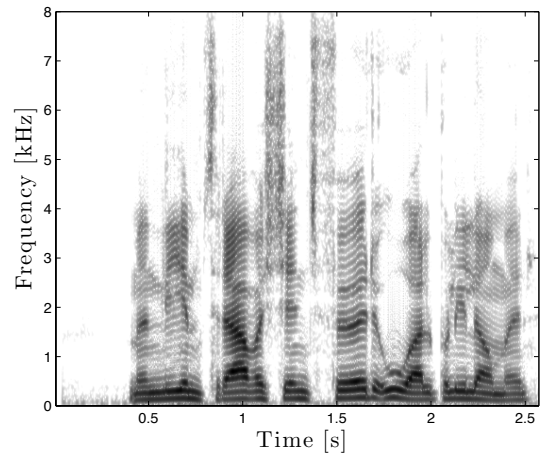
As RFS and LR are both caused by OHC damage, a person with cochlear hearing loss is likely to experience both effects. When simulating both RFS and LR, RFS may affect LR simulation. One of the ideas of simulating RFS by the method described in Section 3.7 was that the level estimation could yield false values. These erroneous values could cause the LR simulation scheme to set the samples differently. Perceptually, simulating both RFS and LR is expected to produce the effects of both of them. However, as LR sets the samples and effectively attenuates the signal severely, it is likely to be the dominating effect.

Figure 4.13 and 4.14 shows the spectrograms and waveforms of the outputs of both RFS and LR simulation. The simulations were according to a third degree and a flat loss of 30dB. These losses were selected because they produced the largest differences from the unprocessed signal.

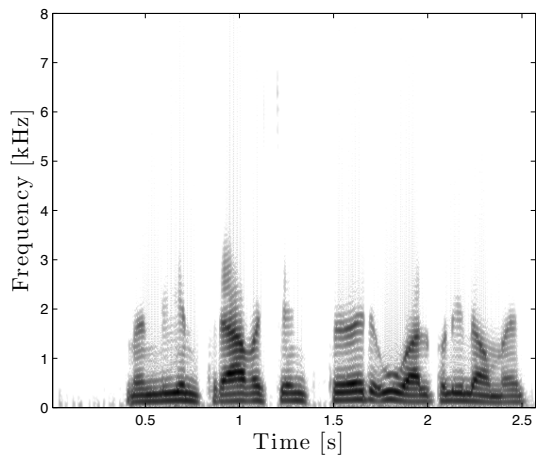
The simulation of a third degree loss sounds almost the same as the one with LR simulation alone. By comparing the spectrograms from Figure 4.13c and Figure 4.13e it can be seen that only a few more components are additionally attenuated. This is visible for the frequencies just higher than the strong components under 2 kHz. The flat loss produced clearer perceptual changes than the third degree loss. Some of the consonants, which were attenuated in LR simulations, had largely disappeared with both RFS and LR. The voiced sound following the vowel /æ/ has also disappeared, making the strong vowels appear even more abruptly than with LR simulation alone. This causes the signal to fluctuate even more, effectively exaggerating the effects of LR. Interestingly, the consonant /f/, which was inaudible when simulating LR, is audible with both RFS and LR simulation. This can be seen in the spectrogram as the frequency components following the stripe of energy at about 1.3 seconds. Combining RFS and LR simulation also appears to attenuate the signal more than LR simulation alone, this is especially visible in Figure 4.14.



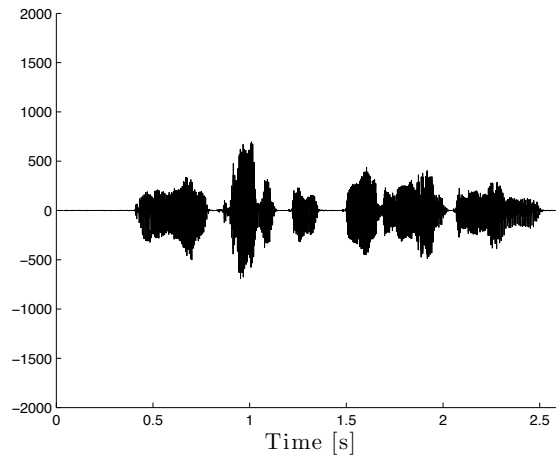
(a) Spectrogram of unprocessed signal



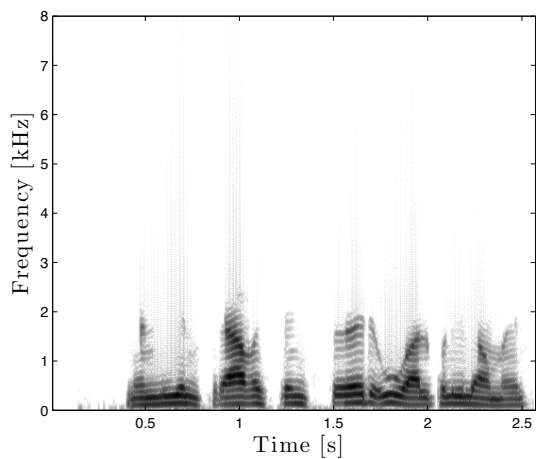
(b) Spectrogram of RFS simulation output



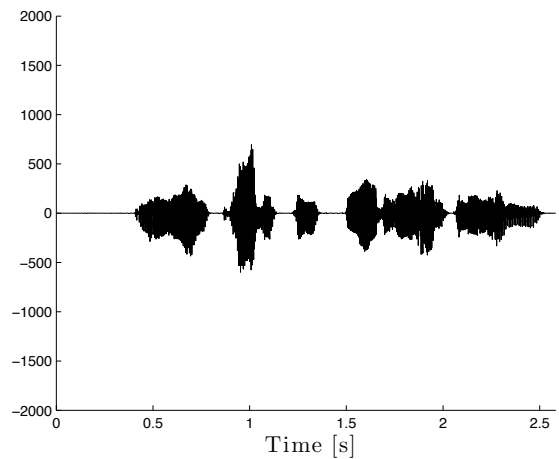
(c) Spectrogram of LR simulation output



(d) Waveform of LR simulation output

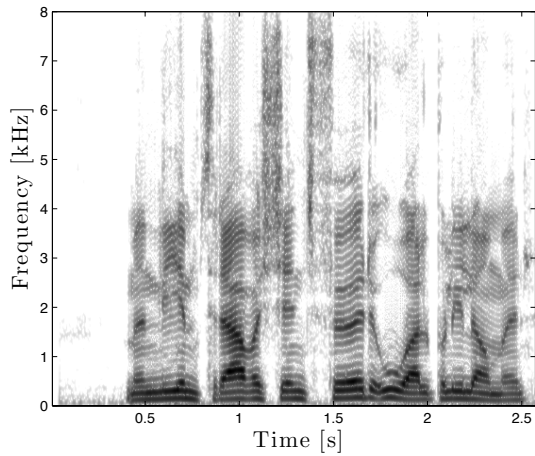


(e) Spectrogram of RFS and LR simulation output

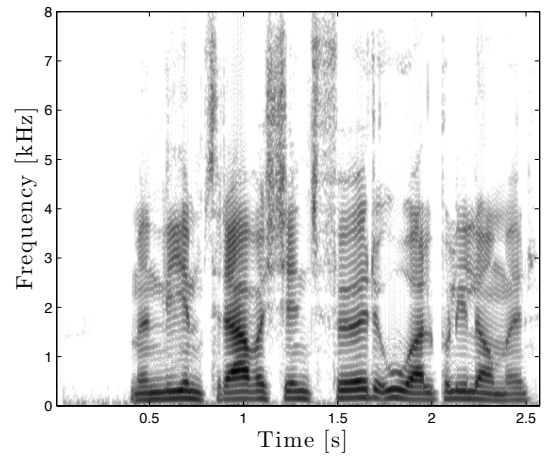


(f) Waveform of RFS and LR simulation output

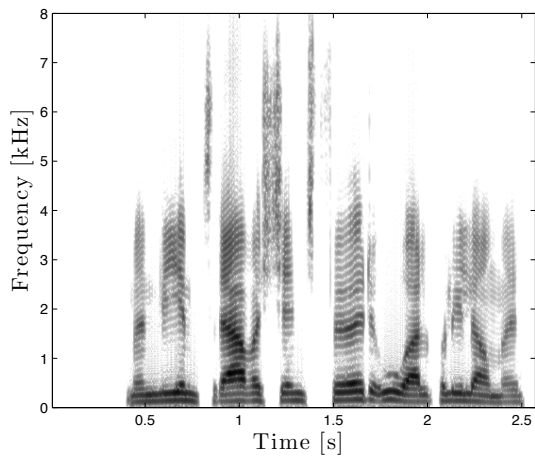
Figure 4.13: RFS and LR simulation according to a third degree hearing loss.



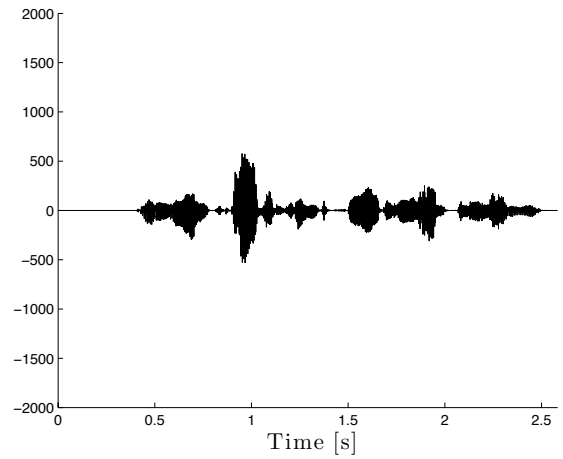
(a) Spectrogram of unprocessed signal



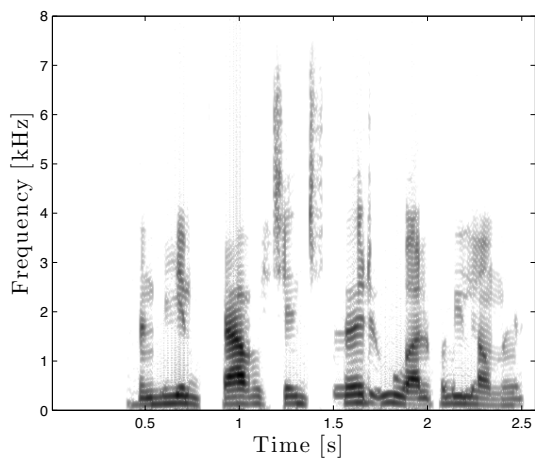
(b) Spectrogram of RFS simulation output



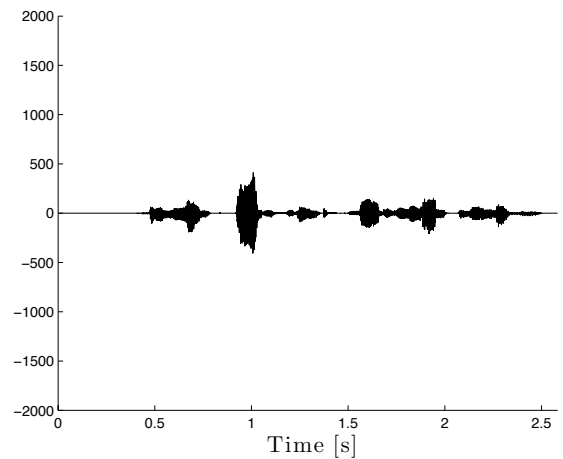
(c) Spectrogram of LR simulation output



(d) Waveform of LR simulation output



(e) Spectrogram of RFS and LR simulation output



(f) Waveform of RFS and LR simulation output

Figure 4.14: RFS and LR simulation according to a flat hearing loss of 30 dB.

5

DISCUSSION AND FUTURE WORK

This chapter will explain the results documented in Chapter 4, and briefly compare them to those of the hearing loss simulator created in *Simulation of Reduced Frequency Selectivity and Loudness Recruitment* [1]. How each individual effect is produced can be hard to explain specifically. Each sample depends on several parameters. It depends on previously estimated levels and hearing loss. In essence this means that the effects will depend on the input signal and the content of that signal. As well as explaining the results, suggestions to how future work can improve the simulator will be presented.

The method of reading sound files and playing audio caused a delay of 192 ms. A delay of this length is likely to be noticeable if the sound should be recorded, processed and then played back as a user is listening. An improvement to the audio playback could be made by using a different strategy for playing audio. Using the Audio Queue Services has a drawback in the sense that it needs three buffers to play a continuous stream of audio, even without processing. If this could be reduced to one buffer, the delay from a hearing loss is entered until the processed frame is played back would be reduced from 192 ms to 64 ms.

RFS simulation proved able to reduce the detail of a vowel's spectral envelope. A processing aspect that affects the spectral envelope is the bandwidths of the filters and the signal level. High levels cause high bandwidths. These bandwidths may cause one filter to pass content from another channel, which could result in spectral interference. The bandwidths depend on the the previously estimated level and the output of a higher filter. Because of this it is not necessarily true that the spectral envelope will degrade for every sound. It will depend on previous and higher frequency components of the sound. Another factor that may affect the envelope is the speaker. An example of this is if a person is speaking fast. In this case the tongue may not have reached its ideal position to create the desired vowel, changing the positions of the peaks in the

spectral envelope [18, p. 39].

RFS simulation caused minor audible effects on speech signals in silence. Previous RFS simulations have induced noise by various methods. These methods clearly affected the perception of the simulated hearing impairments. A comparison between the result of RFS simulation according to a flat loss from this simulator and the previous simulator created in 2011 [1] can be seen in Figure 5.1. The figure shows that the previous simulator caused massive spectral smearing, almost removing the variations in the spectrum. Smearing the spectrum made the signal sound "muffled" [1]. It is not certain that the perceptual effects of this or any other simulation is correct, compared to what hearing impaired people experience. But improvements could be made to create clearer perceptual results. An improvement could be to devise a scheme to reduce the amount of spectral detail. This could be done by averaging the spectrum, essentially the method used by Baer and Moore [24] and Moore, Glasberg and Simpson [27]. But, as it was mentioned in Section 3.7, these methods were quite costly, and may not be directly applicable to a real-time system. However, RFS caused some masking effects on the signal in silence, which may have been caused by increased two-tone suppression.

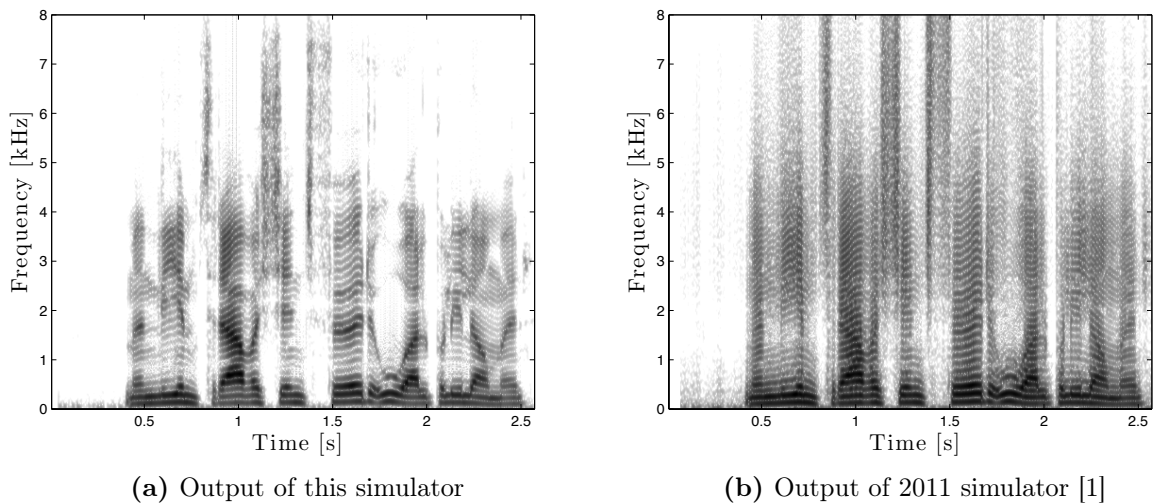


Figure 5.1: Comparison of this simulator's output of RFS simulation and the RFS output of the previous simulator [1]. The simulations are according to a flat loss of 57 dB.

When white noise was added to the signal, the characteristics of the noise appeared to change as the loss increased. What may have caused this was that the strong low-frequency components masked some of the noise, attenuating the components just higher than the speech. Additionally, the higher filters, mainly overlapping the noise, may have produced equal signal levels. This would cause the peaks of the filters to shift downwards, which could create an amplification area around 5-6 kHz.

As mentioned in Chapter 4, RFS simulations produced effects of LR. A plot of the RFS simulation output can be seen in Figure 5.2b. The figure shows that sounds are attenuated and amplified, specifically the amplification can be seen around 1 second.

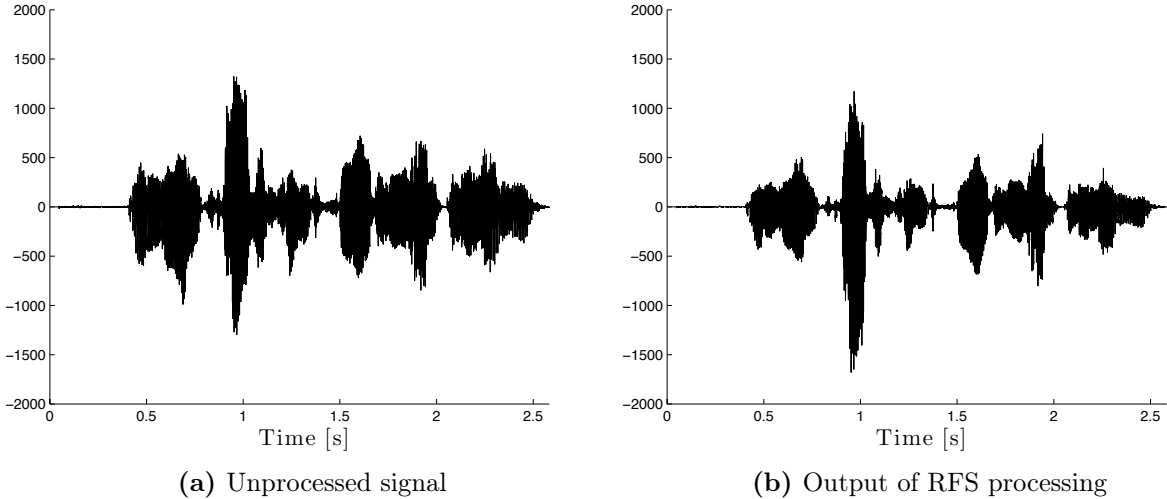


Figure 5.2: Figure 5.2a shows the time plot of the speech signal from Figure 4.1b. Output of RFS simulation according to a flat loss of 57 dB, Figure 5.2b.

The sound that is amplified is the vowel /æ/. As with all the results from the system, exactly what causes this effect is hard to pinpoint. However, the over-recruitment is likely to be caused by the overlapping filters. If a filter overlapping the vowel measures it to have a high level, its tail will broaden. Additionally, if the filter below also overlaps the vowel to an extent, and measures it to have a high level, its peak will shift upwards. Conversely will a higher filter shift downwards if it measures a low level. This frequency shifting, in addition to tail broadening, can lead to the filters causing a buildup over that sound. Buildup means that more than one filter passes a significant amount of the energy of the sound, causing amplification when the subbands are added.

Because of the method chosen to simulate LR, signals were overall attenuated. The attenuation could be avoided by choosing a different function to scale the samples. This could produce LR effects that are similar to what RFS produced, although more controllable. Moore and Glasberg’s [11] model, which was also used to determine the widening factor for the bandwidths, predicted a function equivalent to the one shown in Figure 2.5. The function is similar, but the slope, and not necessarily the threshold, is determined by OHC damage. Their function created different growth factors, and could also account for over-recruitment [11].

Figure 5.3 shows the output of this simulator compared to the one implemented in *Simulation of Reduced Frequency Selectivity and Loudness Recruitment* [1]. As with RFS, which of the two simulators that produce the most realistic result is unknown.

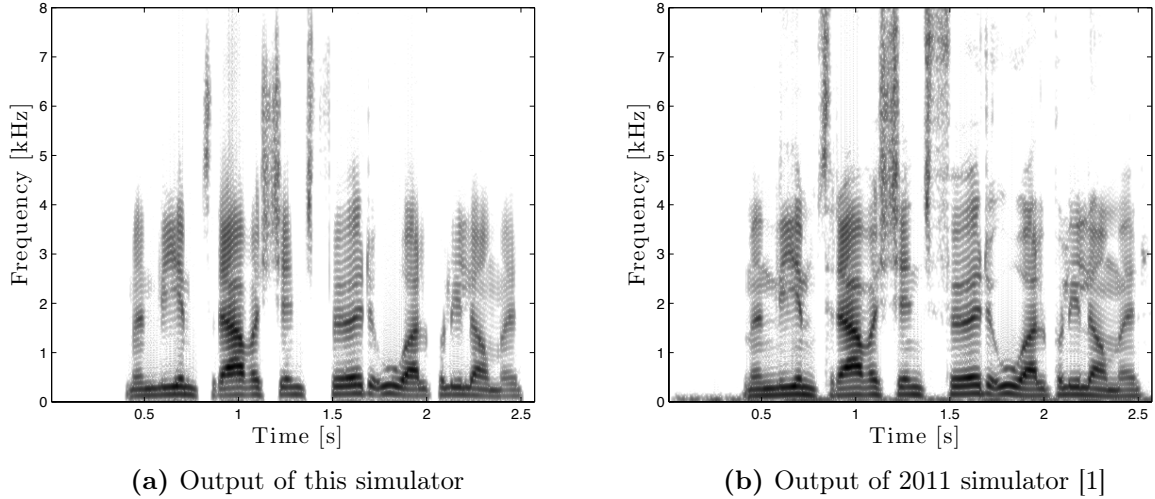


Figure 5.3: Comparison of this simulator’s output of LR simulation and the LR output of the previous simulator [1]. The simulations are according to a flat loss of 30 dB.

However, the expected effects of LR are more apparent in this simulator than in the previous one. This can especially be seen around 1.3 seconds where a consonant has almost disappeared in this simulator. In the 2011 simulator the same consonant is clearly audible, but has more of a pulse-like behavior [1].

As mentioned in Section 4.3, RFS and LR simulation together attenuates the signal more than LR simulation. This indicates that RFS causes the level estimation to produce erroneous levels, increasing the absolute threshold as LR simulation scales the samples.

Cochlear hearing loss will reduce or remove the compression caused by the cochlea [14]. Irino and Patterson [8] found that the compression caused by the DCGC depended on the half-life, λ , used during level estimation. By using that $\lambda = 0$ they found that the DCGC filter caused no compression. This could be employed in the simulator by reducing λ with hearing loss. A consequence of this would be that the implementation of the system would have to change, especially the gain tables.

6

CONCLUSIONS

The implemented hearing loss simulator is able to simulate RFS and LR in real-time. Before or during processing, a user may enter hearing loss values into a control file. These values affect the filter coefficients of the DCGC filters, and the signal is processed accordingly. The delay from hearing loss values are entered until the effects of them are audible is 192 ms.

Simulation of RFS caused minor audible differences, but removed spectral components depending on hearing loss. The higher the hearing loss the more spectral components were removed, indicating that the system is able to produce masking effects. This can be explained by increased two-tone suppression, something the DCGC filter bank was able to produce naturally. RFS is known to increase masking with increasing hearing loss. This means that increased masking is a desirable feature when simulating RFS. Another result of RFS simulation was that it proved able to degrade the spectral envelope of a vowel. This can in some cases lead to confusion between vowels.

LR simulation made some parts of a signal weaker, leaving other parts near normal. This had effects on consonants, removing or attenuating them. By making consonants inaudible or weaker, speech could be harder to understand as a substantial part of the information in speech is carried by the consonants.

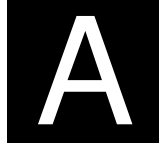
As OHC damage is the cause of both RFS and LR, people with this type of damage is likely to experience both effects. Hence would RFS and LR simulation at the same time be the most realistic simulation scenario. Simulating both RFS and LR exaggerated the effects of LR, and also increased the absolute threshold of the signal. The exaggeration of LR was seen as some consonants completely disappeared where they were only attenuated with LR simulation alone. The increased absolute threshold was seen as the entire signal was attenuated more with both RFS and LR simulation. Increased thresholds could mean that RFS caused the level estimation to produce erroneous

values, which was one of the motives of the RFS simulation scheme.

Overall, the simulator degrades the qualities of the signal that is passed through it. LR caused greater audible effects than RFS. Combining the two made consonants even weaker and vowels shorter. Many of the results are common features of both RFS and LR. In that sense the simulator is likely to produce output that mimics the effects experienced by hearing impaired people well, particularly LR. Additionally, few hearing loss simulators are made to process sound in real-time, and few employ the newly developed DCGC filters. This makes the implemented system stand out as a different way of simulating and presenting the effects of hearing loss.

LIST OF ABBREVIATIONS

AC	Asymmetric compensation filter
CGC	Compressive gammachirp filter
CSV	Comma-separated values
DCGC	Dynamic compressive gammachirp filter
ERB	Equivalent rectangular bandwidth
ERBS	ERB-scale
HP-AC	High-pass asymmetric compensation filter
HP-AC _L	Level estimation HP-AC filter
HP-AF	High-pass asymmetric function
IHC	Inner hair cells
IIR	Infinite impulse response
LP-AC	Low-pass asymmetric compensation filter
LP-AF	Low-pass asymmetric function
LR	Loudness recruitment
OHC	Outer hair cells
PGC	Passive gammachirp filter
RFS	Reduced frequency selectivity



DEGREES OF HEARING LOSS

The following tables are defined by The Norwegian Labour Inspection Authority [1]. All losses are given in dB ¹.

Table A.1: Degrees of hearing loss.

f_c	Degrees of hearing loss		
	1st	2nd	3rd
125Hz	0	5	12
180Hz	0	5	12
250Hz	0	5	12
350Hz	0	5	12
500Hz	1	5	13
750Hz	2	6	15
1000Hz	4	8	20
1500Hz	8	12	25
2000Hz	15	20	35
3000Hz	30	40	50
4000Hz	35	45	55
6000Hz	30	40	50
8000Hz	20	30	40

¹As these values are given for 13 channels, and the simulator has 25 channels, the losses must be interpolated.

Table A.2: Degrees of hearing loss with age

(a) Normal hearing loss with age.

f_c	Age					
	20	25	35	45	55	65
125Hz	0	0	0	0	0	0
180Hz	0	0	0	0	0	0
250Hz	0	0	0	0	0	0
350Hz	0	0	0	0	0	0
500Hz	0	0	0	0	0	0
750Hz	0	0	0	0	0	0
1000Hz	0	0	0	0	0	0
1500Hz	0	0	0	0	0	5
2000Hz	0	0	0	0	5	10
3000Hz	0	0	5	10	15	20
4000Hz	0	0	10	15	20	30
6000Hz	0	5	15	20	30	40
8000Hz	0	10	20	25	35	45

(b) First degree hearing loss with age.


f_c	Age					
	20	25	35	45	55	65
125Hz	0	0	0	0	0	0
180Hz	0	0	0	0	0	0
250Hz	0	0	0	0	0	0
350Hz	0	0	0	0	0	0
500Hz	1	1	1	1	1	1
750Hz	2	2	2	2	2	2
1000Hz	4	4	4	4	4	4
1500Hz	8	8	8	8	8	13
2000Hz	15	15	15	15	20	25
3000Hz	30	30	35	40	45	50
4000Hz	35	35	45	50	55	65
6000Hz	30	35	45	50	60	70
8000Hz	20	30	40	45	55	65

(c) Second degree hearing loss with age.

f_c	Age					
	20	25	35	45	55	65
125Hz	5	5	5	5	5	5
180Hz	5	5	5	5	5	5
250Hz	5	5	5	5	5	5
350Hz	5	5	5	5	5	5
500Hz	5	5	5	5	5	5
750Hz	6	6	6	6	6	6
1000Hz	8	-8	-8	-8	-8	8
1500Hz	12	12	12	12	12	17
2000Hz	20	20	20	20	25	30
3000Hz	40	40	45	50	55	60
4000Hz	45	45	55	60	65	75
6000Hz	40	45	55	60	70	80
8000Hz	30	40	50	55	65	75

(d) Third degree hearing loss with age.

f_c	Age					
	20	25	35	45	55	65
125Hz	12	12	12	12	12	12
f_{RAT}	12	-12	12	12	12	12
250Hz	12	12	12	12	12	12
350Hz	12	12	12	12	12	12
500Hz	13	13	13	13	13	13
750Hz	15	15	15	15	15	15
1000Hz	20	20	20	20	20	20
1500Hz	25	25	25	25	25	30
2000Hz	35	35	35	35	40	45
3000Hz	50	50	55	60	65	70
4000Hz	55	55	65	70	75	85
6000Hz	50	55	65	70	80	90
8000Hz	40	50	60	65	75	85

A large black square with a white letter 'B' inside, centered on the page.

USING THE SYSTEM

The finished system is used as a command line tool on Mac OS X computers. It is tested on a MacBook Pro with OS X 10.6 Snow Leopard. The computer has 4 GB RAM and an Intel Core 2 Duo processor running on 2.4 GHz. The system is only able to read and play wave files sampled at 32 kHz with 16 bits per sample.

For the simulations to be correct according to the signal level, the input signal must conform to certain requirements. The level estimation produces signal levels of 50 dB if the values of \bar{s}_1 and \bar{s}_2 are approximately 316. Signals that are found to produce such values for \bar{s}_1 and \bar{s}_2 often have sample values of 1000 - 2000. A way of testing if an input speech signal has the correct sample values is by running it through LR simulation. As speech often have levels of 50 dB SPL, LR simulation of a flat 50 dB loss should produce an output signal that is barely audible. Increasing the loss to 57 dB should render the output signal inaudible.

The system is run by opening the application `Terminal` and navigating to the folder where the executable `RT_HIS` is stored. When in that folder, the user must issue the following command:

```
./RT_HIS file.wav
```

After the command has been issued, the file given as input is played, and a CSV control file is created. The control file, named `control.csv` is stored in the same folder as `RT_HIS` and can be opened in the application `TextEdit`. `control.csv` allows the user to change the system parameters such as whether or not it should simulate LR or RFS or both. It also allows the user to change losses for the 25 channels the filter bank is composed of.

BIBLIOGRAPHY

- [1] Bertheussen, G. *Simulation of Reduced Frequency Selectivity and Loudness Recruitment*. 2011. Specialization project in Audio and Image Processing. Unpublished.
- [2] Apple Inc. *Audio Queue Services Programming Guide* Cupertino, CA. Apple Inc; 2007-10-31 [2010-07-09, 2012-04-11]. Available from: <http://developer.apple.com/library/ios/DOCUMENTATION/MusicAudio/Conceptual/AudioQueueProgrammingGuide/AudioQueueProgrammingGuide.pdf>
- [3] Strahl, S., Mertins, A. *Analysis and design of gammatone signal models*. J. Acoust. Soc. Am. 2009; 126(5): 2379-2389.
- [4] Ma, N., Green, P., Barker, J., Coy, A. *Exploiting correlogram structure for robust speech recognition with multiple speech sources*. Speech Communication. 2007; 49(12): 874-891.
- [5] Moore, B.C.J. *Cochlear Hearing Loss - Physiological, Psychological and Technical Issues*. 2nd Edition. West Sussex, England: John Wiley & Sons, Ltd; 2007.
- [6] Proakis, J.G., Manolakis, D.G. *Digital Signal Processing. Principles, Algorithms and Applications*. 4th edition. USA: Pearson Prentice Hall; 2007.
- [7] Irino, T., Patterson, R.D. *Dynamic, Compressive Gammachirp Auditory Filterbank for Perceptual Signal Processing*. 2006 IEEE International Conference on Acoustics Speed and Signal Processing Proceedings. 2006; 5: V133-V136.
- [8] Irino, T., Patterson, R.D. *A Dynamic Compressive Gammachirp Auditory Filterbank*. IEEE Transactions on Audio, Speech and Language Processing. 2006; 14(6): 2222-2232.
- [9] Chittka L, Brockmann, A. *Perception Space—The Final Frontier*. PLoS Biol. 2005; 3(4): e137.
- [10] Moore, B.C.J. *An Introduction to the Psychology of Hearing*. 5th Edition. London, UK: Elsevier Ltd; 2004.

- [11] Moore, B.C.J, Glasberg, B.R. *A revised model of loudness perception applied to cochlear hearing loss*. Hearing Research. 2004;188(1-2):70-88.
- [12] Kuruso, A., Miyase, S., Tomiyama, S., Takebe, T. *A Technique to Truncate IIR Filter Impulse Response and Its Application to Real-Time Implementation of Linear-Phase IIR Filters*. IEEE Transactions on Signal Processing. 2003; 51(5): 1284-1292.
- [13] Patterson, R.D., Unoki, M., Irino, T. *Extending the domain of center frequencies for the compressive gammachirp auditory filter*. J. Acoust. Soc. Am. 2003; 114(3): 1529-1542.
- [14] Oxenham, A.J., Bacon, S.P. *Cochlear Compression: Perceptual Measures and Implications for Normal and Impaired Hearing* Ear & Hearing. 2003; 24: 352-366.
- [15] Van Immerseel, L., Peeters, S. *Digital implementation of linear gammatone filters: Comparison of design methods*. Acoustics research letters online. 2003;4:59-64.
- [16] Irino, T., Patterson, R.D. *A compressive gammachirp auditory filter for both physiological and psychophysical data*. J. Acoust. Soc. Am. 2001; 109(5):2008:2022.
- [17] Unoki, M., Irino, T., Patterson, R.D. *Improvement of an IIR asymmetric compensation gammachirp filter*. Acoustical Science and Technology. 2001;22(6):426-430.
- [18] Huang, X., Acero, A., Hon, H-W. *Spoken Language Processing. A Guide to Theory, Algorithm, and System Development*. USA: Prentice-Hall, Inc.; 2001.
- [19] Allen, J.B. *Recruitment compensation as a hearing aid signal processing strategy*. ISCAS '98 - Proceedings of the 1998 IEEE International Symposium on Circuits and Systems. 1998;6 : 565 - 568.
- [20] Holdsworth, J., Nimmo-Smith, I., Patterson, R.D., Rice, P. *Annex C of the SVOS final report: Implementing a gammatone filter bank*. Annex C of APU report 2341. 1998.
- [21] Irino, T., Unoki, M. *A time-varying, analysis/synthesis auditory filterbank using the gammachirp*. Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing; May 12-15 1998; Seattle, WA (USA).
- [22] Loizon, P.C. *Mimicking the Human Ear*. IEEE Signal Processing Magazine. 1998;15(5): 101-130.
- [23] Gustafsson, F. *Determining the initial states in forward-backward filtering*. IEEE Transactions on Signal Processing. 1996; 46(4): 988 - 992.

- [24] Baer, T., Moore, B.C.J. *Effects of spectral smearing on the intelligibility of sentences in noise.* J. Acoust. Soc. Am. 1993; 94(3): 1229-1241.
- [25] Moore, B.C.J., Glasberg, B.R. *Simulation of the effects of loudness recruitment and threshold elevation on the intelligibility of speech in quiet and in a background of speech.* J. Acoust. Soc. Am. 1993; 94(4): 2050-2062.
- [26] Keurs, M.T., Festen, J.M., Plomp, R. *Effect of spectral envelope smearing on speech perception.* J. Acoust. Soc. Am. 1992; 91(5): 2872-2880.
- [27] Moore, B.C.J., Glasberg, B.R., Simpson, A. *Evaluation of a method of simulating reduced frequency selectivity.* J. Acoust. Soc. Am. 1992; 91(6): 3402-3423.
- [28] Powell, S.R., Chau, P.M. *A Technique for Realizing Linear Phase IIR Filters.* IEEE Transactions on Signal Processing. 1991; 39(11): 2425-2435.
- [29] Glasberg B.R., Moore, B.C.J. *Derivation of auditory filter shapes from notched-noise data.* *Hearing Research.* 1990; 47(1-2): 103-138.
- [30] Glasberg, B.R., Moore, B.C.J. *Auditory filter shapes in subjects with unilateral and bilateral cochlear impairments.* J. Acoust. Soc Am. 1986; 79(4): 1020-1033.
- [31] Patterson, R.D. *Auditory filter shapes derived with noise stimuli.* J. Acoust. Soc. Am. 1976; 59(3): 640-654.