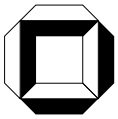
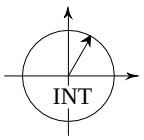


Universität Karlsruhe
Institut für Nachrichtentechnik
Prof. Dr.-Ing. Kristian Kroschel



The nature of planned acoustic trajectories

Diplomarbeit von

Marc Boucek

Hauptreferent	:	Prof. Dr.-Ing. Kristian Kroschel
Betreuer	:	Ph.D. Joseph S. Perkell
Ausgabe	:	05.02.2007
Abgabe	:	05.08.2007

Die Freiheit, es gibt sie wirklich: Sie ist hinter den Mauern, die man sich selbst errichtet hat.

– Unbekannter Autor

Author
Marc Boucek

Email: marc.boucek@gmail.com

Statement of Originality

The work presented in this thesis is, to the best of my knowledge and belief, original, except as acknowledged in the text. The material has not been submitted, either in whole or in part, for a degree at this or any other university.

Furthermore I agree that the thesis will be stored in a public library.

Marc BOUCEK Date

Abstract

The goal of this present study was to implement and test a real-time speech modification algorithm. Prior studies that have demonstrated adaptation to altered speech feedback perturbed steady state vowel formants. A linear prediction coding (LPC) algorithm has been developed to reliably track and shift multiple formants in steady state as well as changing formant trajectories. In an experiment, subjects were asked to pronounce utterances containing diphthongs with formant movement from [a] to [i], such as bike and tight. The transduced speech signal was processed by this algorithm and fed back to the subject with a delay of 10ms. The formants were shifted perpendicular to the straight-line trajectory connecting the start and end points of the transition. The shift was maximal at the transition mid-point and zero at the beginning and end of the transition. Some subjects compensated to the perturbation by altering their produced speech formants in a direction opposite to the perturbation.

Prof. Dr.-Ing. Kristian KROSCHEL Date
INT, Universität Karlsruhe (TH), Germany

Ph.D. Joseph S. PERKELL Date
RLE, Massachusetts Institute of Technology, MA, USA

Prof. Ph.D. Pascal PERRIER Date
ICP, Institut National Polytechnique de Grenoble, France

Acknowledgements

First of all I would like to thank my supervisor Ph.D. Joseph Perkell. Thanks, Joe, for offering me this fascinating project, and for introducing me to the Speech Communication Group.

I would like to express my highest gratitude to my advisor Ph.D. Strajit Ghosh. Satra, thank you so much for all your help and kindness, and all the good times we had. I already miss the 4 o'clock discussions...

I would like to thank Caroline Niziolek for proof reading my thesis. Thanks a lot, Carrie, in particular for you last minute "express" corrections.

I would like to thank Prof. K. Kroschel for supervising my thesis in Germany and for giving me the opportunity to participate in the double degree exchange program with Grenoble. These 2 years in France have broaden my horizon, not only from an academical point of view, but also and in particular from a personal perspective.

I would like to thank Prof. Pascal Perrier for supervising my thesis in France, and for recommanding me to the Speech Communication Group. Thank you for attracting my interest in speech processing... I love it!

Finally, I want to thank my family. Thank you for supporting me all these years and allowing me to achieve my goals. All of this would never have been possible without you : THANK YOU SO MUCH !

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent advances in digital signal processing (DSP) have widely contributed to the development of new technologies. In many fields, DSP has even become an omnipresent and almost indispensable tool. In speech research, signal processing has enabled the realization of experiments and thereby contributed to a better understanding of the human speech production system. This Master's thesis arose precisely from the aim to realize such a new experiment, one which may help to uncover the mystery around the **nature of planned acoustic trajectories**.

1.1 Project description

The project was defined within a series of speech sensorimotor adaptation experiments at the Speech Communication Group at MIT. This particular project was initiated by my supervisor, Dr. Joseph S. Perkell:

> *"Building on Houde's [4, 5] findings, we can test how speech movement trajectories are planned in acoustic space by adapting a paradigm for the study of reaching movements by Wolpert et al. [12] which leaves movement end points unchanged but perturbs perceived trajectory midpoints. Specifically, we wish to test the hypothesis that speakers control the entire acoustic trajectory, including the portion of this trajectory that occurs between the acoustic goal regions of consecutive phonemes (i.e., the portion of the trajectory starting from the end of one acoustic goal and ending at the start of the next acoustic goal)."*
>
> Joseph Perkell, personal communication

To resolve this issue we have designed a speech sensorimotor apparatus that enables us to perturb the acoustic trajectories subjects hear from their own speech in realtime. The changes in production observed as a result of this perturbation will provide insight into the planning mechanisms for speech.

1.1.1 Sensorimotor adaptation

Generally speaking, sensorimotor adaptation (SA) is the phenomenon that occurs when human motor actions adapt to altered sensory feedback. SA has been reported in many fields of motor control. A famous adaptation experiment is the so-called prism experiment, in which subjects were asked to reach a visual target while wearing displacing prisms that perturbed visual feedback. After a few movements, subjects were able to reach the targets and thus learned how to compensate for this perturbation. After the prism was removed it took a while before subjects adjusted back to the natural environment.

1.2 Speech sensorimotor adaptation

A speech sensorimotor adaptation experiment is a particular form of SA where the altered motor control is the produced speech. The sensory feedback can be acoustic, but this is not necessarily required; the feedback can be visual, for instance. Several speech sensorimotor adaptation experiments have been made over the past years to investigate the particular role of auditory feedback in speech production.

1.2.1 Formant shifting SA experiments

In recent years, special attention has been given to a particular class of speech SA experiments where subjects' formants have been shifted in frequency. Formants (or formant frequencies) are resonance frequencies that arise due to a particular shape of the vocal tract when producing voiced sounds. By changing the shape of our vocal tract we influence these formants, allowing us to produce and to discriminate several vowels like [a], [i], [o], and [u]. Since most vocalic information is comprised in the first two formants (the lowest in frequency), vowels are typically represented in the so-called acoustic space, spanned by $f_1$and f_2 axes, where f_1 / f_2 is the frequency of the first / second formant.

A formant-shifting SA experiment consists of perturbing one or more formant frequencies: for instance, shifting the formants of the vowel [a] to the frequencies of the vowel [i]. The modified speech signal is fed back in real time to subjects' ears using encapsulated headphones. The subjects thus perceive the modified speech signal as if they were producing it. Similar to the prism experiment, subjects are able to adapt to this perturbation in order to reach the acoustic target they planned. The first experiment of this kind was conducted by Houde and Jordan [4] in 1998.

1.2.1.1 Houde & Jordan's SA experiment

> *"In two two-hour experiments subjects whispered a variety of words. For those words containing the vowel [e], subjects heard auditory feedback of their whispering. A DSP-based vocoder processed the subject's auditory feedback in real-time, allowing the formants of subject's auditory speech feedback to be shifted. In the adaptation experiment, formants were shifted along one edge of the vowel triangle. For half of the subjects, formants were shifted so subjects heard [a] when they produced [e]; for the other half, the shift made subjects hear [i] when they produced [e]. During the adaptation experiment, subjects altered their production of [e] to compensate for the altered feedback. Subjects exhibited a range of adaptations in*

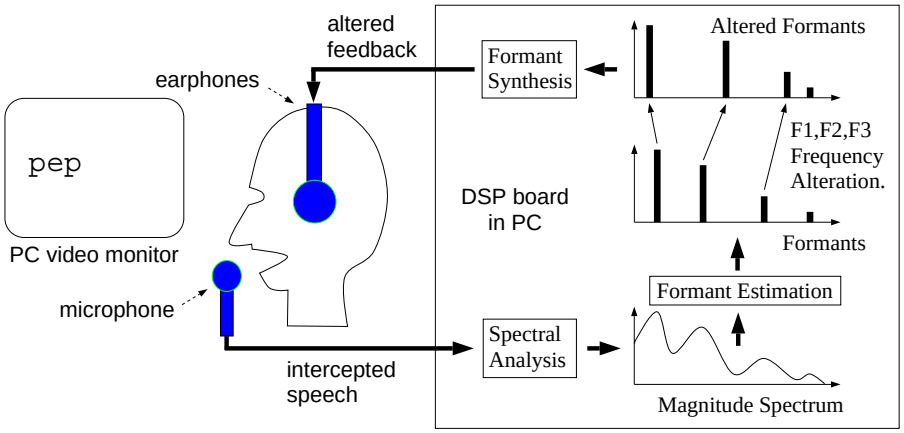


Figure 1.1: Houde & Jordan : Experiment setup (taken from [5])

response to the altered feedback, with some subjects adapting almost completely, and other subjects showing very little or no adaptation. "

Houde & Jordan 1998 1.1

The sensorimotor adaptation apparatus developed by Houde & Jordan is schematically represented in Figure 1.1.

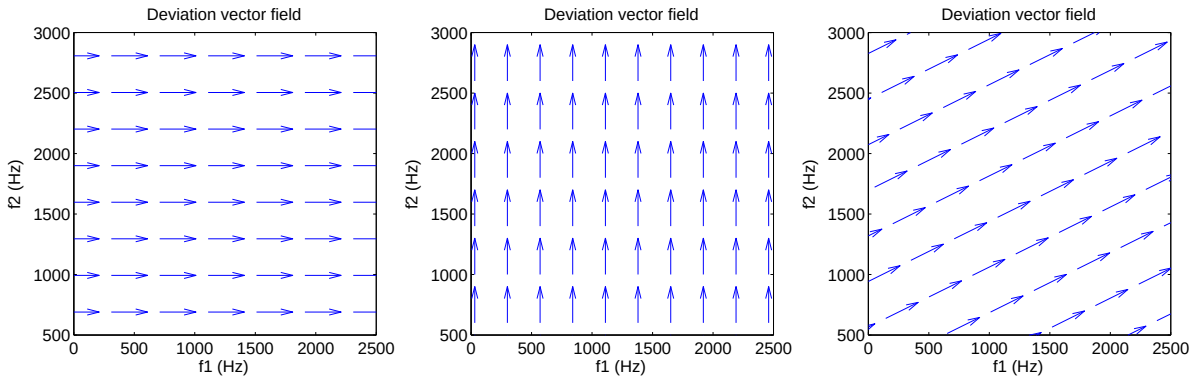
1.2.1.2 Further SA experiments

Based on the original experiment by Houde and Jordan, Villacorta et al. did a similar experiment in 2004-05, followed by Purcell and Munhall[8] in 2006. In Villacorta and Purcell's experiments only f_1 was shifted, whereas in Houde's experiment it was $f_1$and f_2. Furthermore, subjects produced voiced sounds, as opposed to Houde's experiment which used whispered speech. Nevertheless, these three experiments were very similar in the sense that formant frequencies of steady-state vowels were perturbed. A steady-state vowel (e.g. [a]) is represented by a single dot in the acoustic space. Shifting the formants of a steady-state vowel means applying a constant deviation to these formants. This can be interpreted mathematically as a mono directional invariant [1]and homogeneous deviation vector field applied to the acoustic space. Figure 1.2 shows such a deviation vector field, where only f_1, only f_2 or $f_1$and f_2 are deviated from their original location in acoustic space.

1.2.2 The experiment

We have adapted the original experiment formulated by Houde & Jordan in order to extend the studies to non steady-state vowels, i.e. to transitions between two vowels. As Houde & Jordan's experiment was the audio analog of the prism experiment, our experiment can be seen as the audio analog of Wolpert et al.'s [12] experiment.

[1]time invariant, in the sense that the field does not change during a single utterance



(a) Only f_1 shift (b) Only f_2 shift (c) $f_1$and f_2 shift

Figure 1.2: Vector field representation of steady state formant shifting SA experiments

1.2.2.1 Wolpert et al.'s SA experiment

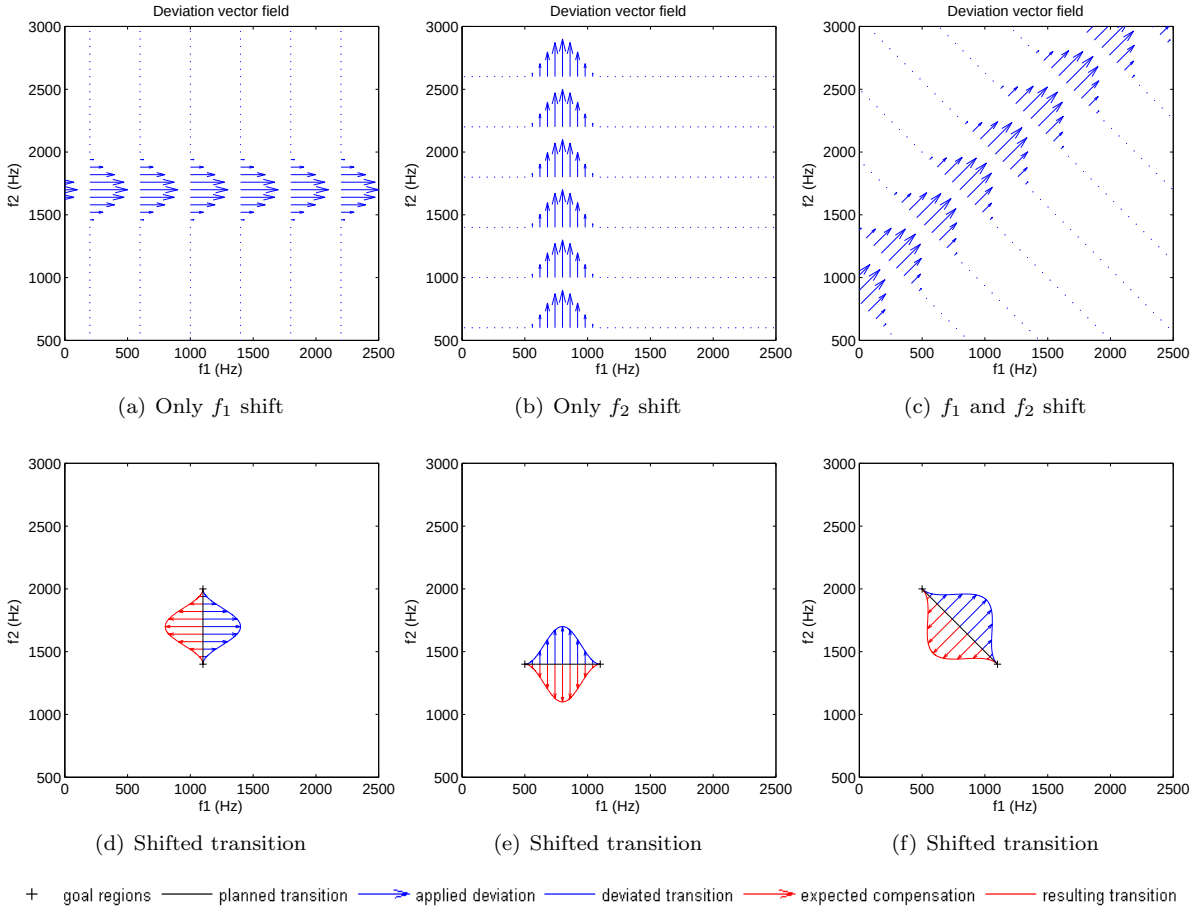
> *"In Wolpert et al. [12] , a subject moved his/her hand back and forth between two targets while viewing only a dot on a screen that tracked his hand position. Initially, the dot tracked hand position accurately. However, as the experiment progressed, a perturbation was introduced that made the dot's path appear to bow to one side. The perturbation was zero at both ends and reached a maximum at the midpoint of the movement. This caused most subjects to compensate by bowing their actual hand trajectories in the opposite direction in order to straighten the resulting trajectory image. This result provided evidence that reaching movements in the tracking task are planned in terms of (x, y) hand coordinates."*
>
> Joseph Perkell, personal communication

We have developed a speech analog of Wolpert et al.'s experiment. Subjects [a] [i] formant trajectories (contained in words like "bike" or "kite") are bowed into one direction: the altered speech is fed back via the developed speech sensorimotor apparatus in real time.

Figure 1.3 principally shows how trajectories are bowed within the acoustic space. The upper panel describe a formant deviation vector field, which will bow subject's formant trajectory as represented below.

1.2.3 Pilot study

We performed a pilot study of the described experiment, in which 7 female and 4 male subjects were involved. Subjects were repeating words like "bike" or "kite" all containing an [a] to [i] transition. On two distinct sessions, subjects' [a] [i] trajectories were shifted either down or up. The results of this pilot study are presented in Chapter 5.



(a) Only f_1 shift (b) Only f_2 shift (c) f_1 and f_2 shift

(d) Shifted transition (e) Shifted transition (f) Shifted transition

Figure 1.3: Deviation vector field

Chapter 2

Speech production

In this Chapter we briefly introduce basic speech production mechanisms, which will be necessary for further understanding of this thesis. Those who are already familiar with speech should directly continue with Chapter 3 where the actual formant shifting algorithm will be introduced.

2.1 The human speech production system

Figure 2.1 on page 6 shows a schematized view of the human speech production system. For the purpose of this project it is not necessary to describe in detail the role played by every single part. Later on we will use the term glottis, also known popularly as vocal cords.

As we will see, speech production can be divided in two main categories: the production of voiced and unvoiced sounds. The fundamental difference between these two types of speech sounds comes from the way they are produced. The vibrations of the vocal cords produce voiced sounds. The unvoiced sounds are created by the constriction of the vocal tract.

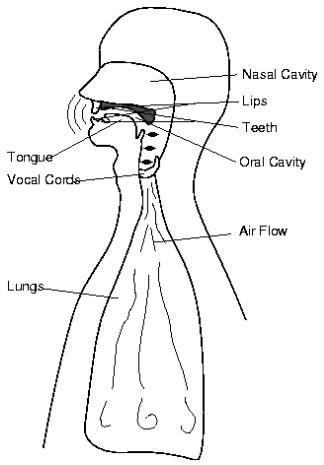


Figure 2.1: The human speech production system (weblink)

In this dissertation our discussion will mostly be focused on voiced speech. The models we will present do not apply, or only partly apply, for unvoiced speech.

2.2 Source-filter model

A good approach for modeling speech is the so-called source-filter model[1] described in [1] and [10], where the production of speech is considered to be the consequence of the generation of one or more sources of sound, which are filtered by the vocal tract. In most cases, the source and filter can be seen as two independent entities.

2.2.1 Vocalic sounds

For vowel production, the cyclic opening and closing of the glottis creates a sequence of pressure pulses, which results in a nearly periodic sound wave: the glottal source. The periodic pulses from this source excite resonant modes of the vocal tract: the filter. The resonances occur at certain frequencies called formant frequencies, or simply formants, which are related to the shape of the vocal tract. Modifying the shape of the vocal tract, e.g. by moving the tongue close to (or far from) the roof of the mouth, will imply a change of each formant's frequency and intensity and hence change the spectral shape of the produced sound.

For voiced sounds, the source can be modeled as a periodic function, with a period T_0 ranging from about $3-7\,ms$ for females and $5-12\,ms$ for males. $F_0 = 1/T_0$ is called the fundamental frequency. Since the source is periodic in the time domain, the resulting spectrum in the frequency domain only contains spectral components at multiples of the fundamental frequency F_0 . A typical source spectrum $|S(f)|$ is depicted qualitatively in the upper left panel of Figure 2.2. The source's location is at the beginning of the vocal tract, i.e. just after the glottis, and schematically represented above the source's spectrum.

2.2.2 Fricatives

Fricatives are a special class of consonants, produced by the forcing of breath through a narrow channel made by placing two articulators close together. For the fricative [f], for instance, these articulators are the lower lip against the upper teeth. As opposed to vowels, the vocal cords stay open and hence they do not vibrate. The airflow forced through the constriction creates a noise called friction which is created in the vicinity of the constriction as represented in the upper right image of Figure 2.2.

The right panel of Figure 2.2 shows the magnitude response $|S(f)|$ of the source signal. The spectral shape of the source is very similar to a white noise, except that the magnitude decreases towards higher frequencies. Since the source is not periodic, its spectrum is continuous, in contrast with vowels.

When producing a fricative, the front cavity (i.e. the resonator) is very small and thus the resonance frequency is rather high. The filter's magnitude response $|T(f)|$for the fricative [s] is represented in the

[1] A good visualization of the source filter model can be found at `http://linguistics.online.uni-marburg.de/free/generalmodules/animations/phonetics/source_filter.html`. Additional information is also provided by [2] where the author describes the properties of the vocal tract in a more detailed way.

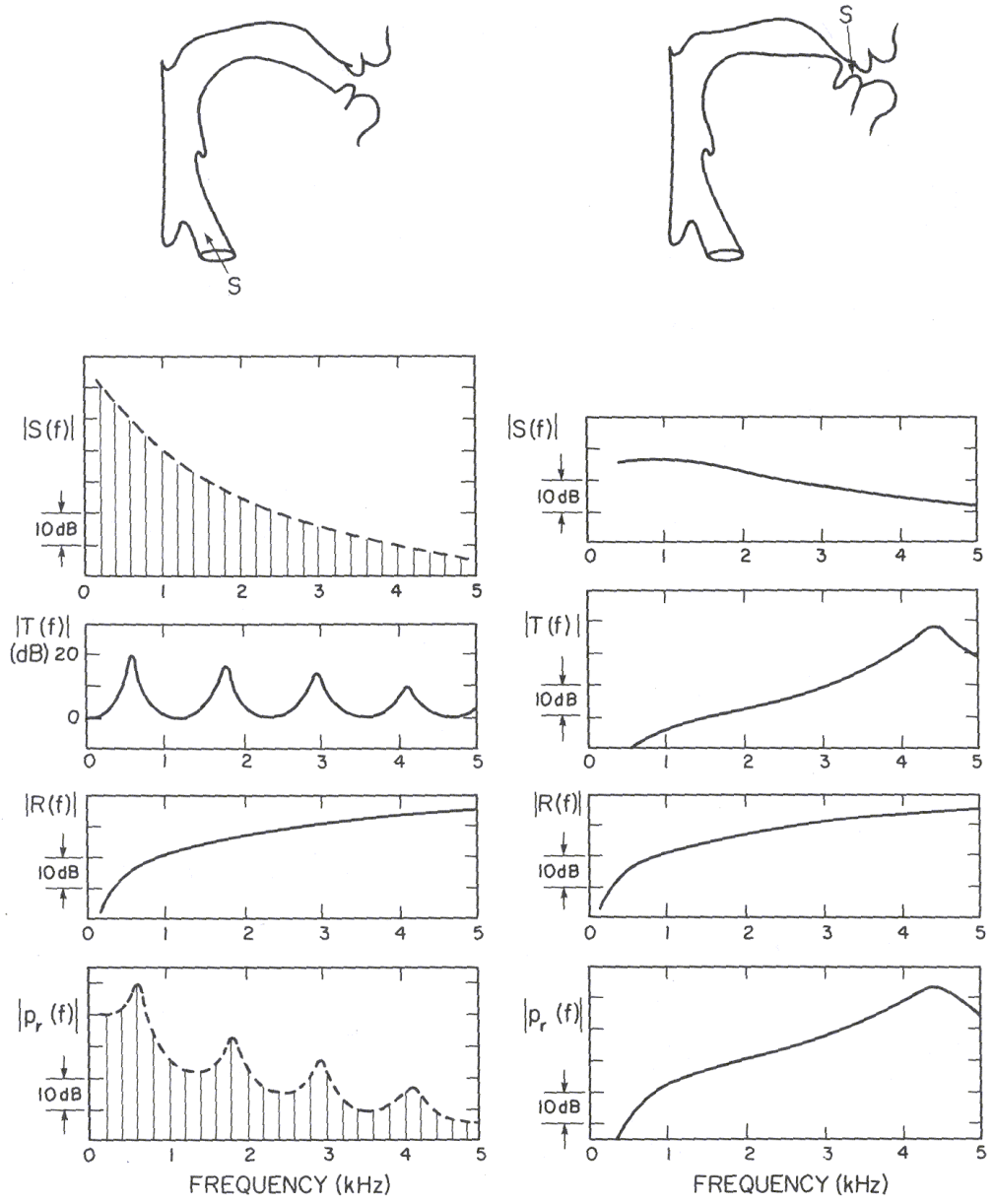


Figure 2.2: The source-filter model of speech production (from [10])

second right panel from top of Figure 2.2.

2.2.3 Overall spectrum

Since the human ear picks up changes in sound pressure, and microphones respond to pressure variations, one important quantity is the sound pressure, which is denoted $p_r(t)$ in [10]. The Fourier transform of the sound pressure $p_r(t)$ at a distance r from the lips can be shown to be

$$p_r(f) = S(f)T(f)R(f) \tag{2.1}$$

$R(f)$ is called the radiation characteristic and represents the losses occurring at the lips. The radiation characteristic rises with frequency with a slope of $6\,dB$ per octave. Its characteristic magnitude response $|R(f)|$ is represented in the second panel from the bottom of Figure 2.2.

The resulting sound pressure magnitude response for vocalic sounds and for the fricative [s] are represented in the lowest panels of Figure 2.2.

2.3 Summary

As we just learned, speech production can be modeled as a concatenation of a source function, a transfer function, and a radiation characteristic. For vowels, the source is a periodic function, which excites resonant modes of the vocal tract, called formants. To produce a particular vowel, we force air through our vocal cords to make them vibrate and then shape our vocal tract according to the vowel we wish to produce. While one and the same vowel can sometimes be reached by different shapes of the vocal tract, e.g. [a], the formants are almost unambiguously related to the corresponding vowel. Thus it is possible to identify a vowel by extracting its spectral envelope, i.e. the filter's transfer function, and locating its formants. Since we are interested in manipulating the formants, i.e. shifting them in the frequency domain, only the filter's transfer function is relevant to us. One should always bear in mind that, when speaking about "shifting formants", implicitly, we mean we are modifying the filter's transfer function $T(f)$.

Chapter 3

Algorithm : an overview

This chapter briefly introduces the formant shifting algorithm and its software implementation.

3.1 A simple audio plug-in

The algorithm can be seen as a simple audio plug-in, principally an audio process "plugged-in between" an audio stream, as illustrated in Figure 3.1.

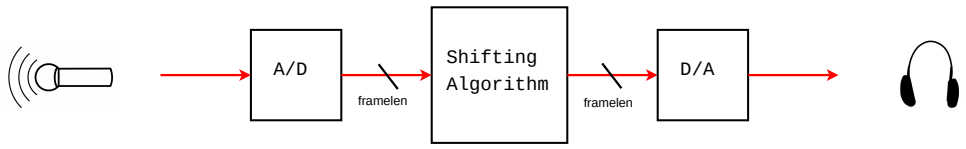


Figure 3.1: Audio Stream

Figure 3.3 schematically shows the structure and signal flow of the algorithm. For every input frame the algorithm delivers an output frame of the same size, which depends on the properties of the soundboard.

3.2 Software implementation

We recall that the shifting algorithm is a tool that allows us to realize the speech sensorimotor experiment described in Section 1.1. Ideally, we would like the headphone feedback to be simultaneous with a subject's actual speech production. Unfortunately, we all know that this is not possible because of hardware and software latencies. However, reducing this latency was one of the most important requirements to comply with, and became a real challenge. Although the original project specifications included the implementation of the algorithm on a DSP board, we finally decided that it was more advantageous to utilize a conventional personal computer instead. We could satisfy the strong real-time requirements by utilizing an additional soundboard, supporting the so-called ASIO[1] audio standard, which allows very

[1] ASIO (Audio Stream Input/Output) is a protocol for low-latency digital audio specified by steinberg `http://steinberg.de/24_0.html`.

low latency audio recording and playback.

3.2.1 Software structure

Basically, the whole software implementation comprises three major entities. First of all, we have a Matlab script that controls the whole experiment. Secondly, we have the formant shifting algorithm and the ASIO class, which are both written in C++. Last but not least, we have a .mex "interface" which connects the C++ functions with the Matlab functions. Figure 3.2 shows how these entities are related to each other.

The whole structure is very elegant, since only functions that needed to be implemented in real time are written in C++ code. All the other functions are written in Matlab, which allows us to use all the provided libraries and thus to develop functions more rapidly. Furthermore, Matlab is much more convenient when it comes to visualizing and analyzing data.

3.2.1.1 Matlab script

The Matlab script controls the overall experiment. While guiding subjects through a GUI, it takes care of all the associated actions, such as initializing the sound card, starting and stopping the recording, defining the amount of shift, etc. In fact more than 30 parameters can be set within the Matlab environment. The whole experiment is set up so that no external help is needed. Subjects see words appearing on a screen, while Matlab opens a recording time slot. After each recording, the shifting algorithm returns the recorded data, which comprises RMS values, formant tracks, recorded input and output audio signal, etc. Based on this data, Matlab updates the GUI displays which indicate the loudness and the length of the utterance. This is to make sure that subjects produce words in an identical way from trial to trial.

3.2.1.2 Mex interface

A .mex file is a special format used by Matlab to embed C++ functions. Our .mex interface remaps and transfers commands and variables to the specific C++ functions. We call it an interface, because that is basically what it is, even though it is not a "hardware" interface.

3.2.1.3 Formant shifting algorithm

The algorithm itself is embedded as a callback function respecting the ASIO specifications. The soundboard writes every new frame into a buffer and then calls the shifting algorithm, passing a pointer to the address of that buffer. The algorithm processes the frame contained in the buffer and returns a processed frame into that same buffer, which is then used by the soundboard for playback.

3.2.2 Single trial action sequence

A specific Matlab sequence, called a single trial, is called back for each word, and manages the calls to the different functions and classes. A typical action sequence is as follows:

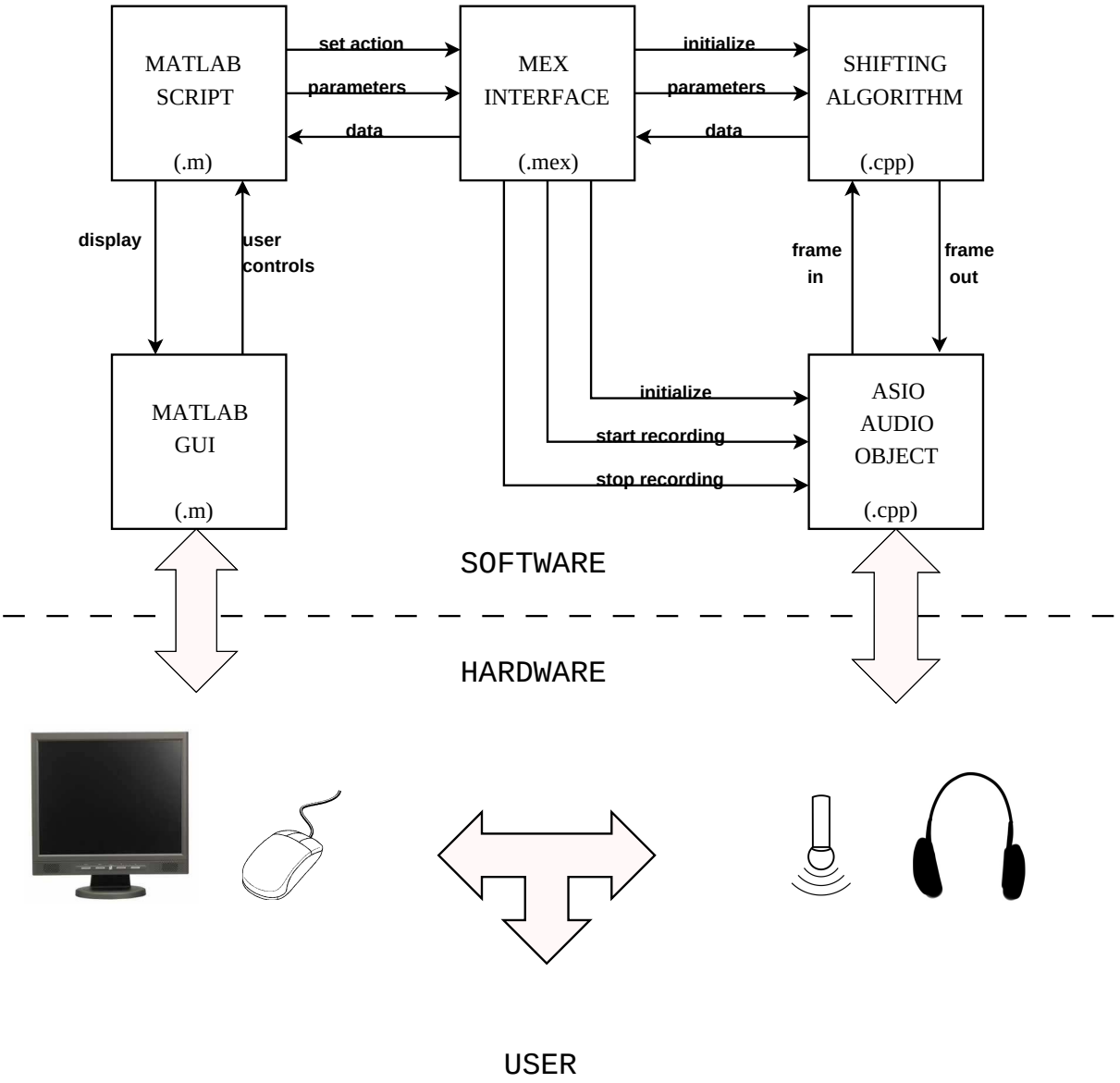


Figure 3.2: Software structure

1. Initialize the audio object (setting different parameters such as sample rate, buffer size, mono/stereo...)
2. Initialize the algorithm (pre-allocating recording space, setting up buffers, ...)
3. Set parameters (experiment specific parameters, such as amount of perturbation, voicing thresholds, forgetting factors, number of formants...)
4. Reset all internal memory allocations (i.e. filter states, recording buffer...)
5. Setup GUI display
6. Wait for user to press play (if currently in pause mode)
7. Display current word on the screen[2]
8. Start recording
9. Wait the amount of time specified by the slider
10. Stop recording
11. Get recorded data.
12. Display RMS and transition speed
13. Analyze data: if data does not satisfy certain criteria, go back to 3)
14. Save recorded data.
15. Load next word and continue with 3).

3.3 Shifting algorithm : Block diagram

Figure 3.3 shows the signal flow of the formant shifting algorithm. It is useful to be familiar with this diagram, since we will further describe every single block in the next chapter.

[2]The list is : “bike”,”kite”,”site”,”light”,”tight”,”fight”,”bite”

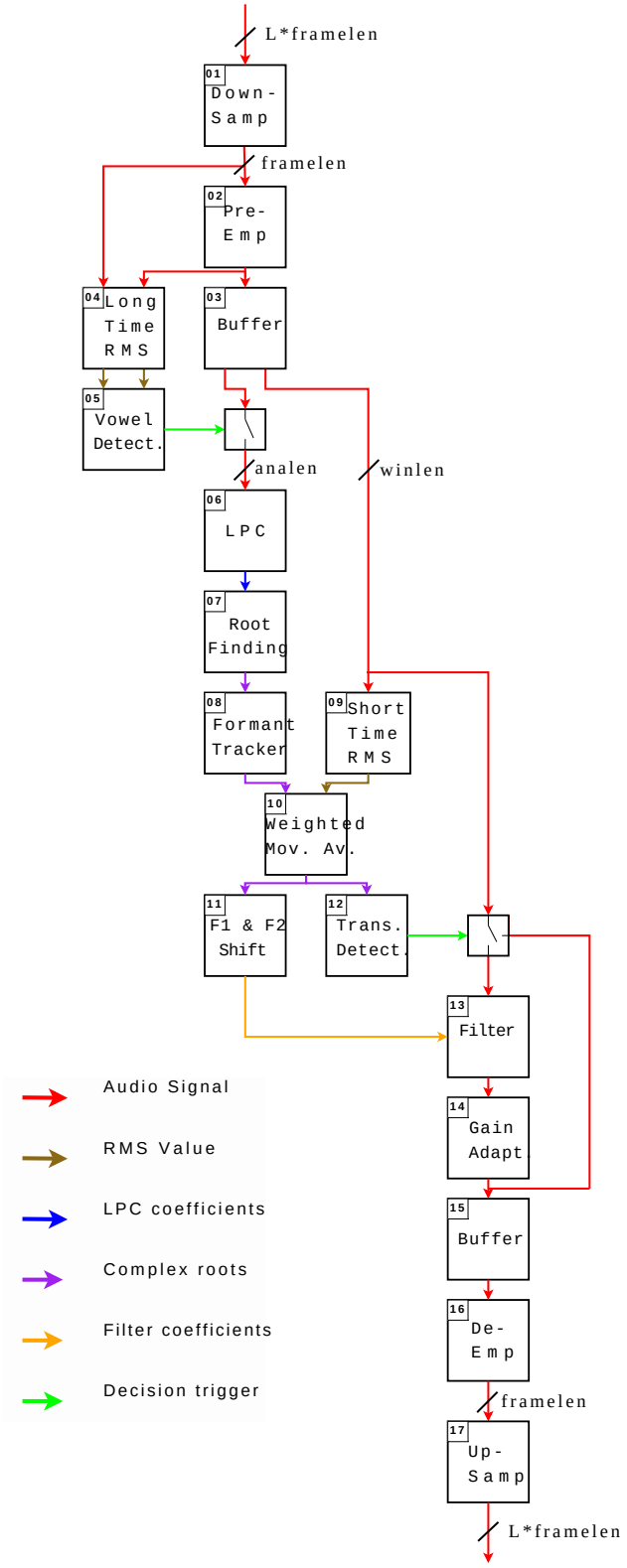


Figure 3.3: Signal flow

Block description:

1. Downsamples each incoming frame by a factor of 4.
2. Pre-emphasizes the signal in order to amplify higher frequencies for a better LPC estimation.
3. Stores the input signal to provide enough samples for the LPC analysis.
4. Calculates the long-time RMS of the original and the pre-emphasized input signal.
5. The vowel detection uses the long time RMS of the original and pre-emphasized signal to detect vowels. If no vowel is detected, the signal will neither be analyzed nor be filtered, but will directly be sent to the pre-output buffer (block 14).
6. LPC analysis, using the autocorrelation method. Provides coefficients of the estimated vocal tract filter.
7. This block computes the complex roots of the LPC coefficients and sorts them according to their angle.
8. The formant tracking algorithm tracks the formants in time, using dynamic programming.
9. Computes the short-time RMS.
10. Formant smoothing : Computes a weighted moving average of the formants over approximately one pitch period, using the short-time RMS as weighting factor.
11. Calculates the deviated trajectory based on a deviation vector field.
12. Transition detection: Analyzes the formant derivatives in time, and enables the filtering when a transition is detected.
13. Performs the formant shift by filtering the signal. The filter is a concatenation of two biquad IIR filters.
14. The gain adaptation applies a gain factor to the filtered signal. This gain is based on the properties of the vocal tract.
15. Pre-output buffer : stores one or more frames before being deemphasized.
16. De-emphasizes the frame.
17. Upsamples the frame by a factor of 4 and writes it to the sound card output buffer.

3.4 Algorithm : Properties and settings

Before moving on to the next chapter, which provides detailed information of every block, it is necessary to describe some of the settings and properties of the algorithm:

F_s	Samplerate of the soundboard.
DMA	Size of the soundboard buffer. In order to reduce latency this buffer size is set to the minimal supported size.
M	Downsampling factor: each incoming frame provided by the soundboard is downsampled before being processed. After processing, the frame is upsampled and sent back to the sound card.
$F_{s_{\downarrow M}}$	Internal samplerate, i.e. after downsampling. In fact, all the processing is done at this downsampled rate, so that one can say that this is the algorithm's internal samplerate.
$framelen$	Size of the internal IO buffer: $frameLen = DMA/M$
$winlen$	Each internal frame (of $framelen$ samples) can be divided in N_w smaller frames, each of the size of $winlen$.
$buflen$	Number of samples stored in the input buffer. These samples will be used for the LPC analysis.
$analen$	Size of the frame used for the LPC analysis, $analen = 2 \cdot N_d \cdot framelen + winlen$, with N_d being the number of frames delay between in and output.
p	LPC order: defines the number of coefficients used for the LPC analysis.

Chapter 4

Algorithm : Block by block

This section aims to describe in detail each block represented in Figure 3.3. Even though some blocks are self-explanatory, for the sake of consistency they are all listed below.

4.1 Downsampling

Before any processing is applied to the signal, the signal is downsampled by factor $M = 4$. The main reason for this is to reduce the overall latency of the sound card. In fact, the overall feedback latency strongly depends on the sampling rate and on the buffer size of the sound card. Most soundboards allow a variable buffer size[1], but only more sophisticated models support low sampling rates down to about $10kHz$. In fact, the most common supported sampling rate is $Fs = 44,1kHz$ which is the established CD standard. Unfortunately this sampling rate is not optimal for formant estimation, for which the commonly used sampling rate is about $Fs = 8 \sim 16kHz$. Hence, downsampling becomes almost mandatory for sound cards which do not support these low sampling rates, but even for those who do, downsampling becomes very interesting when it comes to real-time applications like ours.

Assuming that the computer has endless processing power, the latency of the algorithm is infinitely small and depends neither on the sampling rate nor on the buffer size. However, the sound card delay is strongly dependent on both the sampling rate and the buffer size. While the buffer size is proportional to the latency — the larger the buffer, the longer the delay — the relationship between sampling rate and sound card latency are not linear. The reason for this is the internal hardware structure of the sound card.

In fact, most codecs work with a single, fixed oscillator (quartz) which provides the system's internal clock[2]. The signal is sampled at this particular oscillator rate and then internally downsampled to the desired sample rate F_s. To avoid aliasing, the signal is low-pass filtered before downsampling. This is generally done with a linear phase FIR filter, which guarantees highest quality audio recording. However, the disadvantage of this FIR filtering scheme is the necessity of high filter order p_{filt} to obtain an optimal low-pass characteristic, i.e. strong attenuation in the stop band, sharp slope in the transition

[1] Generally supported buffer sizes are : 64,128,256,512 samples
[2] Mostly 48, 96or 192 kHz

Description	**Shortcut**	**Value**
System's samplerate	F_s	$48\,kHz$
Cut off frequency	F_{pass}	$5770\,Hz$
Stop band frequency	F_{stop}	$6000\,Hz$
Maximal pass band ripple	A_{pass}	$1\,dB$
Minimal stop band attenuation	A_{stop}	$80\,dB$

Table 4.1: Filter properties

band, etc. Since the delay introduced by such a linear phase FIR filter is related to the filter order by $T_{filter} = p_{filt}/(2F_s)$, a high filter order causes a high delay between input and output.

Furthermore, we know that the stop band of the filter should start at $F_{s_{\downarrow M}}/2$ to avoid aliasing, where $F_{s_{\downarrow M}}$ is the desired downsampled rate. It is obvious that the filter order must be raised when lowering the downsampled rate to achieve the same qualitative filtering. Thus, the overall delay of the sound card strongly increases for low sampling rates.

This filter delay adds up with the delay introduced by the buffering. In fact, for one and the same buffer size, the latency introduced by the buffering scheme will linearly increase towards low frequencies. This is because a buffer containing N samples at a low sampling rate represents more time than at a high rate. Since the buffer of the sound card must be filled before being sent to the computer, the delay introduced by the buffering on each side[3] is $T_{buffer} = N/F_s$.

Thus, the overall delay of the sound card for a complete record to playback feedback loop is :

$$T_{soundcard} = T_{A/D} + T_{filter} + T_{buffer} + T_{D/A} \tag{4.1}$$

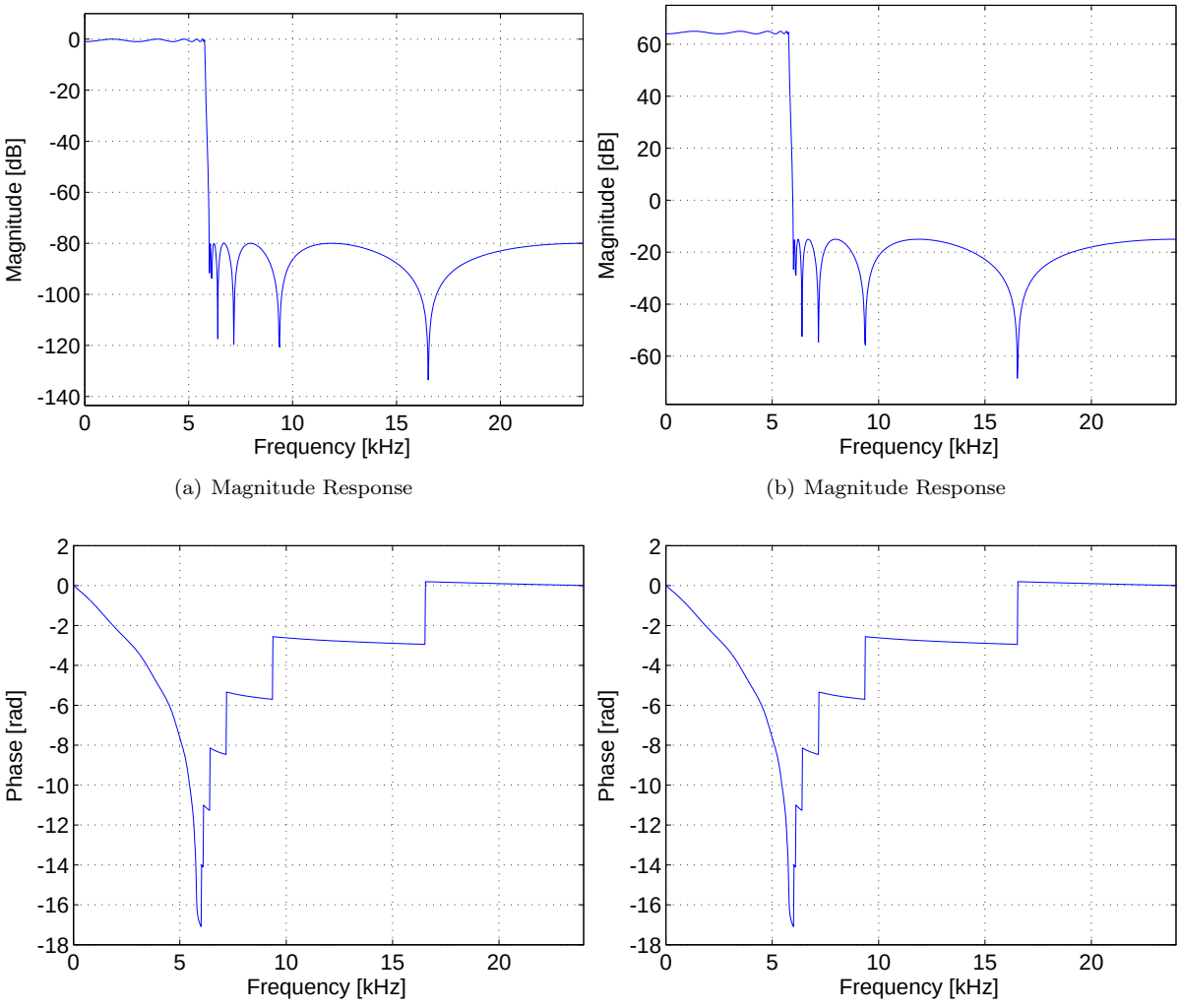
Where $T_{A/D}$ and $T_{D/A}$ are the delays of the analog / digital converters, which are influenced by neither the sampling rate nor the buffer size.

We see that at constant buffer size, halving the sampling rate will double the delay caused by the buffering, and will at least double the delay caused by the filter. We verified this relationship by measuring the input to output latency at various sample rates with a fixed buffer size of 64 samples. We measured a delay of $12\,ms$ at $F_s = 16\,kHz$, $8\,ms$ at $F_s = 32\,kHz$ and $4\,ms$ at $48\,kHz$. We see that the latency at $48\,kHz$ is about $8\,ms$ lower than at $16\,kHz$. In our application these $8\,ms$ are of course very precious; we will save them by downsampling the signal.

4.1.1 Low pass filter

An essential condition for the internal downsampling to be worthwhile is to keep the delay introduced by low-pass filtering small, especially because the filtering will be done twice, before downsampling and after upsampling. Using a minimum phase IIR filter will satisfy this criteria. We chose a 12^{th}order elliptic IIR filter, whose frequency response is represented in Figure 4.1. The filter's properties are listed in Table 4.1. Strictly speaking, an elliptic filter is not a minimum phase filter since all its zeros are not inside but on the unit circle, but it can be considered to be "almost" minimum phase. In fact, changing the zeros radius from 1 to, let's say, $1 - 10^{12}$ would make it a minimum phase filter, and this has almost no

[3]Recording and playback



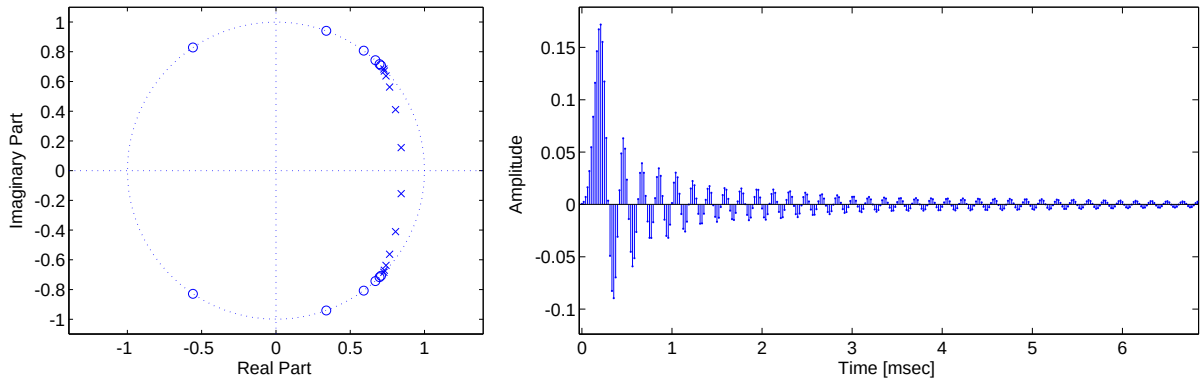
(a) Magnitude Response

(b) Magnitude Response

(c) Phase Response

(d) Phase Response

Figure 4.1: Elliptic low pass filter (left) and modified filter (right)



(a) Poles (x) and zeros (o) in the z plane

(b) Impulse response

Figure 4.2: Elliptic filter: Poles and zeros in the z plane (left) and impulse response (right)

influence on the phase as we can see in Figure 4.1. The gain of the filter changes considerably, though, which we could have expected since we moved the zeros from infinite attenuation to finite attenuation, and this of course dramatically changes the overall gain of the filter. Figure 4.2 shows the poles and zeros of the elliptic filter in the complex z-plane . We can see that all the poles are inside and all the zeros on the unit circle.

The main advantage of the elliptic filter is its very sharp slope at cutoff frequency and its high stop band attenuation, both of which can be achieved with a relatively small filter order. However, the elliptic filter has a very strong phase distortion near the cutoff frequency, which usually appears as ringing[4] near that frequency in the time domain. Nevertheless, this ringing is only perceptible by the human ear if the time delay between the main sound (frequencies in the "linear" phase part) and the ringing (frequencies with distorted phase) is above the so called time-discrimination threshold of hearing. A good rule of thumb is to keep the total impulse-response duration below the time-discrimination threshold of hearing, which according to [3] is about 5 to $10ms$. As we can see in the right panel of Figure 4.2 this rule of thumb is respected, as the impulse response is already very attenuated at $5ms$.

4.1.2 Decimation

After having low pass filtered the signal, we can decimate by factor $M = 4$ without creating any spectral overlap, since all frequencies above the new Nyquist frequency $F_{s_{\downarrow M}}/2 = F_s/8 = 6000\,Hz$ have been removed. Thus the new downsampled signal $s_{\downarrow M}$ can be written as follows:

$$s_{\downarrow M}(k) = s(Mk) \quad , k = 0, 1, \ldots, framelen - 1 \tag{4.2}$$

4.2 Preemphasis

According to [1] the speech signal is attenuated towards high frequencies with a slope of $-6dB/oct$. In order to compensate this effect it is necessary to apply a so-called preemphasis filter before performing

[4] This usually makes the elliptic filter useless for high end audio applications

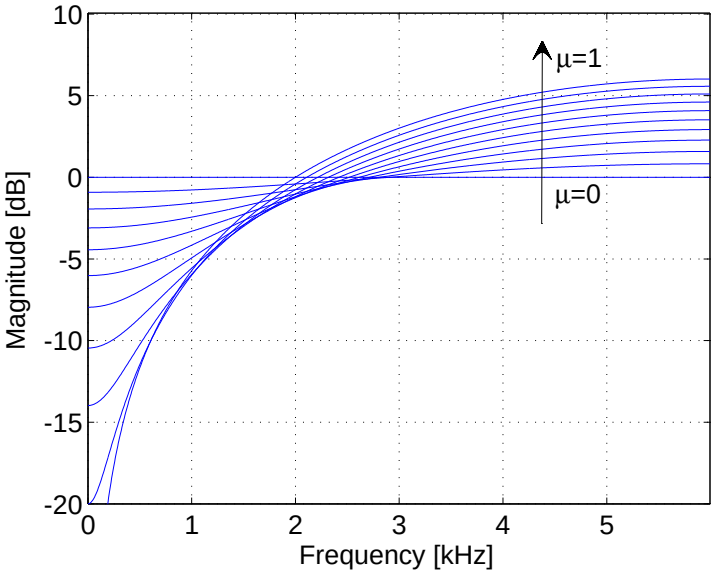


Figure 4.3: Frequency Response of $R(z)$ (preemphasis filter)

the LPC analysis. This will boost higher frequencies and hence improve the formant estimation. The transfer function $R(z)$ of such a filter is presented in Equation 4.3, where μ is an adjustable preemphasis factor.

$$R(z) = 1 - \mu z^{-1} \tag{4.3}$$

We can see the frequency response of such a filter in Figure 4.3 , with μ varying between 0 and 1.

4.3 Input buffer

The main purpose of the input buffer is to provide enough samples for the LPC analysis. This is because the LPC method requires an analysis window that is greater than the downsampled IO frame length of the sound card. The IO frame length should be as small as possible to reduce the overall latency of the algorithm. The particular buffer structure allows us being very flexible with regard to the processing of each incoming frame. Namely, it enables us to adjust the overall process delay, the length of the LPC analysis window and the number of processes per sec with only 2 parameters, which can be set directly from the Matlab environment.

4.3.1 Buffer structure

Figure 4.4 shows the structure of the input buffer. The buffer consists of an odd number of blocks, each of which contains $framelen$ samples:

$$buflen = (2 \cdot N_d - 1) \cdot framelen \tag{4.4}$$

N_d is the effective number of delay frames before an incoming frame is processed and sent to the output. Thus the overall process delay is $N_d \cdot framelen$ samples.

Each time the algorithm is called back by the sound card, the samples in the buffer are moved forward by $framelen$ samples as illustrated in Figure 4.4. An incoming frame will only be processed once arrived in block number N_d. This frame is than divided into N_w smaller frames of the size of $winlen$ which are then individually processed. Thereby the maximal gap between two possible LPC analysis lengths is only $framelen/F_{s_{\downarrow M}}$

Figure 4.4 illustrates how the samples in the buffer are used during the processing. The LPC analysis window (represented in red) is symmetric, thus consisting of a causal and non-causal part with regard to the frame to be processed (in blue). This particular windowing structure will provide the most accurate formant estimation. In fact, samples in the middle of the hanning window (i.e. within the frame to be processed) have a much greater weight than samples on the border of the hanning window during the LPC analysis. The disadvantage is that an incoming frame cannot be processed until all samples needed for the LPC analysis (i.e the non-causal part) have arrived into the buffer. The result is that the process delay is greater than for the non-symmetric analysis scheme, in our case $4ms$. Nevertheless, we accept this additional delay for the following reason:

In our experiment, we are shifting an entire formant trajectory. The formants change rapidly in time during a vowel transition. Thus, the estimated filter coefficients should be representative of the actual frame and not rely on an estimate based only on past samples.

4.4 Long-Time RMS

The purpose of this block is to provide long-time RMS values of the original $s_{org}(i)$ and preemphasized signal $s_{pre}(i)$ for the vowel detection. To reduce calculation, the long-time RMS is calculated on $framelen$ samples and exponentially smoothed using a forgetting factor λ.

$$RMS_{pre}(k) = (1-\lambda) \cdot \sqrt{\frac{1}{framelen} \sum_{i=0}^{framelen-1} s_{pre}^2(i)} + \lambda \cdot RMS_{pre}(k-1) \tag{4.5}$$

$$RMS_{org}(k) = (1-\lambda) \cdot \sqrt{\frac{1}{framelen} \sum_{i=0}^{framelen-1} s_{org}^2(i)} + \lambda \cdot RMS_{org}(k-1) \tag{4.6}$$

where $s(i)$ is the i^{th} sample of the current frame k starting at $i = 0$.

4.5 Vowel Detection

Since we are only interested in perturbing a vowel transition we need to separate voiced from unvoiced sounds. Two simple conditions are sufficient to build a simple vowel detection:

Because of the vibration of the vocal folds during a voiced sound, we expect the RMS value to be higher than for unvoiced sounds. This leads to the first necessary condition:

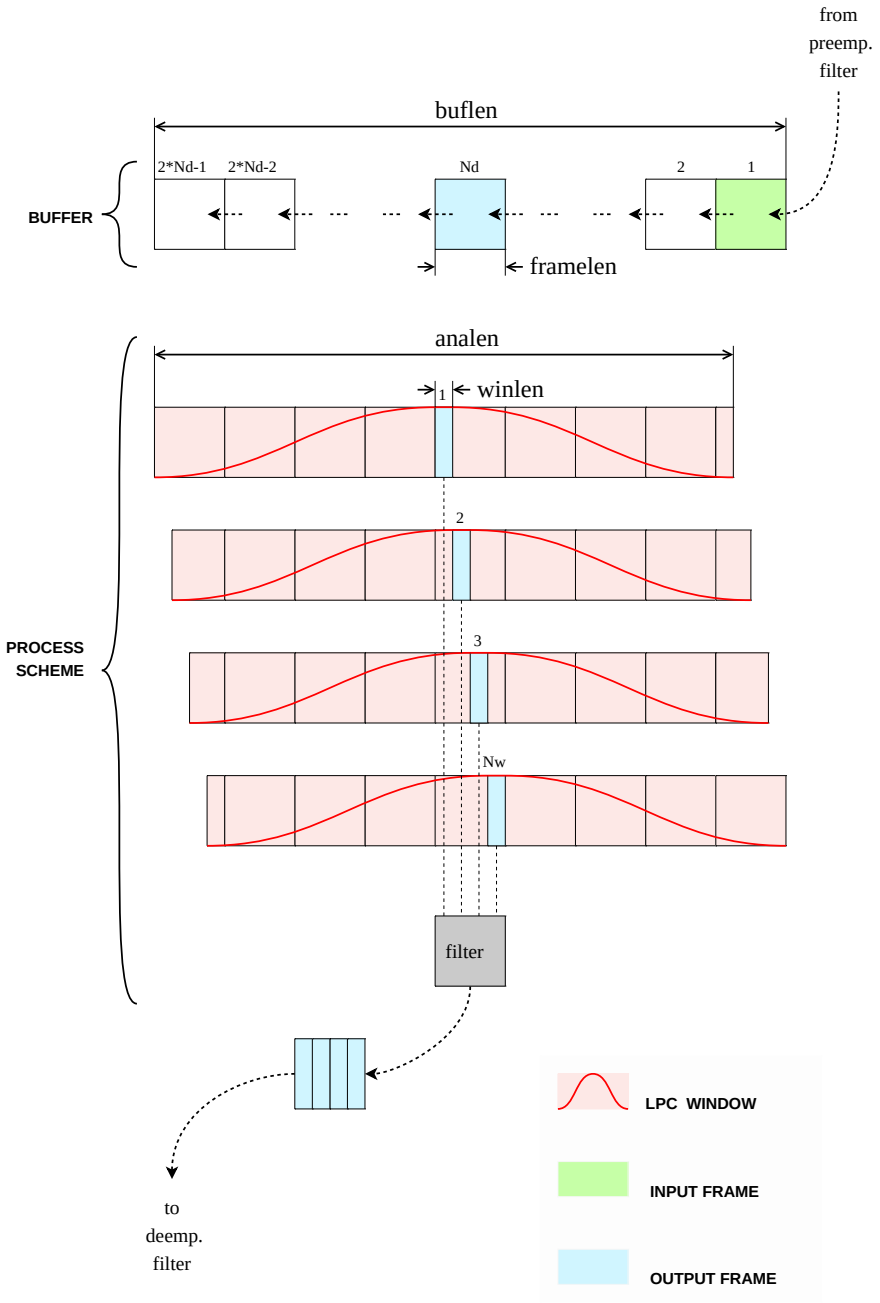


Figure 4.4: Input Buffer & Process Scheme

The input buffer is represented in the upper part of this Figure. It is compound of an uneven number of frames of framelen samples. The size of the buffer can be changed in order to obtain the desired LPC analysis length (represented below in red). Each incoming frame (green block) is delayed before being processed (blue block), so that the LPC analysis can be performed on a symmetric window around the frame to be processed. Below the input buffer we have represented the associated processing scheme, to demonstrate that each frame can be divided in N_w (here $N_w = 4$) smaller frames that are then individually processed. This is to allow a processing at a higher rate than the one imposed by the sound card.

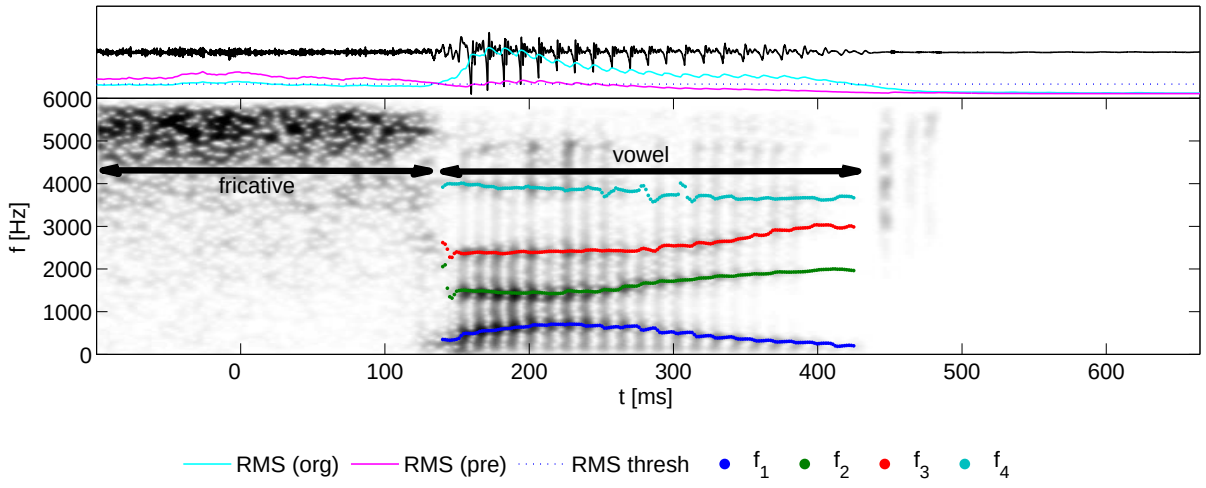


Figure 4.5: Vowel detection

In the upper panel we can see the audio signal, the course of RMS_{org} (magenta) and RMS_{pre} (blue) and also the voiced / unvoiced threshold RMS_{thresh}(dotted line). The bottom panel contains the spectrogram as well as the estimated formant tracks, whenever a vocalic region has been detected. The first part of the signal is the fricative [s], where we can see a high energy concentration at 5 to 8 kHz on the spectrogram. Since the pre-emphasis filter will considerably boost these high frequencies we expect RMS_{pre} to be higher than RMS_{org} during this period, which can be verified in the upper panel. We can see that, even though RMS_{org} is above RMS_{thresh} the formant estimation is disabled for this part of the signal, because $RMS_{org}(k) < RMS_{pre}(k) \cdot RMSratio_{thresh}$. The LPC analysis starts as soon as this high frequency energy has disappeared. We then see that RMS_{org} gets above both RMS_{pre} and RMS_{thresh} which is an unambiguous sign for the existence of a vowel.

$$RMS_{org}(k) > RMS_{thresh} \tag{4.7}$$

where RMS_{thresh} is an adjustable threshold value that is determined during an RMS calibration phase.

This first criterion is a very simple way to detect voicing; however, it fails for fricatives such as [s]. During a fricative, the RMS value exceeds the voicing threshold, because of the continuous noise source created at the constriction of the articulators (see Section 2.2.2). Nevertheless, there is an easy method to distinguish a fricative from a vowel. We recall that the energy concentration of a fricative is at high frequencies, whereas in a vowel, the main energy is concentrated in lower frequencies (see Figure 2.2). Thus we can formulate the second condition:

$$RMS_{org}(k) > RMS_{pre}(k) \cdot RMSratio_{thresh} \tag{4.8}$$

$RMSratio_{thresh}$is an adjustable ratio threshold value[5].

If both conditions are verified, we can proceed from the assumption that the analyzed part of the signal is indeed a voiced sound. The vowel detection then enables the LPC analysis and formant tracking (i.e. Block 5-11). We can see an example detection scenario in Figure 4.5 for the word “site”.

[5]We empirically determined $RMSratio_{thresh} = 1.3$

4.6 LPC Analysis

A widespread method for analyzing and modeling speech signals is the LPC Method [9]. This method relies on the fact that the human vocal tract can be represented as a time-variable filter excited by one or more sources (see Chapter 2). The LPC method is a widely used method in speech analysis, mostly to extract formants of vowels. The LPC method yields an estimation of the vocal tract filter transfer function $T(f)$ as a strictly all-pole model. Since the vocal tract creates resonances, which can be modeled as complex conjugated poles, the LPC provides a very good estimate of the "true" vocal tract resonances.

According to [9] the basic discrete-time model for speech production can be described as a digital filter whose steady-state system function is

$$H(z) = \frac{S(z)}{U(z)} = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}} \tag{4.9}$$

For voiced speech, the parameters of this model are pitch period T_0, gain parameter G and the coefficients $\{a_k\}$ of the digital filter.

In the time domain Equation. 4.9 is described by the difference equation

$$s(n) = \sum_{k=1}^{p} a_k s(n-k) + Gu(n) \tag{4.10}$$

A linear predictor with prediction coefficients $\alpha_k$is defined as a system whose output is

$$\tilde{s}(n) = \sum_{k=1}^{p} \alpha_k s(n-k) \tag{4.11}$$

Its system function is

$$P(z) = \sum_{k=1}^{p} \alpha_k z^{-k} \tag{4.12}$$

The predictor error, $e(n)$, is defined as

$$e(n) = s(n) - \tilde{s}(n) = s(n) - \sum_{k=1}^{p} \alpha_k s(n-k) \tag{4.13}$$

From Eq. 4.13 it can be seen that the prediction error sequence is the output of a system whose transfer function is

$$A(z) = 1 - \sum_{k=1}^{p} \alpha_k z^{-k} \tag{4.14}$$

When the speech signal corresponds exactly to the model of Equation 4.10 , and if $\alpha_k = a_k$, then $e(n) = Gu(n)$. Thus, the prediction error filter, $A(z)$, will be an inverse filter for the system, $H(z)$, of Equation 4.9, i.e.,

$$H(z) = \frac{G}{A(z)} \tag{4.15}$$

This means that if we are able to determine the appropriate coefficients $\{\alpha_k\}$ from the speech signal, we will obtain a good estimate of the spectral properties of the speech signal. The question is: How can we obtain these coefficients?

4.6.1 Finding the coefficients

A basic approach for resolving this issue is to find a set of predictor coefficients that will minimize the mean-squared prediction error. The short-time average prediction error is defined as

$$E_n = \sum_m e_n^2(m) \tag{4.16}$$

$$= \sum_m (s_n(m) - \tilde{s}_n(m))^2 \tag{4.17}$$

$$= \sum_m \Big(s_n(m) - \sum_{k=1}^{p} \alpha_k s_n(m-k) \Big)^2 \tag{4.18}$$

where $s_n(m)$ is a segment of speech that has been selected in the vicinity of sample n, i.e.,

$$s_n = s(m+n) \tag{4.19}$$

We can find the values of α_k that minimize E_n in Equation 4.18 by setting $\partial E_m / \partial \alpha_i = 0$, for $i = 1, 2, ..., p$, thereby obtaining the equations

$$\sum_m s_n(m-i)s_n(m) = \sum_{k=1}^{p} \alpha_k \sum_m s_n(m-i)s_n(m-k) \quad 1 \leq i \leq p \tag{4.20}$$

Using Equations 4.18 and 4.20, the minimum mean-squared prediction error can be shown to be

$$E_n = \sum_m s_n^2(m) - \sum_{k=1}^{p} \alpha_k \sum_m s_n(m)s_n(m-k) \tag{4.21}$$

Equation 4.20 allows us to find the value of the α_k coefficients, but yet two parameters are still undefined: the value of p, i.e. the number of predictor coefficients, and the portion $s_n(m)$ of the speech signal to be analyzed. The authors of [9] describe two methods for determining these parameters: the autocorrelation method and the covariance method. In the next section we will briefly introduce the autocorrelation method, as this is the method we are using in our algorithm.

4.6.2 The autocorrelation method

Using a windowing function $w(m)$ that is identically zero outside the interval $0 \leq m \leq N-1$ with

$$s_n(m) = s(m+n)w(m) \tag{4.22}$$

it can be shown that the short-time average prediction error becomes

$$E_n = \sum_{m=0}^{N+p-1} e_n^2(m) \tag{4.23}$$

In this case, the authors of [9] show that Equation 4.20 can be expressed as

$$\sum_{k=1}^{p} \alpha_k R_n(|i-k|) = R_n(i) \qquad 1 \leq i \leq p \tag{4.24}$$

where $R_n(k)$ is the short-time autocorrelation function, with

$$R_n(k) = \sum_{m=0}^{N-1-k} s_n(m) s_n(m+k) \tag{4.25}$$

Furthermore, Equation 4.24 can be expressed in matrix form as

$$\begin{bmatrix} R_n(0) & R_n(1) & R_n(2) & \cdots & R_n(p-1) \\ R_n(1) & R_n(0) & R_n(1) & \cdots & R_n(p-2) \\ R_n(2) & R_n(1) & R_n(0) & \cdots & R_n(p-3) \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ R_n(p-1) & R_n(p-2) & R_n(p-3) & \cdots & R_n(0) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \cdots \\ \alpha_4 \end{bmatrix} = \begin{bmatrix} R_n(1) \\ R_n(2) \\ R_n(3) \\ \cdots \\ R_n(p) \end{bmatrix} \tag{4.26}$$

This is a $p \times p$ Toeplitz matrix, i.e. it is symmetric and all the elements along a given diagonal are equal.

4.6.3 Levinson Durbin recursion

An efficient method for solving the particular system of equations given by Equation 4.26 is described in [6] and can be stated as follows:

$$E^{(0)} = R(0) \tag{4.27}$$

$$k_i = \Big(R(i) - \sum_{j=1}^{i-1} \alpha_j^{(i-1)} - R(i-j)\Big) / E^{(i-1)} \qquad 1 \leq i \leq p \tag{4.28}$$

$$\alpha_i^{(i)} = k_i \tag{4.29}$$

$$\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i \alpha_{i-j}^{(i-1)} \qquad 1 \leq j \leq i-1 \tag{4.30}$$

$$E^{(i)} = (1 - k_i^2) E^{(i-1)} \tag{4.31}$$

These equations are solved recursively for $i = 1, 2, \ldots, p$ and the final solution is given as

$$\alpha_j = \alpha_j^{(p)} \qquad 1 \leq j \leq p \tag{4.32}$$

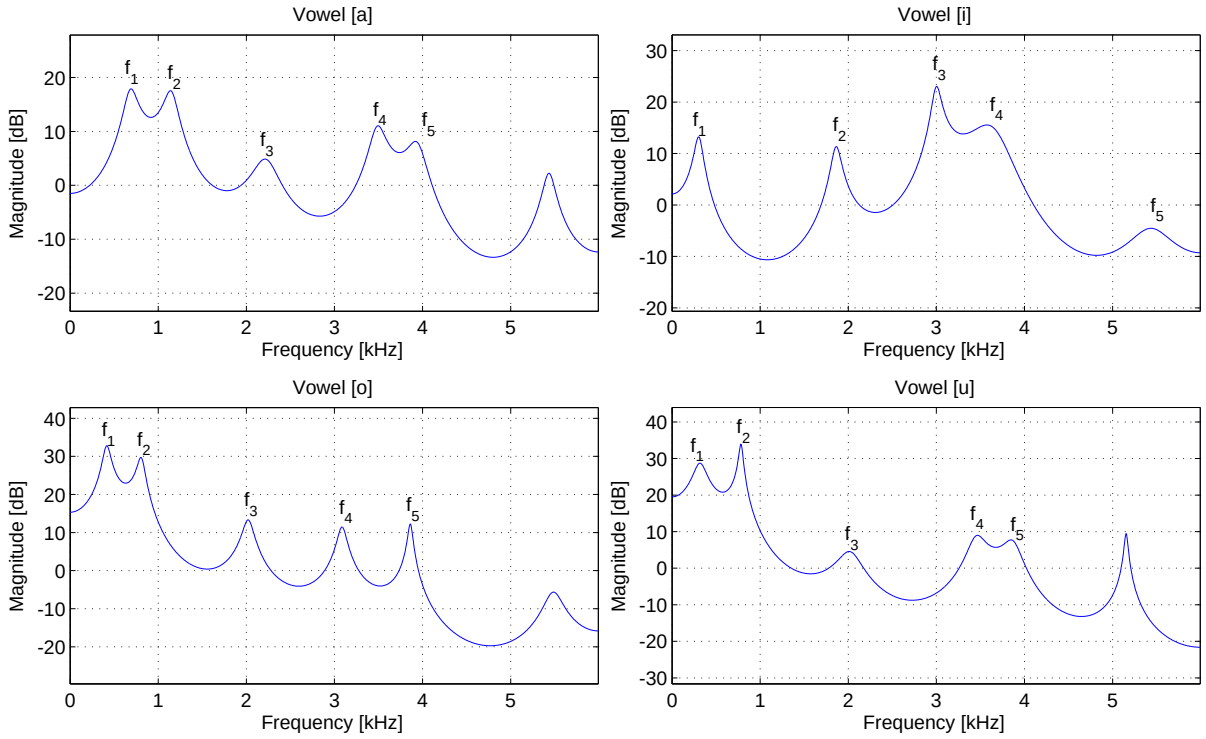


Figure 4.6: LPC estimated magnitude response for the vowel [a] , [o], [i] and [u]

4.6.4 LPC spectral estimation

The LPC method yields the prediction coefficients α_k which provides an estimation of the vocal tract transfer function $V(z)$:

$$V(z) = \frac{1}{1 + \sum_{k=1}^{p} \alpha_k z^{-k}} \tag{4.33}$$

Evaluating $|V(z)|$ on the unit circle, i.e. for $z = e^{j\omega}$, provides the magnitude response of the estimated vocal tract filter. Figure 4.6 shows the spectral estimation of several vowels. Each peak of the spectral envelope corresponds to a formant frequency denoted f_k. Principally, two methods are available to obtain these formants:

- Compute the magnitude response $|V(z = e^{j\omega})|$ and search for local maxima's.
- Find the roots of the polynomial of $V(z)$.

We decided to utilize the first method because it will allow us to use the poles of the filter transfer function to perform the formant shift[6].

[6]See Section 4.16.2 for more information.

4.7 Root finding algorithm

The fundamental theorem of algebra states that a polynomial $P(z)$ of degree n has n roots, some of which may be degenerate. Finding roots of a polynomial is therefore equivalent to polynomial factorization into factors of degree 1. Since the coefficients of this polynomial are real, the roots can only be real, or appear as complex conjugated pairs. Thus Equation 4.33 can be written as follows

$$V(z) = V_c(z) \cdot V_r(z) \tag{4.34}$$

Where $V_c(z)$ contains only complex conjugated pole pairs $\{c_k, c_k^\star\}$,

$$V_c(z) = \frac{1}{\prod_{k=1}^{M}(1 - c_k z^{-1})(1 - c_k^\star z^{-1})} \tag{4.35}$$

and $V_r(z)$ only real poles c_i.

$$V_r(z) = \frac{1}{\prod_{i=1}^{N}(1 - c_i z^{-1})} \tag{4.36}$$

N and M are related to the LPC order P by $P = 2M + N$.

4.7.1 Eigenvalue method

We implemented a polynomial root-finding algorithm described in [7], which uses the eigenvalue method. The eigenvalues of a matrix A are the roots of the "characteristic polynomial" $P(x) = det[A - xI]$. It can be verified that the characteristic polynomial of the special $p \times p$ companion matrix

$$A = \begin{pmatrix} -\frac{a_{p-1}}{a_p} & -\frac{a_{p-2}}{a_p} & \cdots & -\frac{a_1}{a_p} & -\frac{a_0}{a_p} \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \tag{4.37}$$

is equivalent to the general polynomial

$$P(x) = \sum_{i=0}^{p} a_i x^i \tag{4.38}$$

The root-finding algorithm first generates the companion matrix A, based on the polynomial $P(x)$, where the coefficients a_i are the estimated LPC coefficients α_j, for $i = j = 1, 2, ..., p$. The algorithm then iteratively finds the eigenvalues of the Matrix A which are utilized to determinate the roots of $P(x)$ and finally the poles $c_k$of the vocal tract filter function $V(z)$. It is a powerful, robust, and widely used method for low and moderate degree polynomials.

4.7.2 Roots sorting

The root-finding algorithm returns a set of poles $c_k = c_{k_{real}} + ic_{k_{im}}$, where $c_{k_{real}}$ is the real part and $c_{k_{im}}$ the imaginary parts of the poles. As we are only interested in complex conjugated poles, we discard all the poles with $c_{k_{im}} = 0$, which means that all remaining poles are complex conjugated pole pairs $\{c_k, c_k^*\}$ as stated in Equation 4.35. We then transform the remaining poles into polar coordinates and obtain the following representation:

$$c_k = r_k \cdot e^{j\theta_k} \tag{4.39}$$

where r_k is the radius:

$$r_k = \sqrt{(c_{k_{real}}^2 + c_{k_{im}}^2)} \tag{4.40}$$

and θ_k the complex angle of the pole:

$$\theta_k = \arctan\left(\frac{c_{k_{im}}}{c_{k_{real}}}\right) \tag{4.41}$$

The angle of the pole indicates a resonance of the filter function, each of which can be a potential formant frequency f_k, with $f_k = \frac{\theta_k \cdot F_{s_{\downarrow M}}}{2\pi}$, $F_{s_{\downarrow M}}$ being the internal samplerate (i.e. after downsampling).

Figure 4.7 to 4.9 show estimated spectral envelopes of the vocal tract filter function for the vowel [a] (right panel of (b)) and the associated poles of the transfer function (left panel of (b)) obtained by the root-finding and sorting algorithm.

4.8 Formant tracking algorithm

We recall that the LPC analysis provides the best-fit all-pole approximation of the vocal tract filter, where the best fit is obtained by minimizing the quadratic error between the estimated signal and the real signal (Section 4.6). Since the "real" speech spectrum is made of peaks located at the resonance frequencies of the vocal tract, it is very likely that the best-fit approximation of the LPC will contain complex conjugated poles at these frequencies. However, this is not guaranteed, as we will see in the next sections.

4.8.1 Influence of the LPC order p on formant estimation

Choosing the wrong number of LPC coefficients may result in erroneous formant frequencies estimates. If p is too low, the LPC estimation will be very poor, and hence formant frequencies will not be accurate. On the other hand, choosing a high LPC order is most likely to introduce complex conjugated poles that do not represent a formant frequency. Since we expect approximately 5˜6 formants between 0 and 6 Khz, the minimal LPC order should be $p = 10 \sim 12$, to allow at least 5˜6 complex conjugated poles in the transfer function. Increasing the LPC order above that minimal order will surely improve the spectral estimation provided by the LPC, but at the same time will introduce poles that do not correspond to formants. Figure 4.7 shows estimated formant tracks for $p = 10$, Figure 4.8 for $p = 12$ and Figure 4.9 $p = 14$ for the vowel [a] without using any formant tracking algorithm. We see that for $p = 10$, formant tracks are skipped because of poor LPC estimate, for $p = 12$ everything is fine, and for $p = 14$ additional

erroneous formant tracks appear.

A rule of thumb states that the order should be $p \approx F_s/1000$, i.e. $p = 12$[7] . Some more sophisticated heuristics like the one described in [11] estimate the LPC order based on an frame-to-frame signal analysis. However, even when using an optimal LPC order, "wrong" formants will occasionally be determined.

4.8.2 Bandwidth considerations for improved formant tracking

In Figure 4.7 to 4.9 we see that increasing the LPC order also yields a better estimation of the vocal tract filter response. We know that a filter transfer function with a pair of complex conjugated poles is a resonant filter. The frequency at the resonance is defined by the angle and its gain by the radius of the poles. The closer the complex conjugated poles get to the unit circle in the z plane, the higher the peak of the resonance will be.

Figure 4.10 shows how radius of the poles and peakiness of the resonance are related. Usually, the peakiness of a formant is described by its bandwidth, which is related to the radius of the poles as follows:

$$B_k = -\log_{10}(r_k) \cdot \frac{F_{s_{\downarrow M}}}{\pi} \tag{4.42}$$

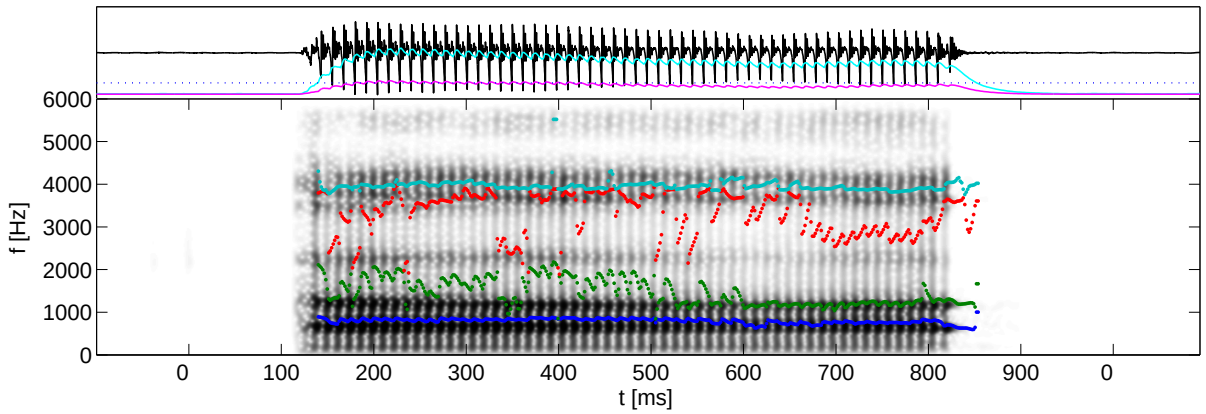
Complex conjugated pole pairs with low radius (i.e high bandwidth) will also be held as formants, since by now every resonance, even if it is a weak resonance, is interpreted as a formant. A very simple way to avoid this would be to determine a bandwidth threshold and to discard all formant candidates that have a bandwidth above this threshold. The problem is that there is no distinct border to separate them, and thus this solution is not satisfactory. We know that resonances with very high bandwidth are certainly not formants, but on the other hand, a resonance with small bandwidth is not necessarily a formant, as we will see in the next section. Nevertheless, the bandwidth is a very good indicator of whether a complex conjugated pole pair represents a formant or not, and will play an important role in our tracking algorithm.

4.8.3 Influence of fundamental frequency on formant estimation

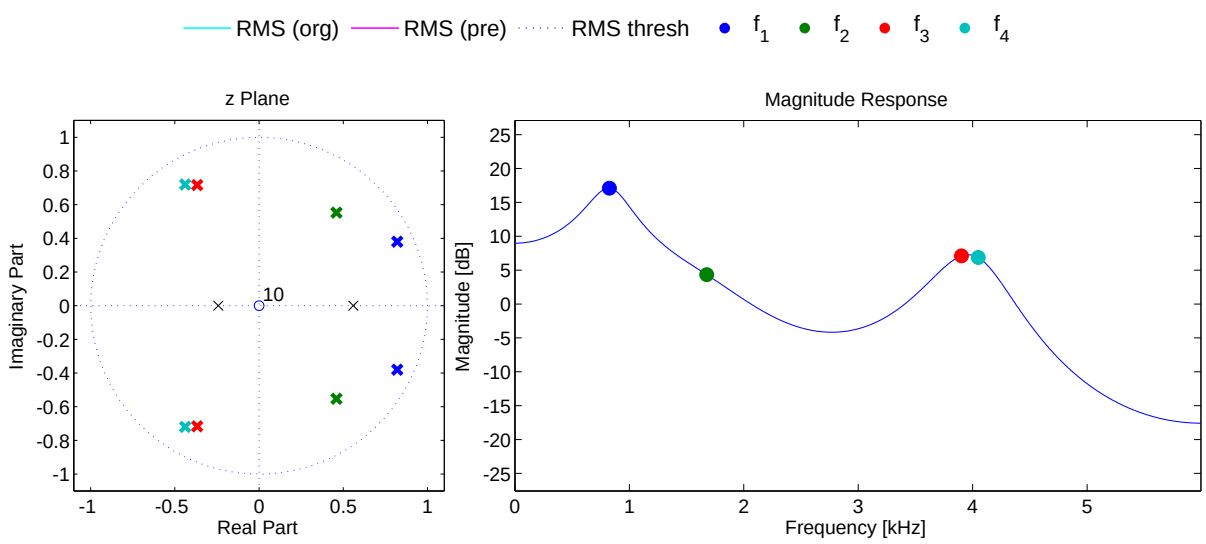
In the previous sections we learned that the LPC order p should be at least high enough to deliver an accurate spectral estimation of the speech signal. We have seen that increasing the LPC order will introduce additional potential formant candidates that are usually very easy to recognize because of their low gain peak. Thus, one strategy could be to overestimate the LPC order and then discard the erroneous formants. Unfortunately, sometimes increasing the order will also introduce more peaks, and thus bandwidth considerations alone will not help to distinguish them.

A factor that strongly affects formant estimation is the fundamental frequency F_0 . We recall that a vowel is principally the concatenation[8] of a periodic source, filtered by the vocal tract filter. Unfortunately,

[7] This would have been correct in our case, but this is not always the case.
[8] According to the source-filter model (Section 2.2)



(a) Audio signal (upper panel) , spectrogram and formant tracks (lower panel)



(b) Extract from (a) at t=220 ms : Poles in the z plane (left) and magnitude response (right)

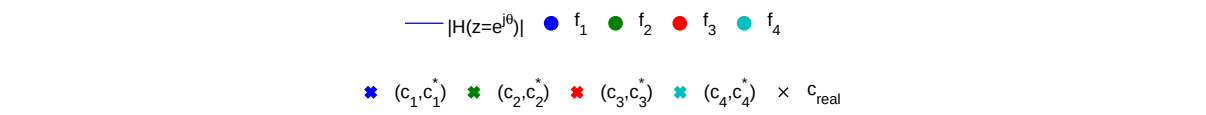
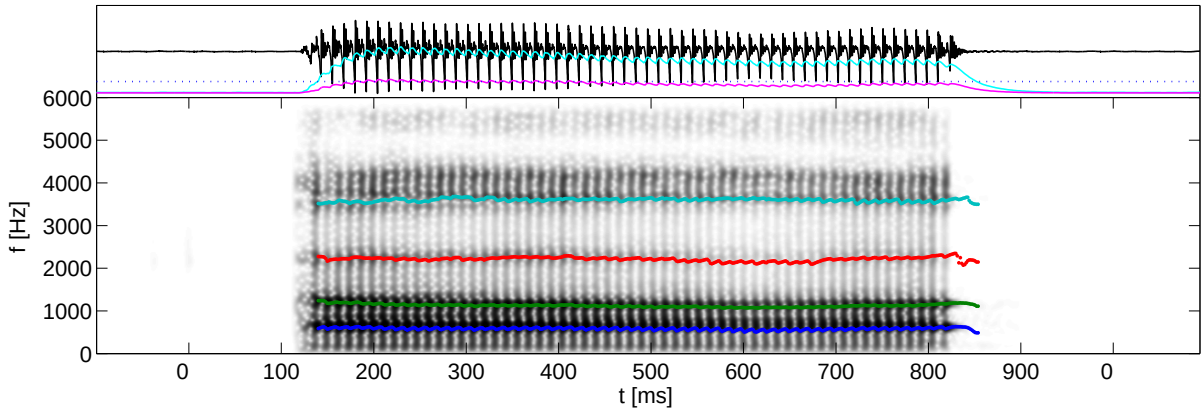
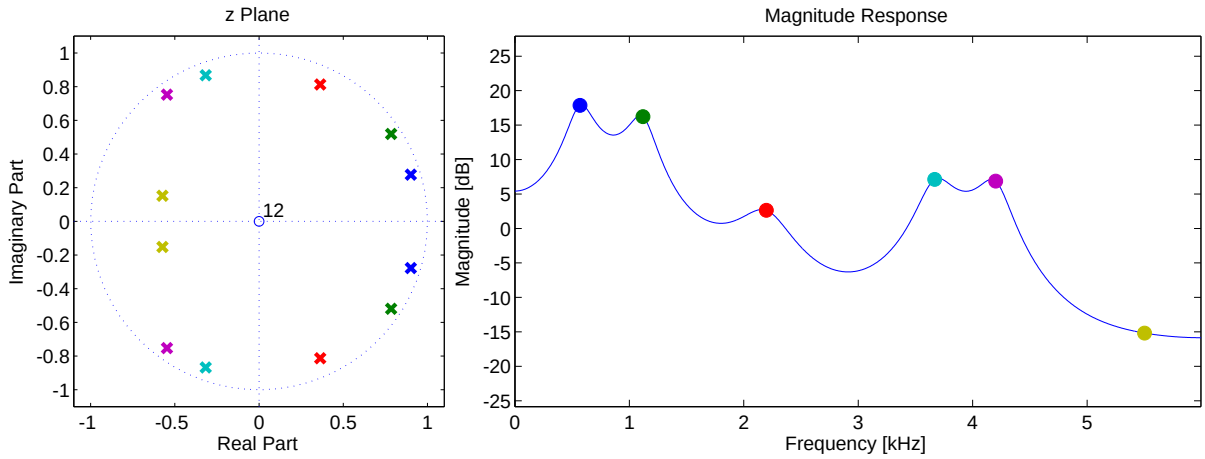


Figure 4.7: LPC estimation of the vowel [a], with LPC order $p = 10$:

Poor LPC estimation due to low LPC order. The resonances are not modeled adequately because the best fit LPC estimation with 10 poles yields 2 real poles (black cross, left panel of (b)) and only 4 complex conjugated pole pairs (colored cross, left panel of (b)) to model a spectrum where 6 formants are expected.



(a) Audio signal (upper panel) , spectrogram and formant tracks (lower panel)



(b) Extract from (a) at t=220 ms : Poles in the z plane (left) and magnitude response (right)

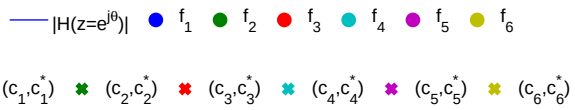
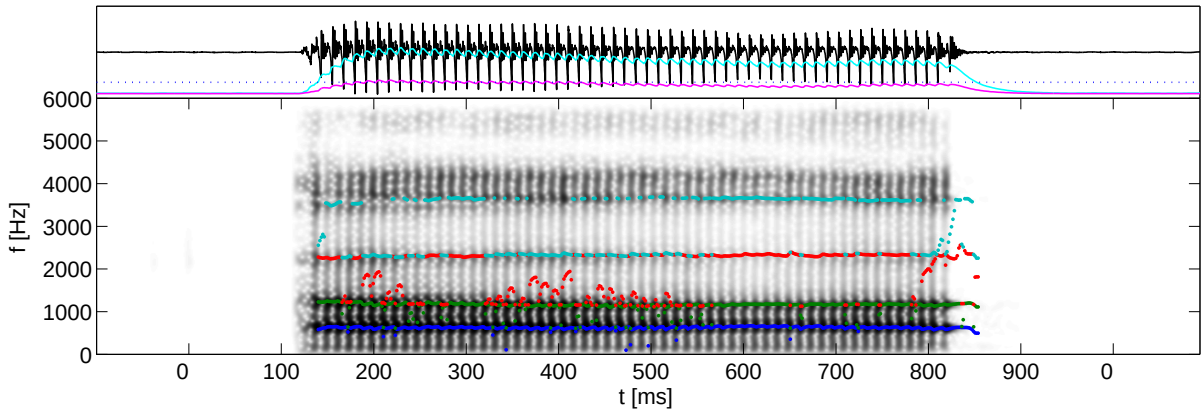
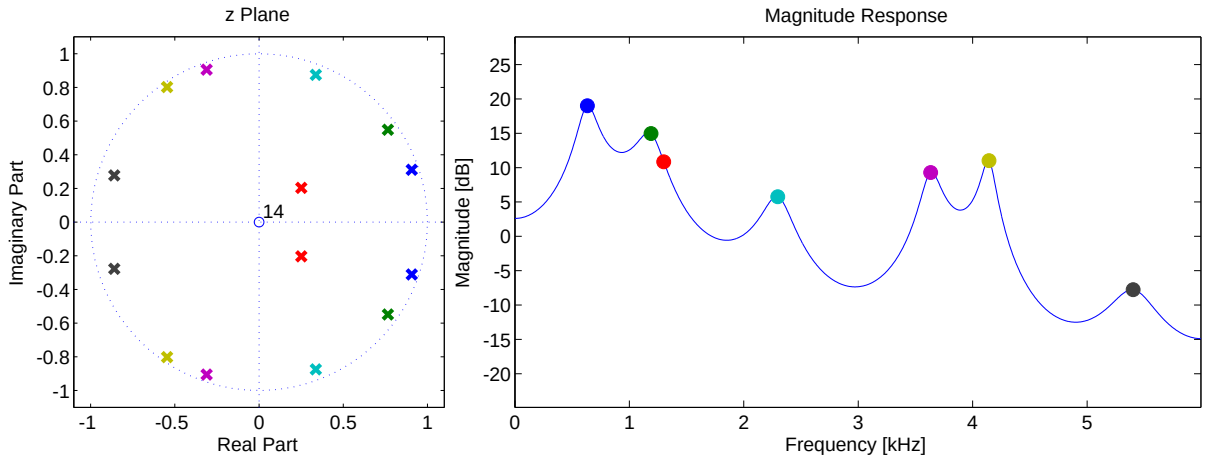


Figure 4.8: LPC estimation of the vowel [a], with LPC order $p = 12$:

All poles are complex conjugated pole pairs (left panel of (b)), most of which causes the filter magnitude response to have a peak, i.e. a formant (right panel of (b)). The resulting formant tracks are very clean as we can see in (a). Note that not every complex conjugated pole in the z plane has its peak (6^{th} pole pair for instance).



(a) Audio signal (upper panel) , spectrogram and formant tracks (lower panel)



(b) Extract from (a) at t=220 ms : Poles in the z plane (left) and magnitude response (right)

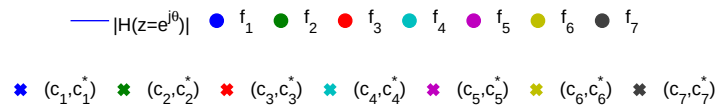


Figure 4.9: LPC estimation of the vowel [a], with LPC order $p = 14$:

Additional erroneous formant tracks appear as a result of high LPC order. The third formant (red dot, right panel of (b)) is not a true formant because it has no corresponding peak in the magnitude response. We can clearly see that the corresponding pole pair (red cross, left panel of (b)) has a very small radius, which means that the gain of the resonance is weak, i.e. its bandwidth is high.

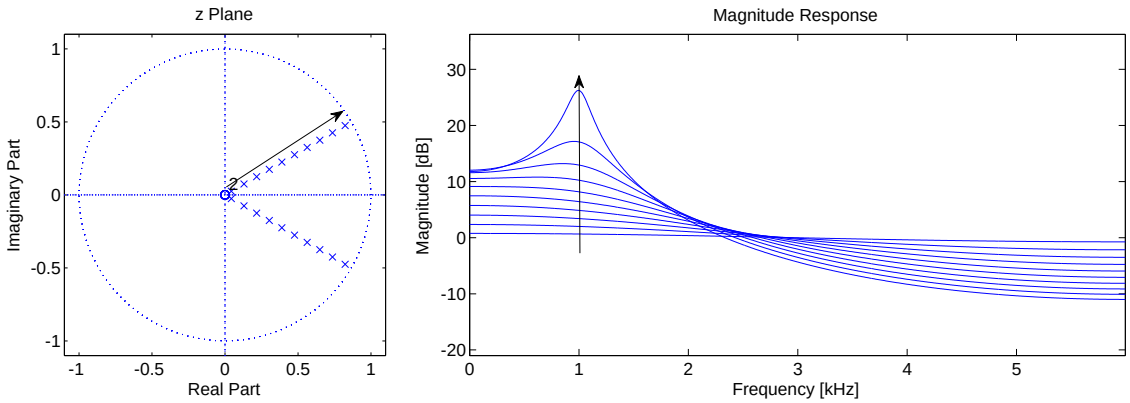
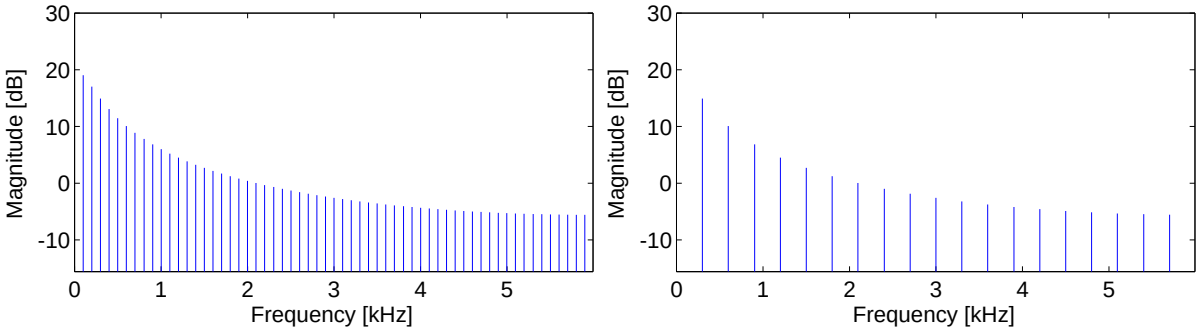


Figure 4.10: Single resonance at $f_{res} = 1000\,Hz$ with varying pole radius $(0.05 < r < 0.95)$

The gain of the peak increases as the pole's radius get closer to the unit circle. When r is small, the attenuation of the resonance is high and thus we do not see a peak in the magnitude response. However, we can clearly see the non linear relationship between pole's radius and bandwidth of the resonance. Note that for $r = 1$ the gain of the peak's resonance would become infinitely high.



(a) Male speaker : $F_0 = 100\,Hz$

(b) Female Speaker: $F_0 = 300\,Hz$

Figure 4.11: Source spectrum of a male and female speaker

Since the source is a periodic function caused by the vibration of the vocal folds, the spectrum is discrete in the frequency domain. High energy concentration appear every $k \cdot F_0$, i.e. at multiples of the fundamental frequency. The source of the male speaker has a higher frequency resolution compared to the female speaker. Extraction of the vocal tract filter function will get worse for increasing F_0 , because the overall spectrum will have more peaks.

the LPC is not a perfect estimate of the vocal tract filter function; it also contains information about the source. Figure 4.11 shows the typical source function for a male and a female speaker. For the male subject, we see that the spectral resolution is much higher than it is for the female subject. This spectral resolution difference affects the smoothness of the estimated LPC spectrum. For a female speaker, the estimated filter response will have more peaks, some of which will not be formants, but simply be located at harmonics of the fundamental frequency. The problem is that these "wrong" formants have a high resonance and hence it will be very difficult to distinguish them from "real" formants.

4.8.4 Physical constraints of the vocal tract

We know that formants are the result of resonances of the vocal tract. These resonances can only occur at specific frequencies, which are directly related to the shape of the vocal tract. Thus only certain combinations of formants are allowed. These relationships are described in [1]. In our tracking algorithm we will not use this relationships, but we will use the fact that each formant can only exist within a certain frequency range.

Another criterion to help determine whether a complex conjugated pole pair represents a formant or not will be based on time continuity constraints. We know that formants have smooth trajectories: even for vowel transitions there are no abrupt discontinuities.

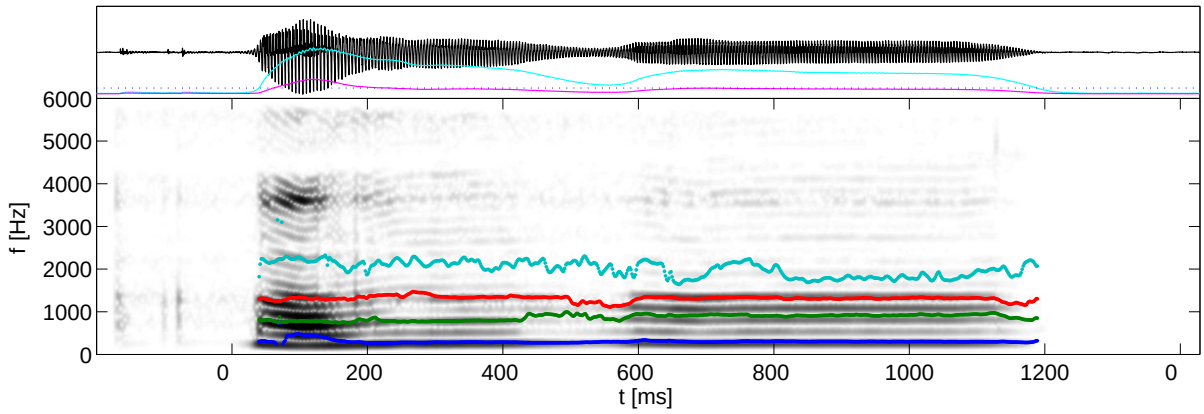
In the literature we found several formant tracking algorithms. It seems that many formant tracking algorithms have been developed over the last past years. We realized that formant tracking was a very complicated task, and is still an active field of research. Our primary goal was to implement a formant tracking algorithm with low computational complexity which at the same time satisfies strong real-time requirements. However, most algorithms we found in the literature were purely offline algorithms, utilizing temporal context information, based on phonetic or semantic rules for instance. Nevertheless we found a tracking algorithm that we adapted to fit our needs.

4.8.5 Original formant tracking algorithm

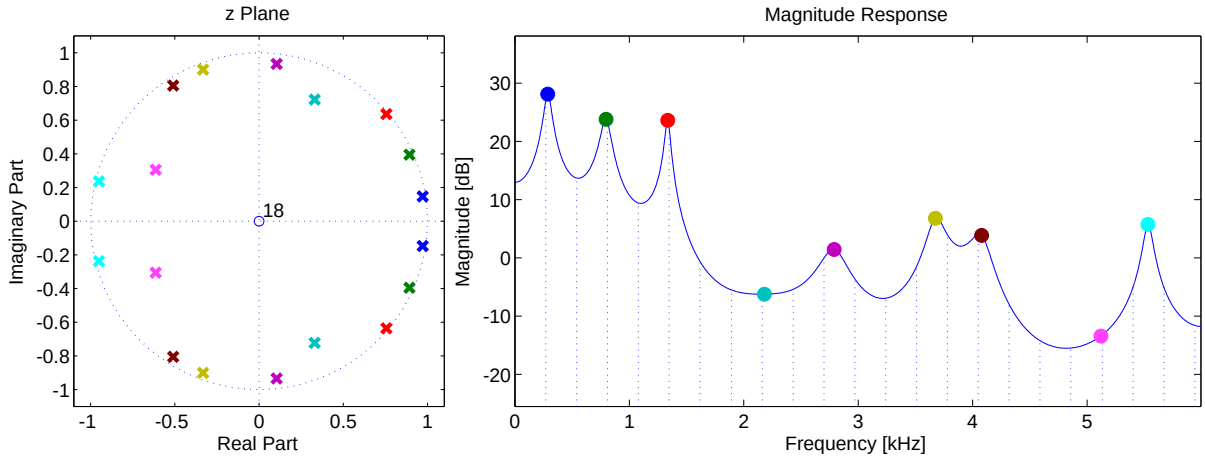
The formant tracking algorithm described in [13] is based on dynamic programming (DP). The basic idea behind it is to define a cost for each possible formant based on speech specific characteristics. At each frame a cost is calculated for every single formant candidate. We call a formant candidate each of one of the complex conjugated pole pairs provided by the LPC analysis. In addition to this, a frame-to-frame transition cost relying on continuity constraints of formant tracks is applied to penalize large frequency flaws. The optimal path trough a trellis of candidate frequencies minimizes the overall accumulated cost throughout all possible paths. This optimal path is traced back from the end to the start of the recorded speech signal. Figure 4.13 shows a typical node-transition representation of this algorithm, commonly known as Viterbi algorithm.
The DP cost function is defined as:

$$C(t,n) = C_{local}(t,n) + \min\left\{C_{tran}\left((t,n),(t-1,m)\right) + C(t-1,m)\right\} \tag{4.43}$$

Where $C(t,n)$ is the cumulative cost at node (t,n). $C_{local}(t,n)$ is the local cost at (t,n), which reflects



(a) Audio signal (upper panel) , spectrogram and formant tracks (lower panel)



(b) Extract from (a) at t=220 ms : Poles in the z plane (left) and magnitude response (right)

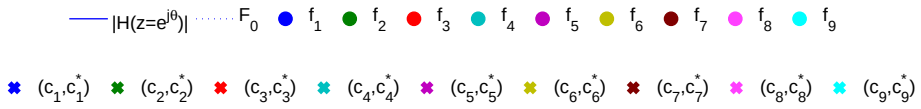


Figure 4.12: LPC estimation of the vowel [a], with LPC order $p = 18$, for a female speaker

We can see the source's periodicity appear as horizontal stripes in the spectrogram in (a), which are spaced every $k \cdot F_0$. We see that the formant tracks are very similar to the structure of the stripes on the spectrogram. On the left panel of (b) we see that the poles of the first formants are close to the unit circle and thus the corresponding magnitude response has pointed peaks at these frequencies. The spectral shape of the source has been added to illustrate that the first three formants spuriously reflect the spectral shape of the source.

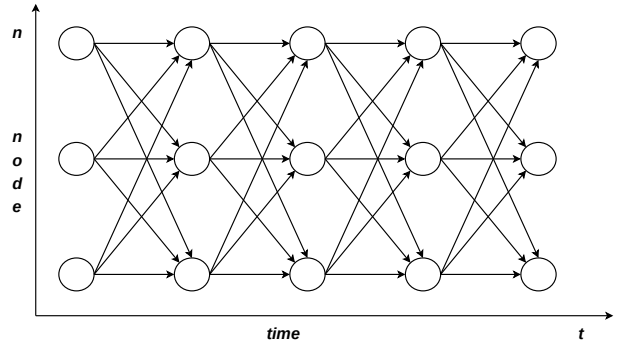


Figure 4.13: Possible paths through a trellis using DP.

information about formants without temporal context. In [13] $C_{local}(t,n)$ and $C_{tran}\left((t,n),(t-1,m)\right)$ are defined as follows:

$$C_{local}(t,n) = \sum_i \left\{\alpha_i B_i^2 + \beta_i \left|F_i - F_{n_i}\right| / F_{n_i}\right\} \tag{4.44}$$

$$C_{tran}\left((t,n),(t-1,m)\right) = \sum_i \gamma_i \left(F_i(t) - F_i(t-1)\right)^2 \tag{4.45}$$

where $\{F_i, B_i\}$ is the frequency-bandwidth pair of the i^{th} component of the mapping at node (t,n), for $i = 1, 2 \ldots N$. A $F_i(t-1)$ and $F_i(t)$ are the frequencies of the i^{th} component of the mapping at node $(t-1,n)$ and node (t,n) respectively. F_{n_i} is defined as a neutral vocal tract frequency around which the formant track should fluctuate. This is to penalize deviations from neutral formant frequencies. Furthermore, each formant is generously bounded to a specific range, for instance like this:

$$100 < F1 < 1500\,;\; 500 < F2 < 3500\,;\; 1000 < F3 < 4500\,;\ldots$$

We can see from the equations above that formant candidates with narrow bandwidths, and those with slowly-varying frequencies, will accumulate lower costs than other (erroneous) candidates and hence will be identified as the true formant tracks.

4.8.6 Modified formant tracking algorithm

We modified the algorithm described in Section 4.8.5 to a fully realtime algorithm. We utilize the same principle as the Viterbi algorithm, but as opposed to the described algorithm, we perform the Viterbi search throughout formant candidates themselves. We impose a decision at every frame and thus fully eliminate time constraints. We reach this by incorporating the transition cost directly in the node cost, which we define as follows:

$$C_{node}(t,n) = \alpha(t)B_i + \beta(t)\left|F_i(t) - F_{n_i}\right| + \gamma(t)\left|F_i(t) - \tilde{F}_i(t-1)\right| \tag{4.46}$$

with

$$\tilde{F}_i(t-1) = (1-\lambda)\hat{F}_i(t-1) + \lambda\tilde{F}_i(t-2) \tag{4.47}$$

$\hat{F}_i(t-1)$ is the estimated formant frequency extracted from the previous Viterbi search. $\tilde{F}_i(t-1)$ is the exponentially smoothed formant estimate containing information from past formant estimates with an adjustable forgetting factor λ[9] ,with $0 < \lambda < 1$. Furthermore, we introduce time-varying factors $\alpha(t)$, $\beta(t)$ and $\gamma(t)$ that define the relationship of the three cost criteria over time. We will explain their exact purpose later on.

Figure 4.13 shows a typical constellation of the node-transition network resulting from the previous considerations. We schematically added possible decisions for a particular scenario. In this example, the optimal path (green line) is indeed the original formant estimation given by the LPC. Each candidate has been confirmed to be a real formant. The algorithm did not discard any erroneous formants. In case of a wrong estimate the algorithm "jumps" this pseudo-formant and the path is pursued in the next line below, and so on.

4.8.7 Formant tracking: start conditions

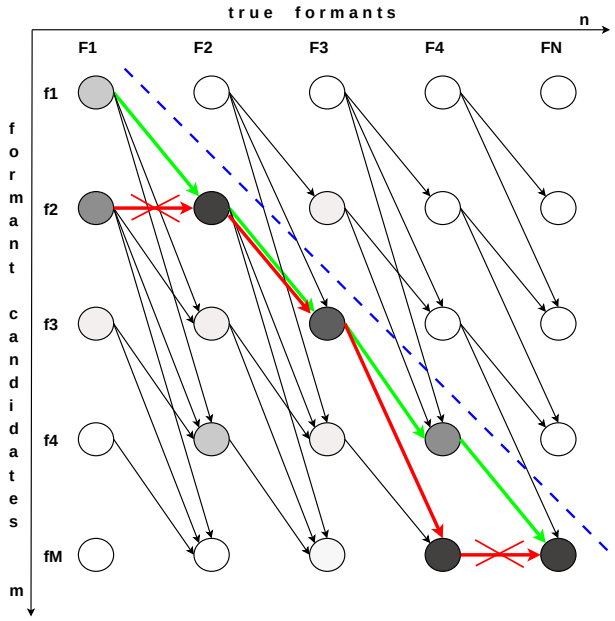
Up to now we have not described the role of the weighting factors $\alpha(t)$, $\beta(t)$ and $\gamma(t)$, that principally control the start of tracking algorithm. These factors play an important role in our algorithm and are based on the following idea: We can use the fact that we need to track formants in a very specific context, i.e. to perform our experiment. During the experiment, we know exactly which vowels subjects will produce, because they are asked to repeat words prompted on a screen that we have previously defined. These words all start with the vowel [a] and end with the vowel [i]. Thus we have a huge advantage with regard to a conventional formant tracking task, where there is absolutely no available vocalic information.

Since we know the average formants for the vowel [a] we can define them as the neutral formant frequencies F_{n_i} to start with:

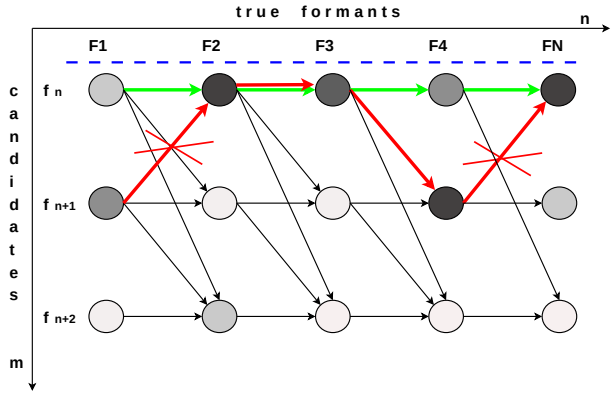
$$F_{n_1} = 700\,Hz; F_{n_2} = 1500\,Hz; F_{n_3} = 2500\,Hz; F_{n_4} = 4000\,Hz \tag{4.48}$$

We know that these formant values will be accurate at least at the beginning of the utterance, i.e. before the [a] to [i] transition starts. Therefore, we can start the tracking by defining a high value for $\beta(t)$ to penalize any large deviations from these expected formants. At the same time, we impose a very low value for $\gamma(t)$, since no past information is available to adequately measure formant discontinuities. Finally we choose a constant value for $\alpha(t)$ to penalize formants with low bandwidth, a criterion that is always valid to eliminate weak resonances. Once the tracking has started we change the value for $\beta(t)$ and $\gamma(t)$ to smoothly transition from fluctuations around constant formant frequencies $F_{n_i}$to fluctuations around exponentially smoothed formant estimations $\tilde{F}_i(t-1)$ from the past. This particular starting scheme helps to launch the tracker and to make sure that the "right" formants are tracked from the

[9] We choose a high forgetting factor so that the values of $\tilde{F}_i(t)$ only change slowly in time. This makes the tracker more robust against occasional wrong formant decisions.



(a) Original viterbi path



(b) Transformed viterbi path

Figure 4.14: Original and transformed Viterbi path through a trellis of formant candidates

In this example we represent the cost as a greyscale color, where black means a very low cost and white an infinitely high cost (i.e. this formant candidate is not in the specified formant range). The transitions themselves do not have a cost , they simply indicate possible ways through the trellis. The red path represents the node combination with the lowest accumulated cost (i.e. the combination of all the darkest nodes). However, this combination is forbidden because horizontal transitions are not allowed, otherwise one and the same formant candidate could count as two different formants, which is impossible. The green line represents the path that minimizes the accumulated cost respecting the allowed transition possibilities. Please note that all the nodes above the blue dashed line are forbidden nodes, because a formant candidate f_m can not be a true formant F_n if $m < n$. In fact a formant candidate can potentially be a formant of lower index, but never a formant of higher index. Thus, for convenience we can transform the network represented in (a) by considering only nodes below the blue line. The result of this transformation is shown in (b). The blue line has been added for clarity.

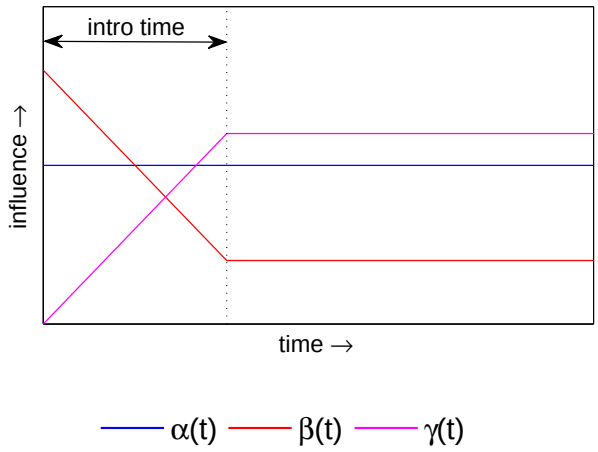


Figure 4.15: Tracking starting scheme

During the so called intro time (about 100 ms), formants are selected only based on their bandwidth (constant blue line) and on deviation form predefined expected formant values (red line). As time goes on, formant decisions are less and less influenced by the deviation from predefined formant values. Instead, deviations from the moving average past formant decision (magenta line) are taken more and more into account.

beginning.

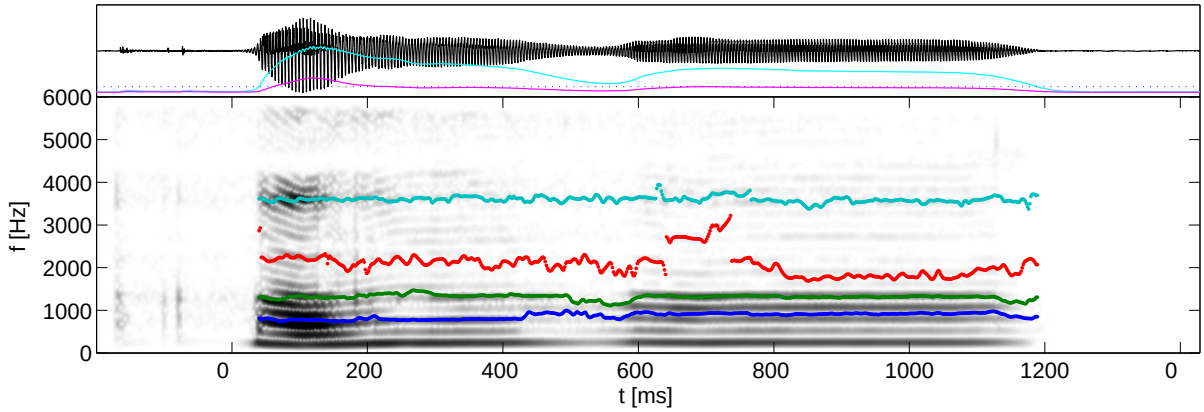
4.8.8 Results

The developed tracker provides very good results as long as the true formants are among the formant candidates provided by the LPC analysis (i.e. the sorted complex conjugated poles of the transfer function). It is capable of distinguishing right from wrong formants and of accurately tracking them in real time. However, this formant tracking algorithm does not work when the signal is very noisy, i.e. when none of the formant candidates provided by the LPC is a true formant. The reason for this is that this tracking algorithm does not influence or change the value of provided formant candidates; it simply chooses the combination of those formants that will minimize the overall cost based on bandwidth, continuity constraints and deviation from predefined formant frequencies.

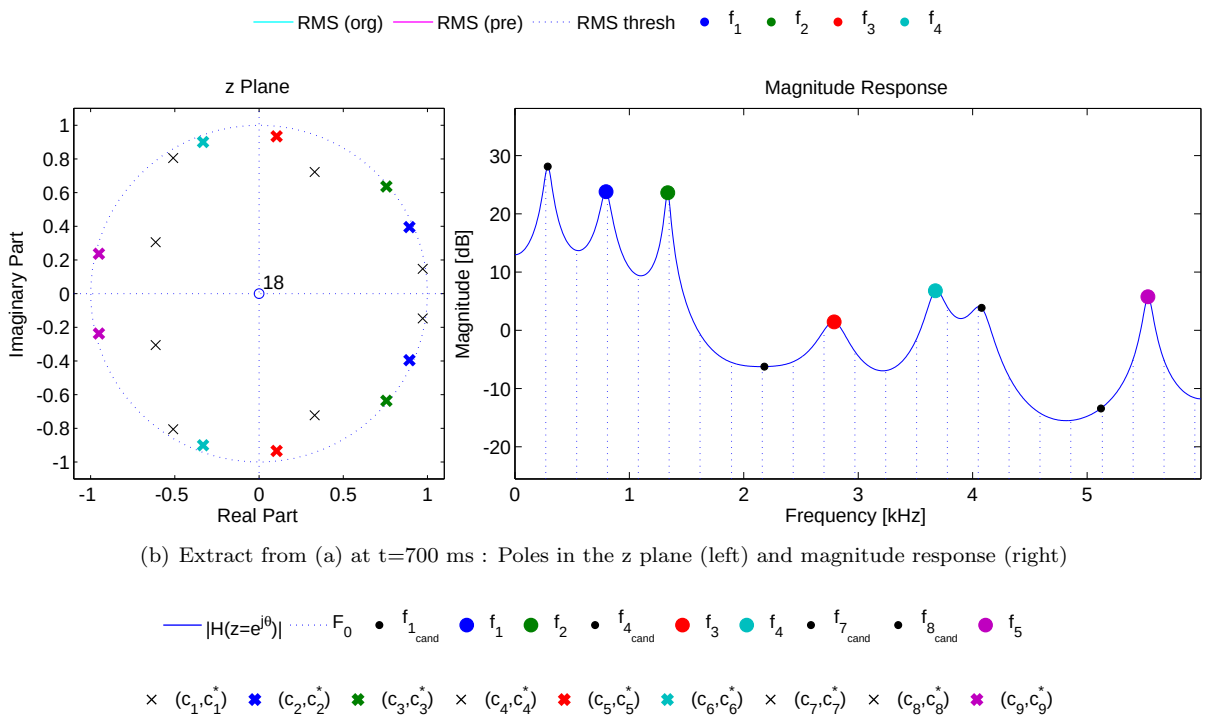
4.9 Short-Time RMS

We calculate the short-time RMS of each window to be filtered, thus over *winlen* samples, which in our case corresponds to windows of $t_{RMS_{short}} = 1,33\,ms$. The short-time RMS is calculated in the same way as was the long-time RMS (Section 4.4) except that the window is smaller, and that no forgetting factor is used for smoothing.

$$RMS_{short}(k) = \sqrt{\frac{1}{winlen} \sum_{i=0}^{winlen-1} s^2(i)_{org}} \tag{4.49}$$



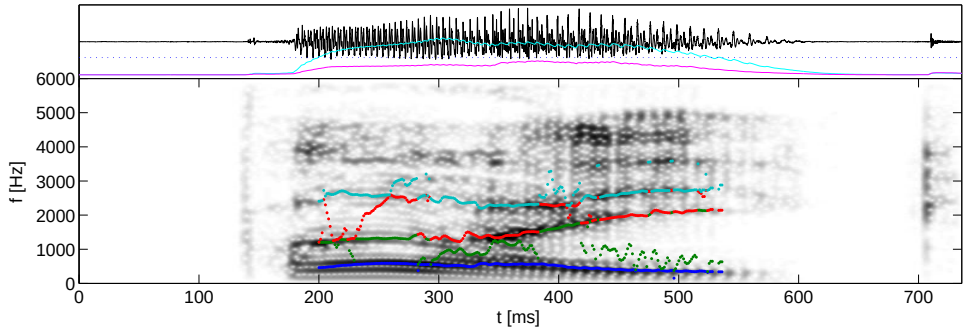
(a) Audio signal (upper panel) , spectrogram and formant tracks (lower panel)



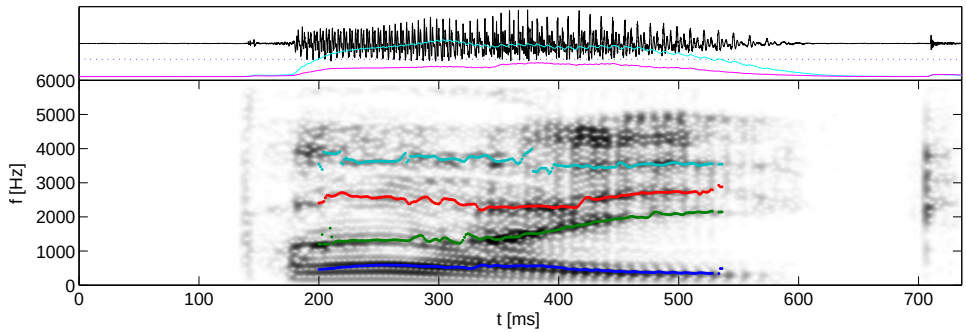
(b) Extract from (a) at t=700 ms : Poles in the z plane (left) and magnitude response (right)

Figure 4.16: Tracked formants of the vowel [a], with LPC order $p = 18$, for a female speaker

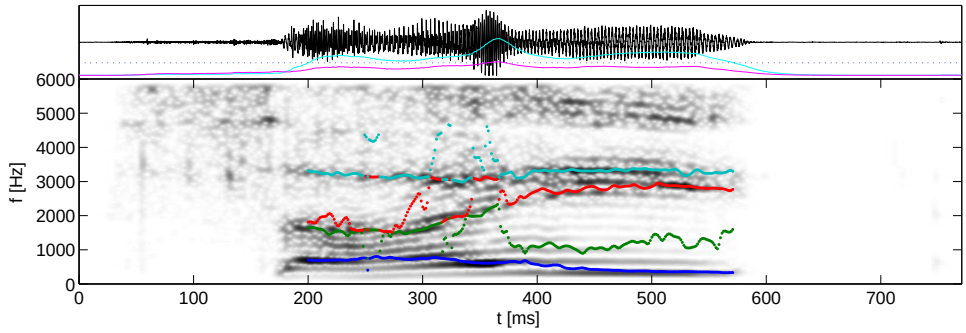
This example shows the impact of the tracking algorithm, applied to the spoken utterance represented in Figure 4.12. We remember that the provided LPC spectrum of the vocal tract filter was poor because of the speaker's high fundamental frequency. We can compare the formant tracks with and without tracking algorithm and we see that there was an amelioration : The first formant candidate has been recognized as a "false" formant, even though it has a notable peak. The fourth formant candidate has been discarded because of its high bandwidth, etc... However, even though the tracking algorithm found the optimal path throughout all possible combinations of formant candidates, the resulting formant tracks are still not very accurate, because the estimated vocal tract filter response is falsified due to the speaker's high fundamental frequency. Unfortunately, nothing can be done about that.



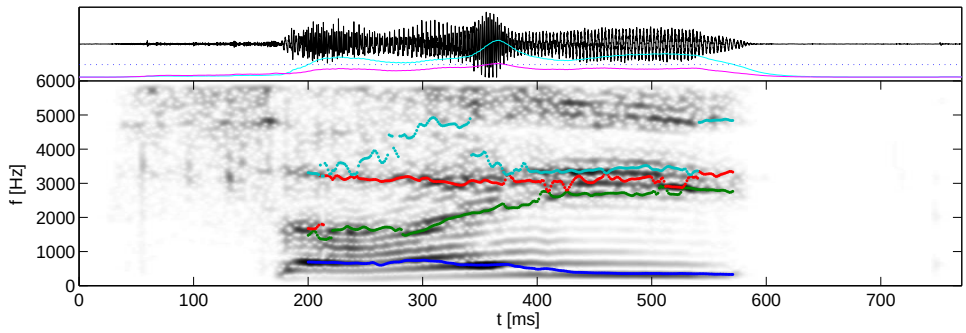
(a) Male speaker, without formant tracking



(b) Male speaker, with formant tracking

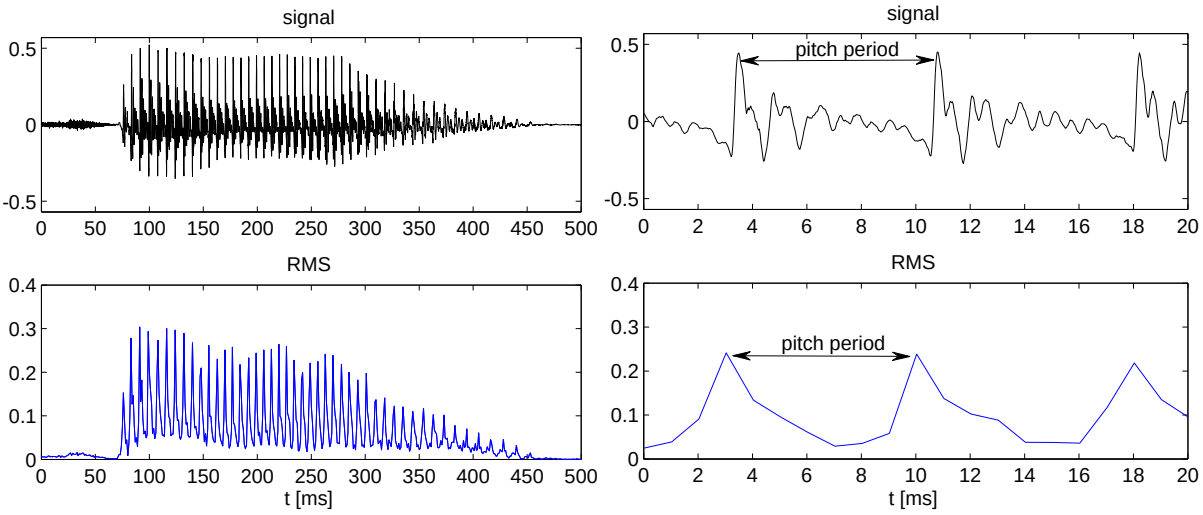


(c) Female speaker, without formant tracking



(d) Female speaker, with formant tracking

Figure 4.17: Formant tracking examples for [a] to [i] vowel transitions spoken by male and female speakers



(a) original (b) zoom

Figure 4.18: Audio signal and short time RMS

The upper panel of (a) shows a recorded speech signal of the vowel [a] for a male speaker. Below we see the course of the short time RMS. The right side of the Figure shows an extract of that speech signal. We can see that the RMS reflects the periodicity of the speech signal. The pitch period here is about 7 *ms.*

k is the index of the processed frame of *winlen* samples where $s(i)$ is the i^{th} sample of that frame.

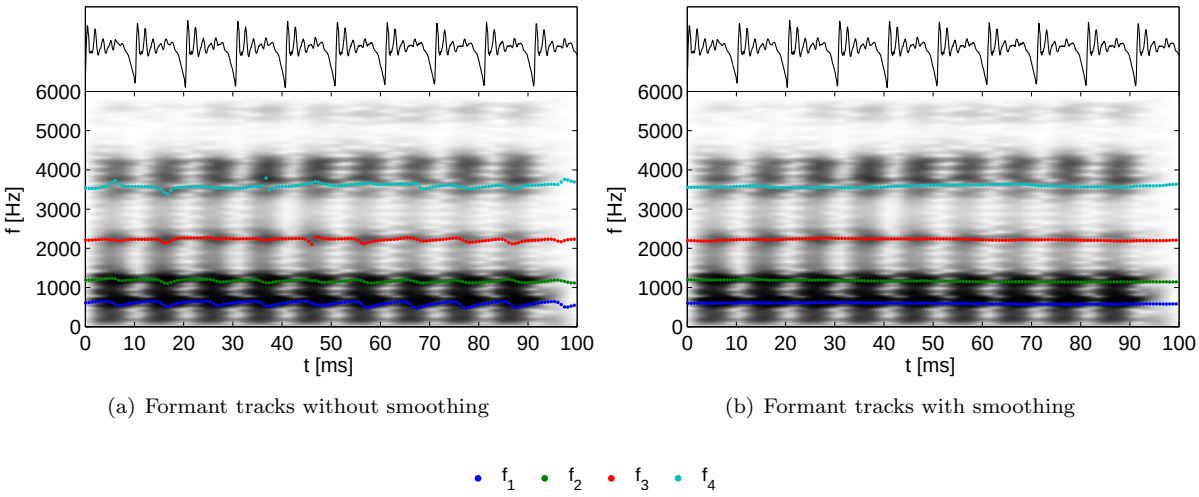
The upper left panel of Figure 4.18 shows a recorded signal of the diphthong [:i] as contained in the word "bike" and the regime of the short time RMS in the lower left panel. The right panel of Figure 4.18 shows an extract of the signal and the short time RMS as described in the previous section.

We can see that the course of the RMS is similar to the original signal. The time interval between two peaks indicate the fundamental pitch period of the speaker. In the next Section, we will describe how we are using the short time RMS to improve the formant estimation.

4.10 Formant smoothing

Due to strong real-time requirements it is necessary to minimize the LPC analysis window size as much as possible. Usually, a window size of approximately 2 pitch periods is utilized by conventional formant estimation algorithms. Minimizing the LPC window size affects the accuracy of the spectral estimation. As the window gets smaller, the estimated formants start oscillating with the pitch period. We do not wish to describe why this is happening. However, we know that the most accurate point in time to estimate the formants of the vocal tract is at the closing phase[10] of the glottis. Some formant estimation algorithms estimate the formants synchronously to the pitch period of the speech signal. The estimation is then performed only during the closing phase of the glottis. However, these algorithms are rather complicated and time consuming to implement. Thus we decided to implement an easy-to-implement formant smoothing routine based on these considerations.

[10] Better reflects the vocal tract filter because of no sub glottal resonances.



(a) Formant tracks without smoothing

(b) Formant tracks with smoothing

Figure 4.19: Formant tracks with and without smoothing

In the left panel we see how formant tracks oscillate with the pitch period. The fluctuations are considerable and will affect the filtering. The right panel shows the formant tracks after smoothing. The smoothing is done over approximately one pitch period. The short time RMS of the speech signal serves as weighting factor for the weighted moving average process to reduce the oscillations.

The closing phase of the glottis is characterized by a peak in the speech signal. The short-time RMS will also have a peak at that moment as we can see in the right panel of Figure 4.19. Our idea is simply to take the value of the short-time RMS as a weighting factor to perform a weighted moving average of the formants over approximately one pitch period. Thus, formant estimations during the closing phase of the glottis will be taken much more into account, and hence the overall estimation will be more accurate. The positive effect of doing so is that we also reduce the influence of potential false estimations. This is because the probability of a "wrong" LPC estimation increases when the signal energy is low. When such a "wrong" estimation occurs, the RMS will be low and hence will not have a big effect.

We perform the smoothing over approximately one pitch period which we determine for each subject before starting the experiment. Let's denote N_{pitch} the number of processed frames corresponding to one pitch period. For each formant f_i at a given frame k we calculate the weighted moving average as follows:

$$f_{i_{wma}}(k) = \frac{\sum_{n=0}^{N_{pitch}-1} RMS_{short}(k-n) \cdot f_i(k-n)}{\sum_{n=0}^{N_{pitch}-1} RMS_{short}(k-n)} \tag{4.50}$$

We implemented a modified version of Equation 4.50 to lower the computational cost. We use the fact that only one new element adds to the sum at each frame, and one disappears. Thus we can efficiently calculate the moving average by adding and subtracting values to the previous moving average formant value that we update at every frame:

$$f_{i_{wma}}(k) = f_{i_{wma}}(k-1) + WMA_{new}(k) - WMA_{old}(k) \tag{4.51}$$

with

$$WMA_{new}(k) = \frac{RMS_{short}(k) \cdot f_i(k)}{RMS_{short}(k)} \tag{4.52}$$

and

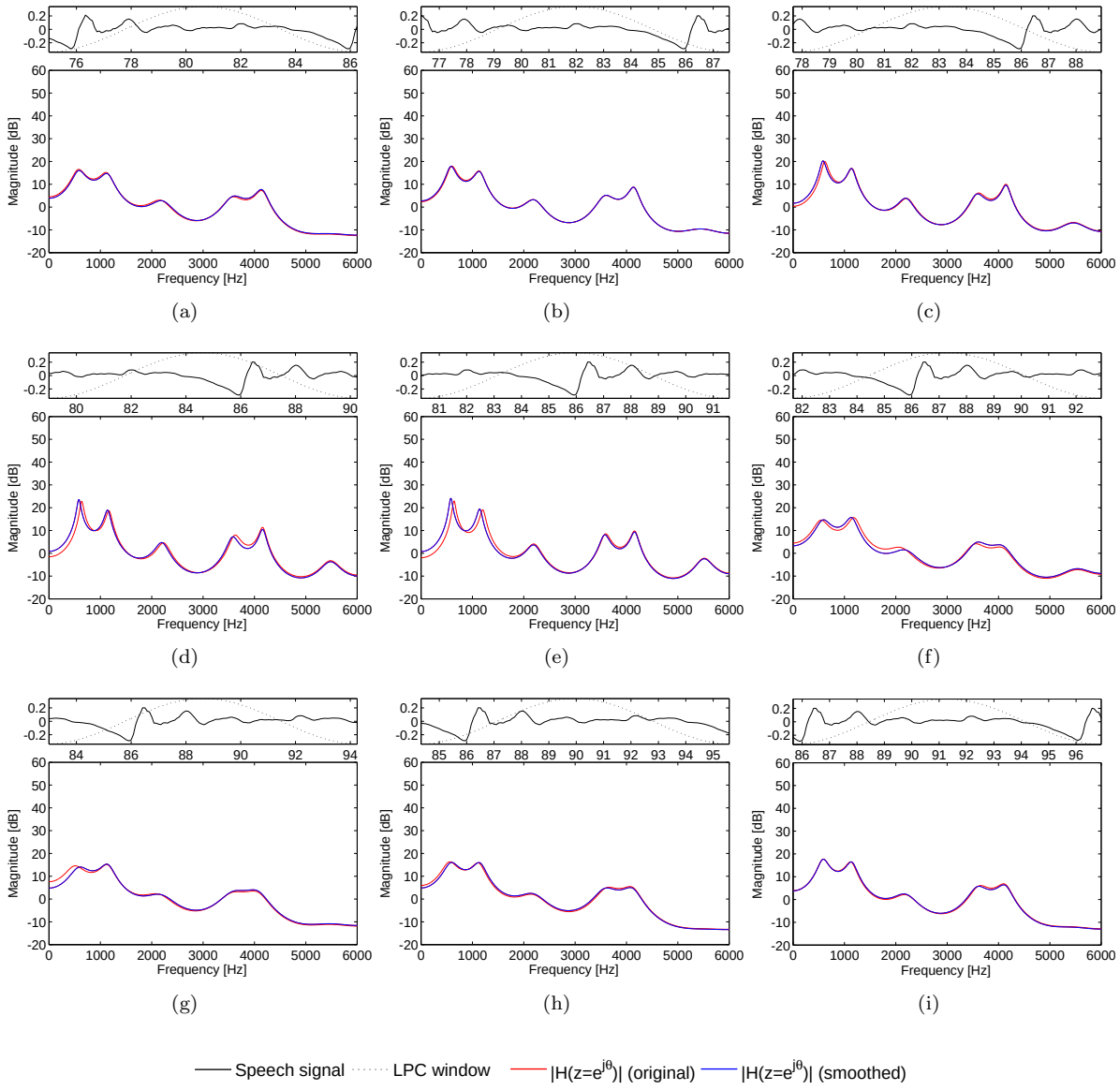


Figure 4.20: Fluctuation of formants due to small LPC analysis window

This Figure shows an extract of Figure 4.19 for approximately one pitch period. The time sequence starts at 81 *ms (a) and ends at* 91 *ms (i). The upper panel of each subplot shows the frame of the speech signal (black line) that is used for the LPC analysis. The frame is windowed with a hanning window (black dotted line) before analysis. In the lower panel of each subplot we see the estimated LPC magnitude response of the represented frame. The blue line represents the "formant smoothed" magnitude response, the red line the original magnitude response. We see that the original filter response fluctuates around the smoothed response.*

$$WMA_{old}(k) = \frac{RMS_{short}(k - N_{pitch} + 1) \cdot f_i(k - N_{pitch} + 1)}{RMS_{short}(k - N_{pitch} + 1)} \tag{4.53}$$

4.11 Formant deviation

Until now, all the blocks we have described were more or less focusing on formant estimation and tracking. We actually need the formants for two reasons:

- On the one hand, they are necessary to perform the formant shift, properly speaking. In fact, we need the poles of the vocal tract transfer function to compensate them with zeros in the filter transfer function. We then add new poles — with adjusted angles with regard to the original poles — in order to attain the desired new formant frequency.
- On the other hand, they are required because the new formant trajectory is relative to the original trajectory. In fact, we wish to deviate subject's actual [a] to [i] formant transition, by bowing it in one direction in the acoustic space (See Section 1.2.2 on page 3). The way the trajectory is shifted must satisfy the criteria described in the following section.

4.11.1 Requirements

First of all, we wish to leave the transition end points unchanged. Thus, the deviation function must be equal to zero at both ends. This eliminates all possible deviation functions which only asymptotically converge to zero at the ends. Secondly, we wish the deviation function to be symmetric to the axis through the midpoint of the transition. Thirdly, the amount of perturbation should be lengthwise relative to the original trajectory. Finally, the perturbation should be independent of an eventual compensation by the subjects.

To satisfy these requirements we developed a deviation function that can be seen as a vector field that is perpendicular to the original trajectory. To calculate that field we need to deduce each subject's average transition start and endpoints prior to the actual experiment. We will determine these points during the first phase of the experiment, where no perturbation is applied.

4.11.2 Deviation vector field

Let's assume we know the subject's average transition start and endpoints, which we call $f_{i_{start}}$ and $f_{i_{stop}}$.

We can extract the angle α of that transition with regard to the f_1 axes.

$$\alpha = \arctan\left(\frac{f_{2_{stop}} - f_{2_{start}}}{f_{1_{start}} - f_{1_{stop}}}\right) \tag{4.54}$$

We can then build a rotation matrix A_{rot}.

$$A_{rot} = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{pmatrix} \tag{4.55}$$

This rotation matrix allows us to represent the trajectory in a transformed space, where the new axes are rotated with regard to the axes of the acoustic space. Every 2 dimensional point in the acoustic space $\overrightarrow{f_{org}}$ will be transformed as follows:

$$\overrightarrow{f_{rot}} = A_{rot} \cdot \overrightarrow{f_{org}} \tag{4.56}$$

In this transformed space, the transition is now horizontal as we can see in Figure 4.21. It is now very easy to generate a deviation that will bow the trajectory into one direction. We can simply apply any symmetric function that has a maximum in its midpoint and is equal to zero at the start and end points (i.e. at the rotated transition start and end points). We chose to use a hanning function because its derivative is equal to zero at the endpoints, and hence the deviation will not start or end abruptly. Now we can very easily generate a deviation vector that will translate every point by a value dev_{rot} in the y direction of the rotated space. It is defined as follows and only depends on the value of $f_{1_{rot}}$,i.e. the x coordinate in the rotated space:

$$dev_{rot}(f_{1_{rot}}) = \begin{cases} dev_{max} \cdot \left(\frac{1}{2} - \frac{1}{2}\cos\left(\frac{2\pi f_{1_{rot}}}{f_{1_{rot_{stop}}} - f_{1_{rot_{start}}}}\right)\right) & , \; f_{1_{rot_{start}}} < f_{1_{rot}} < f_{1_{rot_{stop}}} \\ 0 & , else \end{cases} \tag{4.57}$$

We can see that only points somewhere in between the rotated goal regions will be shifted, with a deviation that will climax in the middle of the transition, and only depending on the x coordinate of the rotated point. The maximal deviation value can be adjusted by changing the parameter dev_{max}which represents the maximal euclidean deviation distance in Hz of the acoustic space.

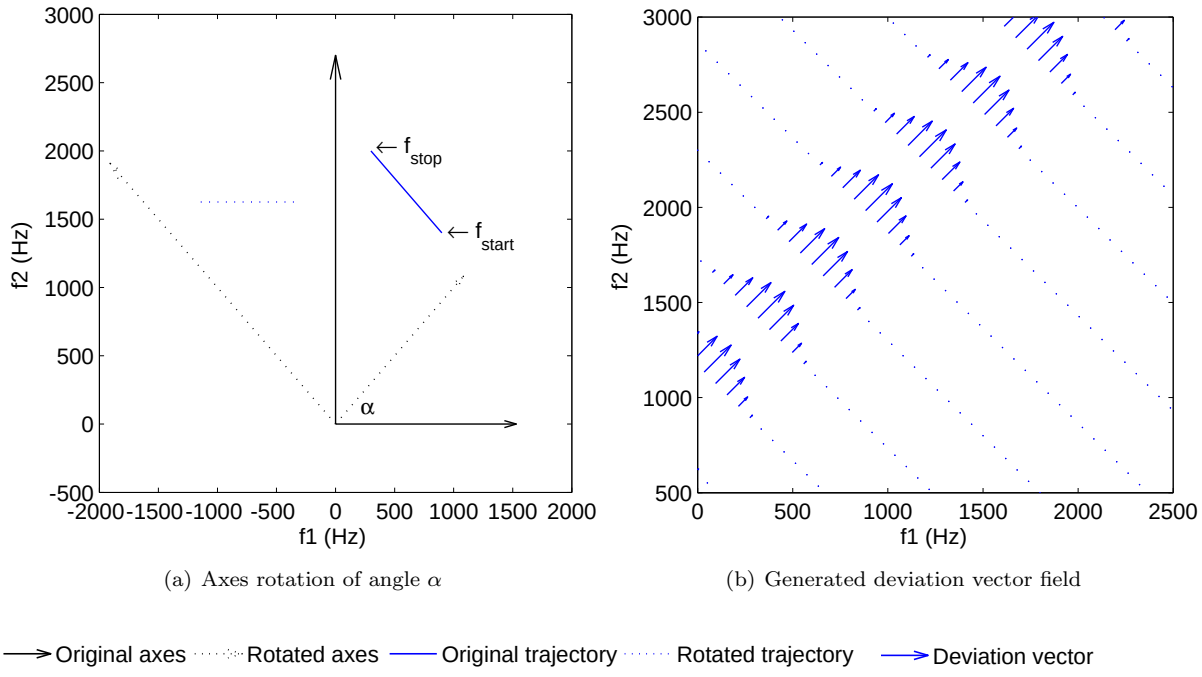
We can now determine the new shifted formants in the acoustic space by simply rotating the axes back:

$$\begin{pmatrix} f_{1_{shift}} \\ f_{2_{shift}} \end{pmatrix} = \begin{pmatrix} f_{1_{rot}} \\ f_{2_{rot}} + dev_{rot} \end{pmatrix} \cdot A^{-1} \tag{4.58}$$

Where $A^{-1}$is the inverse of A.

The advantage of transforming the axes is that we can easily apply a deviation that is perpendicular to the transition (once rotated) by simply changing its y coordinate. We can use any arbitrary function to define the deviation, and thus avoid complicated parametric two-dimensional functions. Furthermore, by only considering the x coordinate of the rotated space, we generate a field that is constant along any line perpendicular to the transition. This is very important as we need to make sure that, if subjects compensate for the perturbation, the applied deviation will not change. This is guaranteed, as we expect subjects to compensate in the opposite direction to the deviation, and thus perpendicular to the trajectory (see Section 1.1).

We can see the generated deviation field in the right panel of Figure 4.21.



(a) Axes rotation of angle α

(b) Generated deviation vector field

Figure 4.21: Axes transformation and vector field generation

In (a) we can see an arbitrary [a] to [i] trajectory (blue line). This trajectory is represented in the acoustic space, i.e. within the black axes. Based on the start and end points we rotate the trajectory (blue dotted line) so that it is horizontal in the transformed space. We then apply a vertical translation and rotate the deviated point back to the original acoustic space and thereby obtain a deviation field as it is represented in (b).

4.12 Transition detection

In order to determine the start and end points of the formant trajectory we have implemented a simple [a] to [i] transition detection. It is principally based on the derivative in time of the first two formants. We know that, during an [a] [i] transition the first formant $f_1$will drop and the second formant f_2 will rise. As soon as both f_1 drops and f_2 rise, we can proceed from the assumption that the transition has started.

The transition detection is comprised of 3 detection stages:

4.12.1 Stage one

In order to prevent wrong detections, we impose the formants to be in a certain range. Of course, we define that range around average formant values for the vowel [a] as this is the start point of our transition we want to detect. Thus the first condition is:

$$f_{i_{def}} - f_{i_{tol}} < f_i < f_{i_{def}} + f_{i_{tol}}, i = 1, 2 \tag{4.59}$$

Where $f_{i_{def}}$is a predefined formant, in our case the expected formants for the vowel [a], and $f_{i_{tol}}$is a tolerance around these expected values.

4.12.2 Stage two

In this stage we compare the derivatives in time of $f_1$and f_2. We formulate the second condition as follows:

$$\frac{\delta f_1}{\Delta t} < \frac{\delta f_2}{\Delta t} \tag{4.60}$$

Where the derivatives are calculated in $[Hz]/[ms]$.

4.12.3 Stage three

Stage three can adopt three states:

- "$noTransition$"
- "$duringTransition$"
- "$wasTransition$"

As a default, stage three is set to "$noTransition$" . In this state, it collects the results from stage one and stage two and, if both stages detect a transition at least a few times in a row, the state is set to "$duringTransition$". The first collected formants in that state are saved in a variable $startFmts$.

Once the internal state is set to "*duringTransition*", stage one is disabled, and only decisions from stage two are taken into account. The state is maintained to "*duringTransition*" as long as stage two detects a transition. As soon as stage two does not detect a transition, stage three checks if the transition was long enough. If so, the last formants are saved in a variable *stopFmts*, and the internal state is set to "*wasTransition*", which blocks the transition detection. If the detected transition was not long enough, stage three returns to the default state "*noTransition*".

At the end of each recording, the variables *startFmts* and *stopFmts* are collected and utilized to determine the subject's average trajectory. In addition to this, whenever the internal state is set to "*duringTransition*", the transition detection enables the filter process, which is described in the next section.

4.13 Filtering

In order to shift the formant to the desired frequency we utilize a digital filter in the time-domain. We shift the formant frequencies by compensating the original poles with zeros and by adding new poles that will be shifted in frequency with regard to the original poles. A simple digital filter with two complex conjugated pairs of zeros and poles will be sufficient to perturb the first two formants f_1 and f_2. The system function of such a filter is

$$H(z) = \frac{\prod_{k=1}^{2}(1 - c_k z^{-1})(1 - c_k^{\star} z^{-1})}{\prod_{k=1}^{2}(1 - \hat{c}_k z^{-1})(1 - \hat{c}_k^{\star} z^{-1})} \tag{4.61}$$

where $\{c_k\}$ are the original poles and $\{\hat{c}_k\}$ the shifted poles. Equation 4.61 can be transformed to

$$H(z) = \frac{\prod_{k=1}^{2}(1 - 2r_k \cos(\theta_k) z^{-1} + r_k^2 z^{-2})}{\prod_{k=1}^{2}(1 - 2r_k \cos(\hat{\theta}_k) z^{-1} + r_k^2 z^{-2})} \tag{4.62}$$

where $f_k = \frac{\theta_k \cdot F_{s\downarrow M}}{2\pi}$, is the original formant frequency and $\hat{f}_k = \frac{\hat{\theta}_k \cdot F_{s\downarrow M}}{2\pi}$ the shifted formant frequency, which we obtain using the deviation functions described in Section 4.11. The new vocal tract system function $\hat{V}(z)$ is then given by Equation 4.65

$$\hat{V}(z) = V(z) \cdot \frac{\prod_{k=1}^{2}(1 - c_k z^{-1})(1 - c_k^{\star} z^{-1})}{\prod_{k=1}^{2}(1 - \hat{c}_k z^{-1})(1 - \hat{c}_k^{\star} z^{-1})} \tag{4.63}$$

We recall that the vocal tract system function can be separated in two parts:

$$V(z) = V_c(z) \cdot V_r(z) \tag{4.64}$$

Where $V_c(z)$ contains only complex conjugated pole pairs $\{c_k, c_k^{\star}\}$,

$$V_c(z) = \frac{1}{\prod_{k=1}^{M}(1 - c_k z^{-1})(1 - c_k^{\star} z^{-1})} \tag{4.65}$$

and $V_r(z)$ only real poles c_i.

$$V_r(z) = \frac{1}{\prod_{i=1}^{N}(1 - c_i z^{-1})} \tag{4.66}$$

N and M are related to the LPC order P by $P = 2M + N$.

Thus, Equation 4.63 can be written as follows:

$$\hat{V}(z) = \frac{1}{\prod_{i=1}^{N}(1 - c_i z^{-1})} \cdot \frac{1}{\prod_{k=1}^{M}(1 - c_k z^{-1})(1 - c_k^{\star} z^{-1})} \cdot \frac{\prod_{k=1}^{2}(1 - c_k z^{-1})(1 - c_k^{\star} z^{-1})}{\prod_{k=1}^{2}(1 - \hat{c}_k z^{-1})(1 - \hat{c}_k^{\star} z^{-1})} \tag{4.67}$$

This finally leads to Equation 4.68 by removing the compensated poles and zeros.

$$\hat{V}(z) = \frac{1}{\prod_{i=1}^{N}(1 - c_i z^{-1}) \cdot \prod_{k=1}^{2}(1 - \hat{c}_k z^{-1})(1 - \hat{c}_k^{\star} z^{-1}) \cdot \prod_{k=3}^{M}(1 - c_k z^{-1})(1 - c_k^{\star} z^{-1})} \tag{4.68}$$

We see that, the new vocal tract system function only contains poles, with two complex conjugated poles than have been altered with regard to the original poles.

4.14 Gain adaptation

In the previous section we described how we can easily shift a formant by simply manipulating the angle of the poles of the transfer function. Thereby we only changed formant frequencies, i.e. we did neither modify the radius r_k of each pole nor did we change the spectrum's overall gain. Unfortunately, a formant shift does not only affect the peak's frequency, it does also change its gain. This is because each pole influences the poles in its direct neighborhood. This is particularly true for the first pole c_k, because it is very close to its complex conjugated pole $c_k^{\star}$. Thus it is necessary to compensate for this undesired phenomenon.

In fact, this was a difficult task, since we wanted to make the formant shift sound as natural as possible. We have studied various ways to achieve this. The methods we have developed are all listed in Appendix A. We finally decided to use one method that relies on the physical properties of the vocal tract.

In [1] the vocal tract is modeled as a 4-tube resonator. Based on this model, the spectral contribution of each formant is defined as follows:

$$|H_k(f)| = |H_k(s = j\omega)| = \frac{s_k s_k^{\star}}{|(s - s_k)|\,|(s - s_k^{\star})|} \tag{4.69}$$

Where $\{s_n, s_n^{\star}\}$ are the complex conjugated pole pairs of each formant in the s-plane. Transforming Equation 4.69 to the z plane yields Equation 4.70

$$|H_n(f)| = \left|H_n(z = e^{j\omega})\right| = \frac{(1 - c_k)(1 - c_k^{\star})}{|(1 - c_k z^{-1})|\,|(1 - c_k^{\star} z^{-1})|} \tag{4.70}$$

We can easily see that the spectral contribution of each formant defined by Equation 4.70 is $0dB$ for $f = 0Hz$, because numerator and denominator are equal.

In our case, we don't have $0dB$ at $f = 0Hz$ because the system function provided by the LPC analysis also contains a gain factor G. This gain factor is a "leftover" from the source. However, we know that the spectral contribution of the vocal tract is $0dB$ at $f = 0Hz$. Whenever we shift a formant, we must ensure that this contribution stays at $0dB$ for $f = 0Hz$. This means that the overall gain of the vocal tract filter at $f = 0Hz$, i.e. $|T(z = 1)|$ must stay constant. Let's call $\hat{V}_n(z)$ the shifted and gain adapted transfer function. Furthermore we call $G(\theta = 0)$ the gain of the original, and $\hat{G}(\theta = 0)$ the gain of the formant shifted vocal tract magnitude response at $f = 0Hz$.

$$\hat{V}_n(z) \quad = \quad \frac{G(\theta = 0)}{\hat{G}(\theta = 0)} \cdot \hat{V}(z) \tag{4.71}$$

with

$$\frac{G(\theta = 0)}{\hat{G}(\theta = 0)} = \frac{|(1 - c_i)(1 - c_i^\star)|}{|(1 - \hat{c}_i)(1 - \hat{c}_i^\star)|} \underbrace{\frac{\prod_{k \neq i}^{M} |(1 - c_k)(1 - c_k^\star)|}{\prod_{k \neq i}^{M} |(1 - c_k)(1 - c_k^\star)|}}_{=1} \tag{4.72}$$

This leads to the simple relation

$$\frac{G(\theta = 0)}{\hat{G}(\theta = 0)} = \frac{|(1 - c_i)(1 - c_i^\star)|}{|(1 - \hat{c}_i)(1 - \hat{c}_i^\star)|} \tag{4.73}$$

and finally to

$$\frac{G(\theta = 0)}{\hat{G}(\theta = 0)} = \frac{1 - 2r_i \cos(\theta_i) + r_i^2}{1 - 2\hat{r}_i \cos(\hat{\theta}_i) + \hat{r}_i^2} \tag{4.74}$$

Where $\frac{G(\theta=0)}{\hat{G}(\theta=0)}$ is the gain factor that we need to apply to the signal in order to achieve the desired gain adaptation. A simple way of applying this gain adaptation factor to the signal would be to incorporate it directly into the filter function that performs the formant shift. However, this will introduce noise since this gain factor is not updated at each sample. Large fluctuations in the gain factor will imply an abrupt "jump" in the signal every time the factor is updated. This is why we chose to apply the gain factor directly to the signal, but only update the gain factor at zero crossings in the signal. This will minimize the large "jumps" and thus considerably reduce noise.

4.15 De-emphasis

Before the signal can be sent back to the sound card it needs to be de-emphasized. This is achieved by filtering the signal by the inverse of the preemphasis filter as introduced in Section 4.2. The transfer function of such a filter is :

$$R^{-1}(z) = \frac{1}{1 - \mu z^{-1}} \tag{4.75}$$

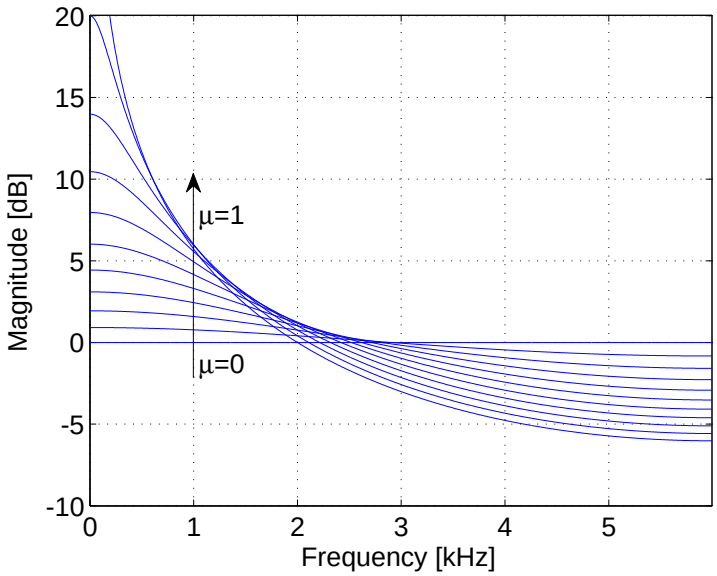


Figure 4.22: Deemphasis filter

Figure 4.22 shows the frequency response of $R^{-1}(z)$ for μ varying between 0 and 1 .

4.16 Upsampling

Now that all the processing is done, we simply need to upsample the signal by factor M to match with the soundboard's sampling rate.

4.16.1 Interpolation

The first step consists of interpolating the signal with $M-1$ zeros:

$$s(n) = \begin{cases} s_{\downarrow M}(k) & , n = Mk\,, \\ 0 & , else \end{cases} \qquad , k = 0, 1, \ldots, framelen - 1 \tag{4.76}$$

4.16.2 Filtering

To avoid spectral imaging, we need to filter the interpolated signal. We utilize the same filter that we used for downsampling the signal.

Chapter 5

The Experiment

7 female and 4 male subjects were involved in our pilot study. Subjects were repeating words like "bike" or "kite" all containing an [a] to [i] transition while their recorded speech was fed back in real-time through headphones. On two distinct sessions of 25 minutes each, subjects' [a] [i] trajectories were shifted either down or up. Each session was comprised of 4 distinct phases, with one additional training phase, to familiarize subjects with the experiment.

5.1 The 4 phases

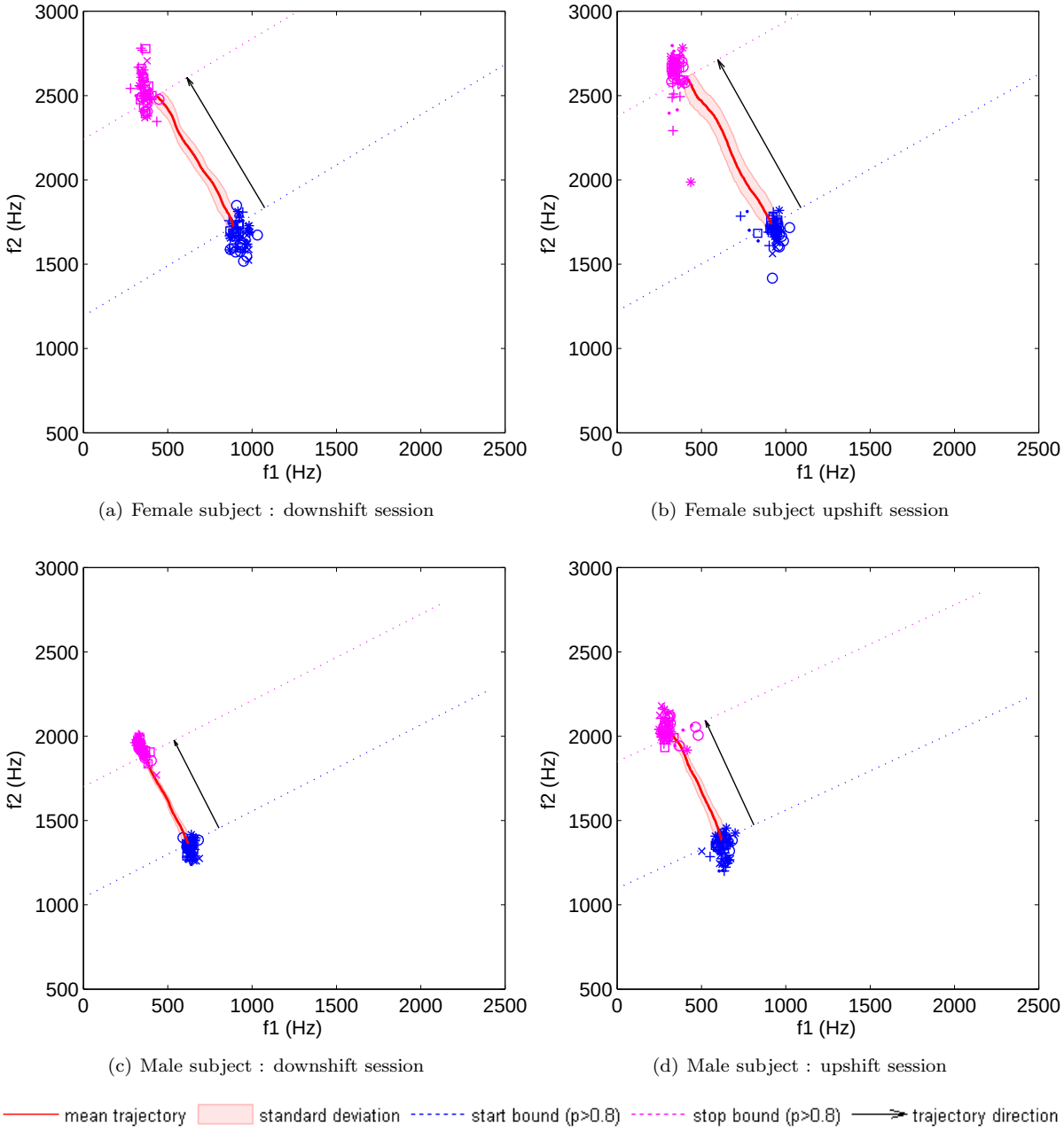
5.1.1 Start phase

During the so-called start phase (or baseline phase) subjects hear their own speech without any modifications. During this phase subject's trajectory start and endpoints are detected, as described in Section 4.12. At the end of the baseline phase, a Matlab function analyzes the collected data to determine a "baseline" trajectory, i.e. subjects average trajectory without perturbation. This baseline trajectory will later be used for comparison with subjects' shifted trajectories.

Based on the baseline trajectory, a Matlab function calculates a best linear fit approximation through all trajectories. The angle α of that linear approximation is used to create the rotation matrix for the deviation vector field as described in Section 4.11.2. We now rotate all the detected start and endpoints so that they are aligned horizontally and generate a histogram of their distribution. We then define the bounds of the deviation vector field so that a certain percentage p of the detected start and endpoints will be outside the field. We chose $p = 0.8$, which means that 80 % of start and end points will be outside. Figure 5.1 illustrates the described procedure for a male and a female subject.

5.1.2 Ramp phase

During the ramp phase, subject's trajectory is bowed towards the left for the downshift, and towards the right of the acoustic space for the up shift. This bowing grows linearly during the entire ramp phase and reaches maximal perturbation at the end. Figure 5.2 shows extracts of the ramp phase for the male



(a) Female subject : downshift session

(b) Female subject upshift session

(c) Male subject : downshift session

(d) Male subject : upshift session

Figure 5.1: Collected trajectory start and endpoints and generated deviation vector field boundaries

This Figure shows collected data from the start phase of one male and female subject. The blue cluster indicates subject's transition start points, i.e. the vowel [a], the magenta cluster indicates the end points (vowel [i]). The red line is the average trajectory of all trials. The standard deviation around this average trajectory is represented in a lighter red. As we can see, for the male subject the trajectories are very homogeneous, whereas the female subject shows more dispersion. Note that the dots outside the cluster in (b) are wrong detections. The determined field boundaries are indicated as dashed lines. The total euclidean distance between the two boundaries is represented as a black arrow, which points towards the end of the transition. As we can see, the transition length in Hz is much greater for the female subject.

subject, whose baseline phase is represented in Figure 5.1 panel (a).

5.1.3 Stay phase

During the entire stay phase, subjects trajectory is shifted at full perturbation. In our experiment the full perturbation was set to 200 Hz euclidean distance. This is the distance from the midpoint of the original, to the midpoint of the shifted trajectory in the acoustic space, i.e. where the bowing is maximal.

5.1.4 End phase

During the end phase, perturbation is turned off again. If subjects compensated for the previous perturbation, their trajectory should now return to the baseline trajectory.

5.2 Results

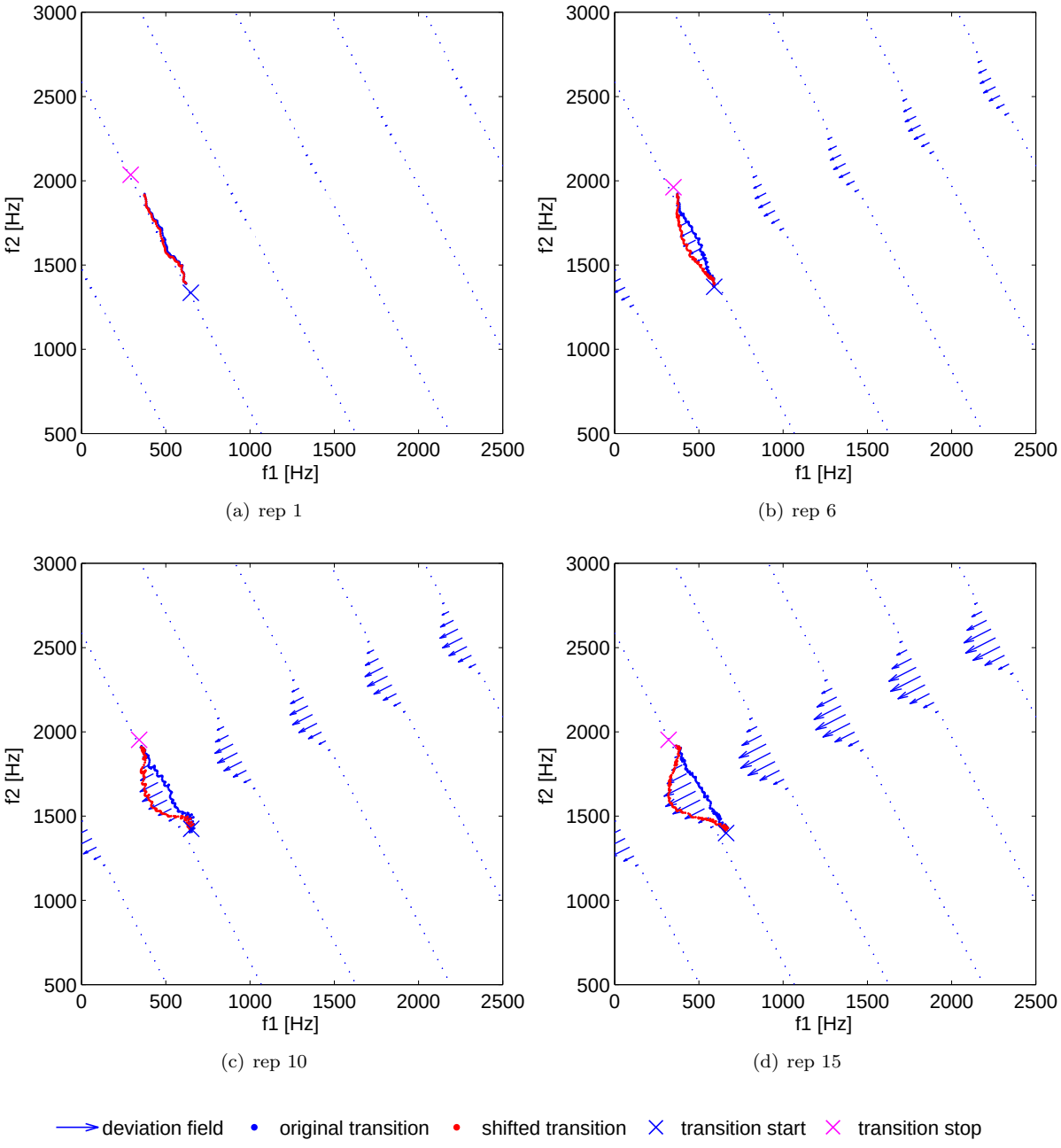
The results for the female subject are represented in Figure 5.3 and 5.4 , and in Figure 5.5 and 5.6 for the male subject. The x axis represents time: one epoch stands for one spoken utterance. Each plot represents a slice along the transition: The first plot at the bottom represents a slice at 0 %of the transition (start bound of the deviation field in Figure 5.1). The next plot above shows one slice at $12,5$ % of the transition, and so on until reaching the end point of the transition, which is represented in the upper plot. In each plot, the blue line represents the euclidean distance (in Hz) between subject's actual transition and their baseline transition. The gray surface indicates the amount of perturbation that was applied. One can observe that the perturbation grows during the ramp phase, stays maximal within the "stay" phase and falls back to 0 for the last phase. Furthermore, the bowing of the trajectory reaches its maximal value at the midpoint of the transition[1], and gets smaller towards the endpoints[2].

5.3 Conclusion and future work

During the study, only some of the 7 female and 4 male subjects showed a significant compensation. However, even for these subjects, the compensation was not necessarily as we would have expected it.

Nevertheless, since some subjects showed a compensation effect, the Speech Communication Group is planning to pursue the experiment with more subjects.

[1] See 50 % subplot
[2] See 0 % or 100 % subplot



(a) rep 1

(b) rep 6

(c) rep 10

(d) rep 15

Figure 5.2: Ramp phase: Perturbation increases linearly

Linear bowing of the trajectory during the ramp phase. The deviation field is always perpendicular to the transition, and the detected start and endpoints are in the vicinity of the field boundaries.

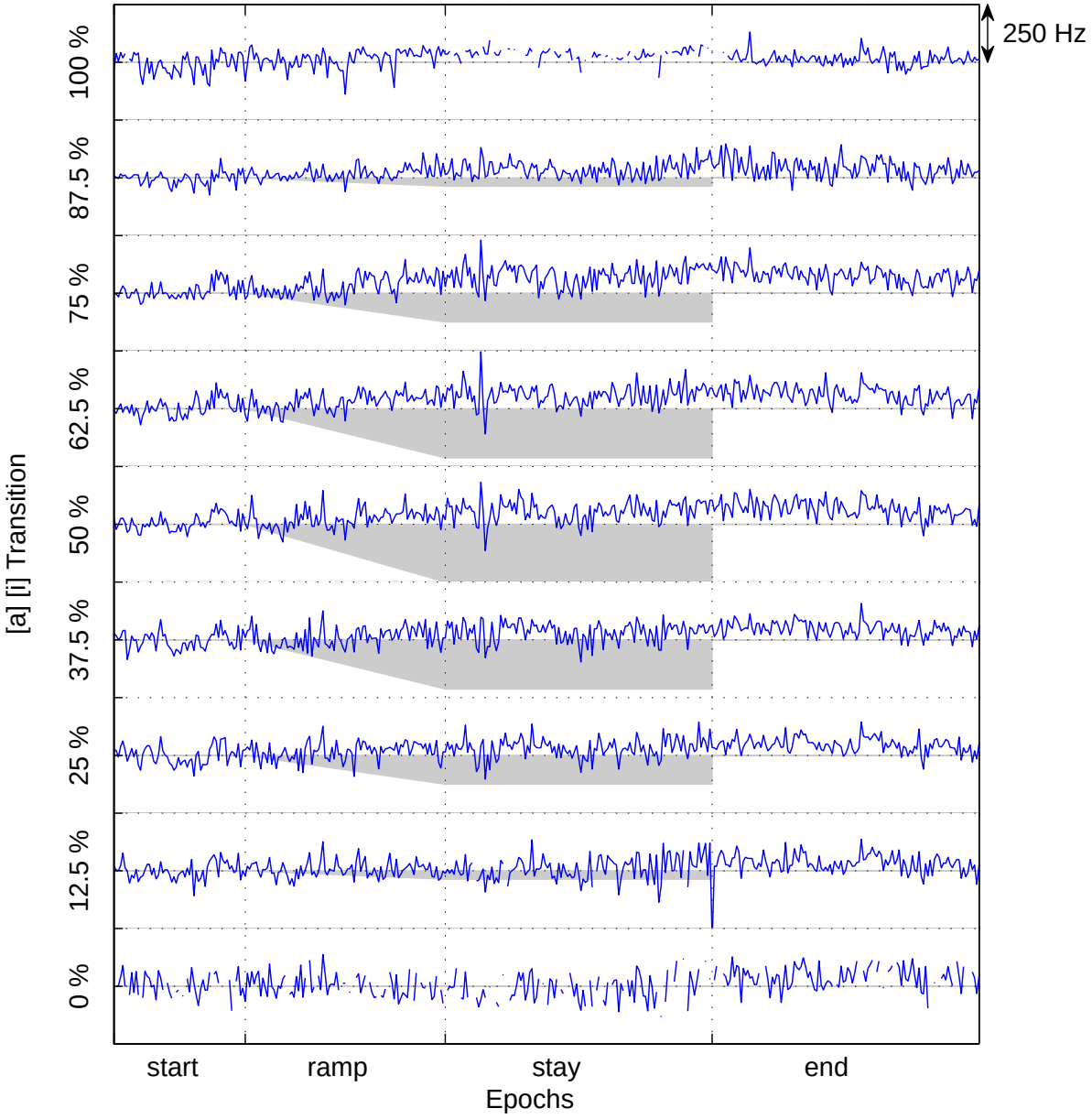


Figure 5.3: Female subject : downshift

Subject's compensation towards the opposite of the perturbation starts at the end of the ramp phase and is maintained throughout the end phase. The compensation seems to be higher in the middle of the transition than at the bounds.

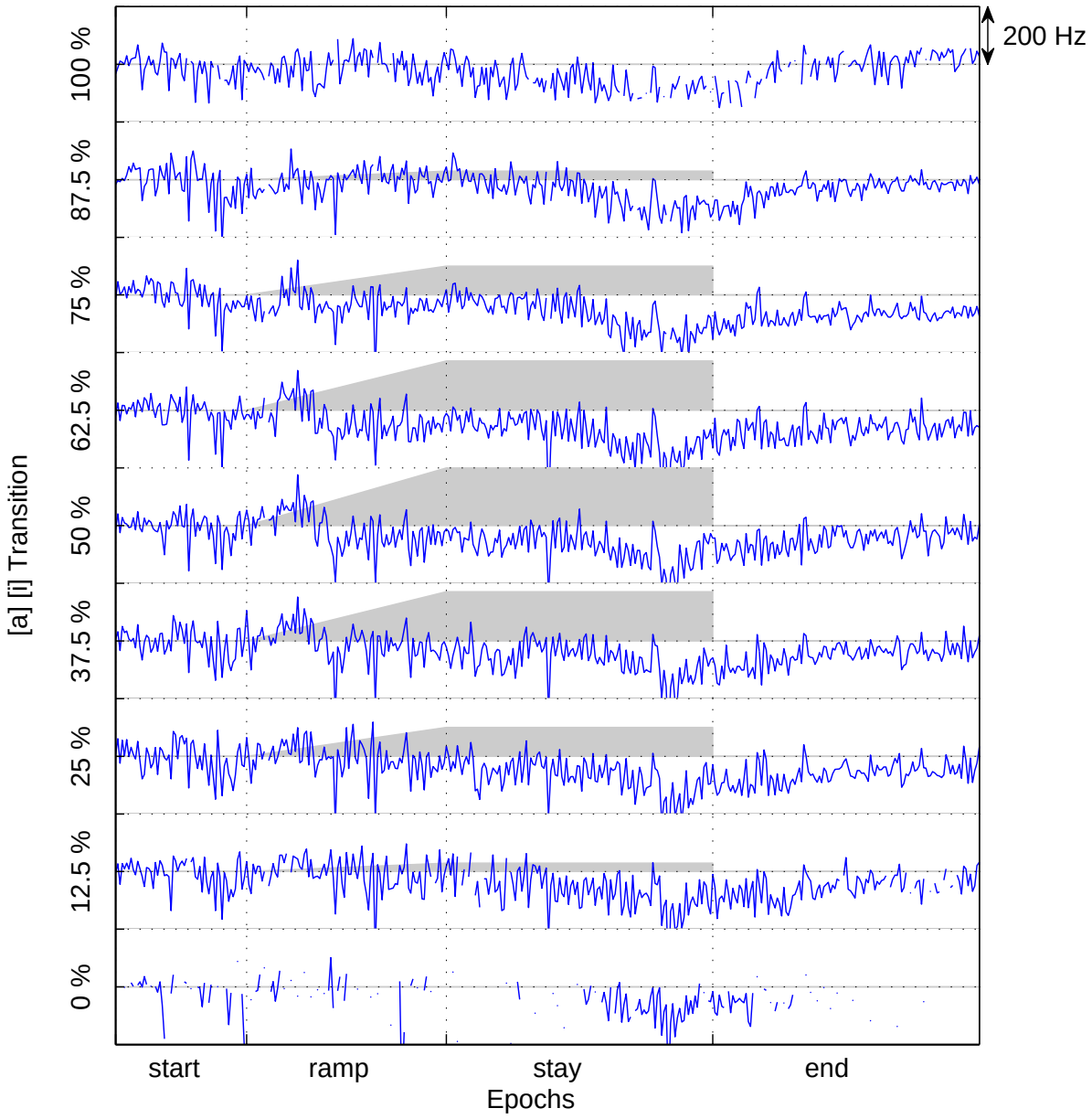


Figure 5.4: Female subject : upshift

First, subject's compensation is very weak and then strongly increases at the end of the "stay" phase. Is it just a coincidence ?

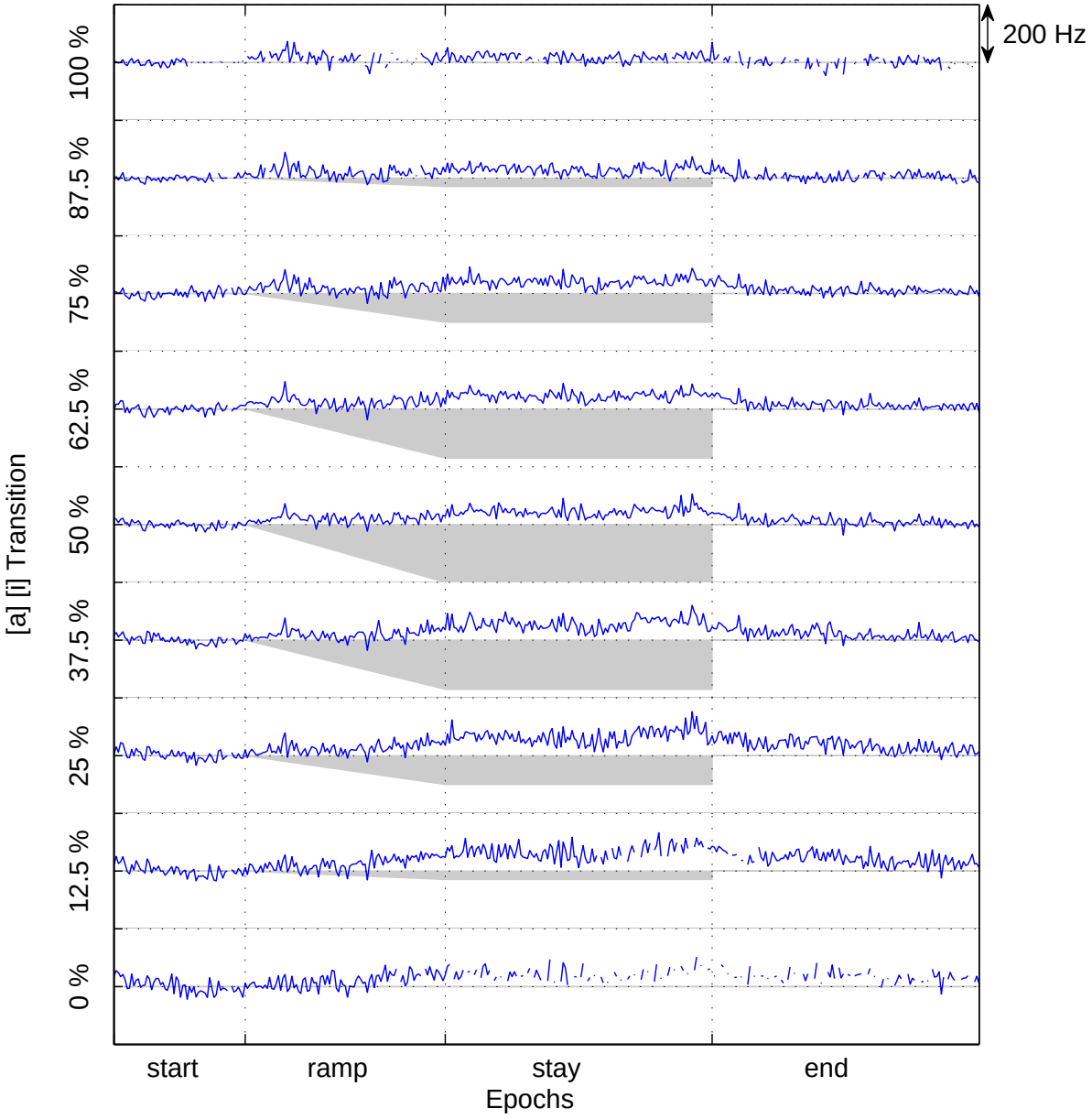


Figure 5.5: Male subject : downshift

Although one can clearly see the compensation, it seems to be greater on one side of the transition.

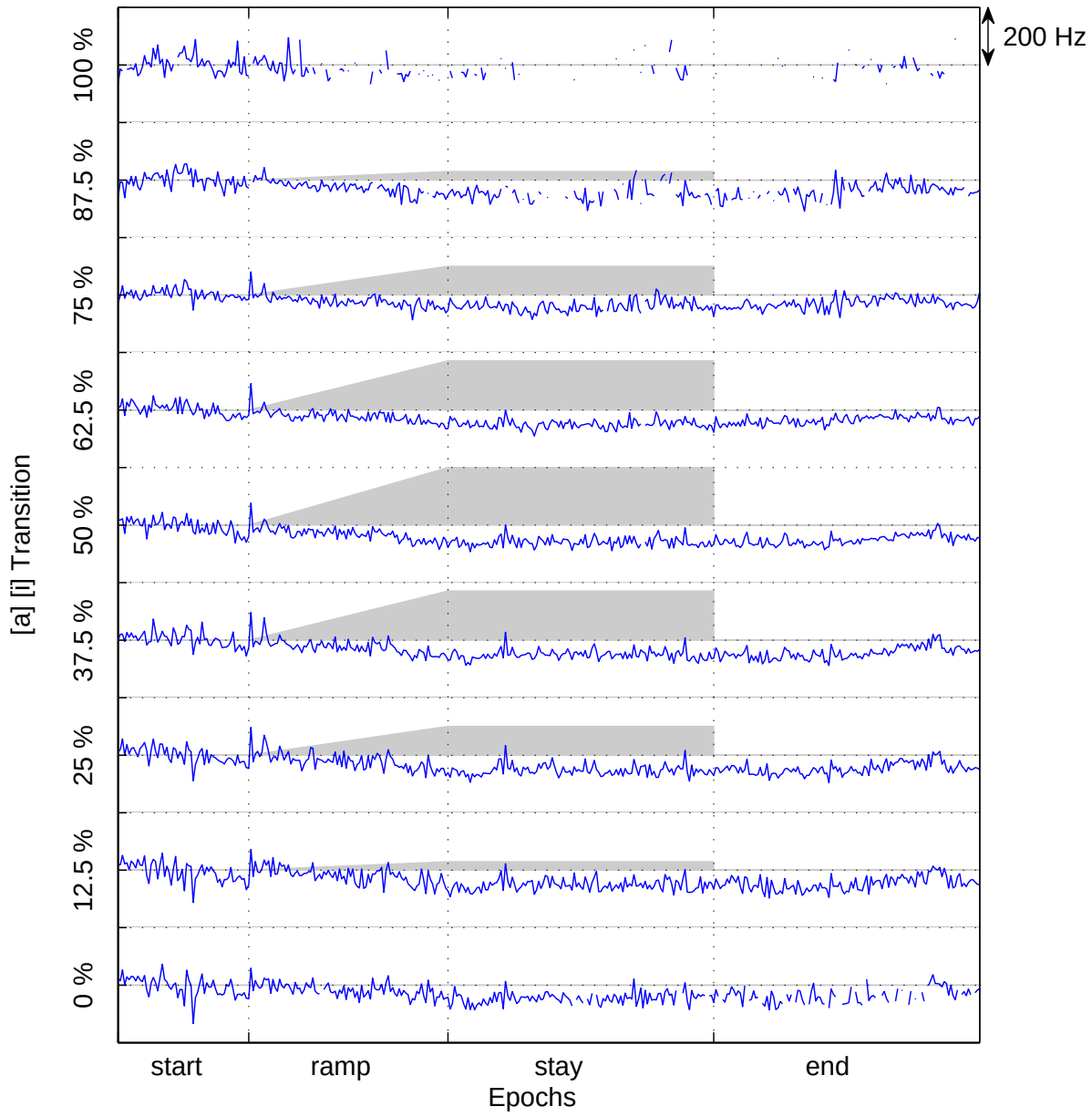


Figure 5.6: Male subject : upshift

Here the compensation seems to be equally distributed along the trajectory.

Appendix A

Gain issues

In Section A.3 we described a gain adaptation method, to control the gain changes introduced by a formant shift. Since we developed various methods to reach this, we have summarized these methods here.

A.1 Peak gain adaptation (Method 1)

The basic idea is to control the formant's peak gain by calculating a gain factor that will compensate the undesired gain change due to the frequency shift. We therefore calculate the original peak's gain expressed in Equation A.1

$$\begin{aligned} G(\theta_i) &= \left|V(z=e^{j\theta_i})\right| && \text{(A.1)} \\ &= \frac{G}{\prod_{k=1}^{M}|(1-c_k e^{-j\theta_i})(1-c_k^{\star}e^{-j\theta_i})|} && \text{(A.2)} \end{aligned}$$

The new peak's gain, i.e. the shifted peak's gain, is given by Equation A.3

$$\begin{aligned} \hat{G}(\hat{\theta}_i) &= \left|V(z=e^{j\hat{\theta}_i})\right| && \text{(A.3)} \\ &= \frac{G}{\prod_{k=1}^{M}\left|(1-\hat{c}_k e^{-j\hat{\theta}_i})(1-\hat{c}_k^{\star}e^{-j\hat{\theta}_i})\right|} && \text{(A.4)} \\ &= \frac{1}{\left|(1-\hat{r}_i)(1-\hat{r}_i e^{-j2\hat{\theta}_i})\right|}\frac{G}{\prod_{k\neq i}^{M}\left|(1-c_k e^{-j\hat{\theta}_i})(1-c_k^{\star}e^{-j\hat{\theta}_i})\right|} && \text{(A.5)} \end{aligned}$$

Now, we only need to multiply the ratio of these two gain factors with the shifted system function $\hat{V}(z)$ and obtain the peak normalized system function $\hat{V}_n(z)$ given by Equation A.6.

$$\hat{V}_n(z) = \frac{G(\theta_i)}{\hat{G}(\hat{\theta}_i)} \cdot \hat{V}(z) \qquad \text{(A.6)}$$

with the normalizing factor

$$\frac{G(\theta_i)}{\hat{G}(\hat{\theta}_i)} = \frac{1}{\left|(1-\hat{r}_i)(1-\hat{r}_i e^{-j2\hat{\theta}_i})\right|} \frac{\prod_{k=1}^{M}\left|(1-c_k e^{-j\theta_i})(1-c_k^\star e^{-j\theta_i})\right|}{\prod_{k\neq i}^{M}\left|(1-c_k e^{-j\hat{\theta}_i})(1-c_k^\star e^{-j\hat{\theta}_i})\right|} \tag{A.7}$$

We see that, in order to control the peak's gain, we are introducing a gain factor that will affect all the other peaks, i.e. change their magnitude.

Note: We recall that we are adapting $V(z)$, which is the predicted system function after having preemphasized the signal. This means that maintaining the peak's gain at the same level in $V(z)$ will cause a peak gain slope of $-6dB/oct$ after having deemphasized the signal.

A.2 Peak radius adaptation (Method 2)

To avoid that the magnitude of the neighbored peaks change we have to modify the method introduced in Section **??**. We still want to keep the peak's magnitude constant during a shift without affecting the other peaks. Therefore, instead of using an overall gain factor, we change only the peak's radius[1] $\hat{r}_i$. We can write Equation A.7 as follows:

$$\frac{G(\theta_i)}{\hat{G}(\hat{\theta}_i)} = \frac{1}{\left|(1-\hat{c}_i e^{-j\hat{\theta}_i})(1-\hat{c}_i^\star e^{-j\hat{\theta}_i})\right|} \cdot \underbrace{\frac{\prod_{k=1}^{M}\left|(1-c_k e^{-j\theta_i})(1-c_k^\star e^{-j\theta_i})\right|}{\prod_{k\neq i}^{M}\left|(1-c_k e^{-j\hat{\theta}_i})(1-c_k^\star e^{-j\hat{\theta}_i})\right|}}_{G_{Rest}} \tag{A.8}$$

Using G_{Rest} to describe the gain term added by the poles which will not be shifted leads to Equation A.9

$$\frac{G(\theta_i)}{G(\hat{\theta}_i)} = \frac{G_{Rest}}{\left|(1-\hat{c}_i e^{-j\hat{\theta}_i})(1-\hat{c}_i^\star e^{-j\hat{\theta}_i})\right|} \tag{A.9}$$

We recall that we wish to leave the gain of the peak unchanged during a frequency shift. Setting $G(\theta_i) = G(\hat{\theta}_i)$ satisfies this requirement and we can transform Equation A.9 to obtain

$$\left|(1-\hat{c}_i e^{-j\hat{\theta}_i})(1-\hat{c}_i^\star e^{-j\hat{\theta}_i})\right| = G_{Rest} \tag{A.10}$$

which leads to

$$\left|\left(1-\hat{r}_i\right)\cdot\left(1-\hat{r}_i e^{-j2\hat{\theta}_i}\right)\right| = G_{Rest} \tag{A.11}$$

and can be modified to

$$\left(1-\hat{r}_i\right)\cdot\sqrt{1+\hat{r}_i^2-2\hat{r}_i\cos(2\hat{\theta}_i)} = G_{Rest} \tag{A.12}$$

and finally to Equation A.13 by squaring on both sides

[1] The radius of the formant whose frequency is being shifted.

$$\left(1-\hat{r}_i\right)^2 \cdot \left(1+\hat{r}_i^2-2\hat{r}_i\cos(2\hat{\theta}_i)\right) = G_{Rest}^2 \tag{A.13}$$

Equation A.13 can be written as 4^{th} order polynomial

$$\hat{r}_i^4 - K\hat{r}_i^3 + 2\left(K-1\right)\hat{r}_i^2 - K\hat{r}_i + 1 - G_{Rest}^2 = 0 \tag{A.14}$$

with

$$K = 2\left(1+\cos(2\hat{\theta}_i)\right) \tag{A.15}$$

Equation A.14 can be solved using an iterative[2] root-finding algorithm. Only one value for $\hat{r}_i$ is a real number satisfying $0 < \hat{r}_i < 1$. Once the new radius $\hat{r}_i$ has been calculated it is used as the new pole's radius within the system function.

A.3 Gain adaptation at $0Hz$ (Method 3)

The third method relies on the physical properties of the vocal tract. In [1] the vocal tract is modeled as a 4-tube resonator. Based on this model, the spectral contribution of each formant is defined as follows

$$|H_k(f)| = |H_k(s=j\omega)| = \frac{s_k s_k^\star}{|(s-s_k)|\,|(s-s_k^\star)|} \tag{A.16}$$

Where $\{s_n, s_n^\star\}$ are the complex conjugated pole pairs of each formant in the s-plane. Equation A.16 is equivalent to Equation A.17 in the z-domain

$$|H_n(f)| = \left|H_n(z=e^{j\omega})\right| = \frac{(1-c_k)(1-c_k^\star)}{|(1-c_k z^{-1})|\,|(1-c_k^\star z^{-1})|} \tag{A.17}$$

We can easily see that the spectral contribution of each formant defined by Equation A.17 is $0dB$ for $f = 0Hz$.

In our case, we don't have $0dB$ at $f = 0Hz$ because the system function provided by the LPC analysis also contains a gain factor G. This gain factor is a "leftover" from the source. However, we know that the spectral contribution of the vocal tract is $0dB$ at $f = 0Hz$. Whenever we shift a formant, we must ensure that this contribution stays $0dB$ at $f = 0Hz$. This means that the overall gain at at $f = 0Hz$, i.e. $|V(z=1)|$ stays constant.

This requirement is similar to those introduced in Method 1 and 2. The main difference is that we do not "adapt" the formant's peak gain, but we "adapt" the gain at $f = 0Hz$.

$$\hat{V}_n(z) \quad = \quad \frac{G(\theta=0)}{\hat{G}(\theta=0)} \cdot \hat{V}(z) \tag{A.18}$$

[2]It is theoretically possible to solve Equation A.14 analytically using Ferrari's formula, but this is a very laborious work. Analytical solutions provided by Maple do not fit in 1 page...

with

$$\frac{G(\theta=0)}{\hat{G}(\theta=0)} = \frac{|(1-c_i)(1-c_i^\star)|}{|(1-\hat{c}_i)(1-\hat{c}_i^\star)|} \underbrace{\frac{\prod_{k\neq i}^{M}|(1-c_k)(1-c_k^\star)|}{\prod_{k\neq i}^{M}|(1-c_k)(1-c_k^\star)|}}_{=1} \tag{A.19}$$

And finally we obtain the simple relation

$$\frac{G(\theta=0)}{\hat{G}(\theta=0)} = \frac{1-2r_i\cos(\theta_i)+r_i^2}{1-2\hat{r}_i\cos(\hat{\theta}_i)+\hat{r}_i^2} \tag{A.20}$$

A.4 Results (Method 1, 2 & 3)

We can see how the presented adaptation methods modify a single f_1 formant shift in Figure A.1 for f_1 being shifted down (left size) and up (right size). Formant shifts with regard to f_2 are represented in Figure A.2.

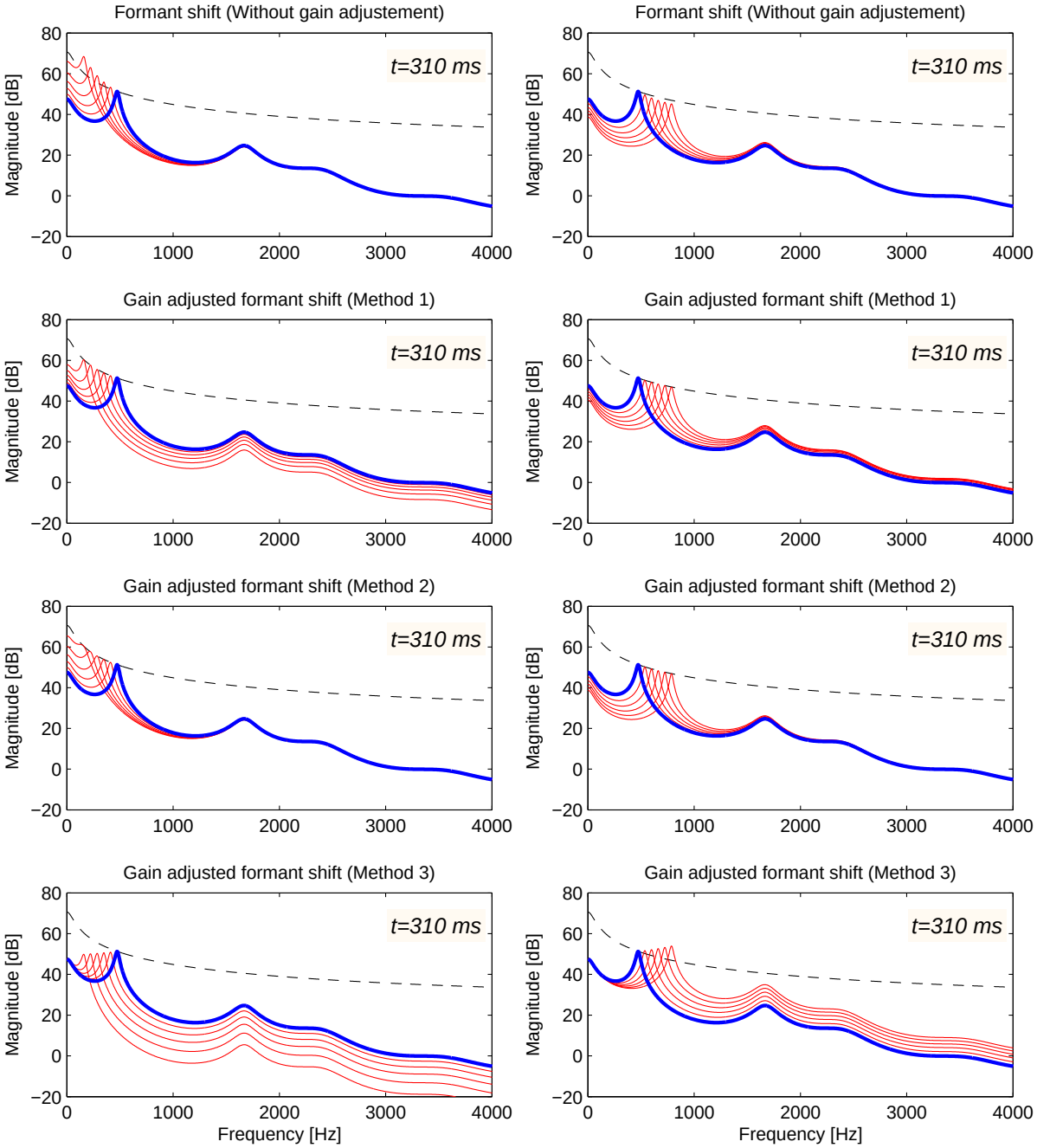
In each of these Figures we can see the original spectrum represented as a blue line. Each red line represents a "shifted" spectrum. The amount of perturbation ranges from 0 to $300Hz$ for an up- or downshift. The black dotted line shows the $-6dB/oct$ slope as discussed before.

Each panel of Figures A.1 and A.2 represents one of the methods used, while the first panel on top stands as reference, i.e. no gain adaptation method has been used.

We can clearly see that the first method changes the gains of the entire spectrum, while the second method only slightly changes the peaks next to it. Each peak is on the $-6dB/oct$ slope (dotted line). In the lowest panel (method 3) we can see that the gain a $f = 0Hz$ does not change during a shift. Furthermore, the formant's gain decreases when shifted towards lower frequencies and increases during an upward-shift.

A.5 Summary

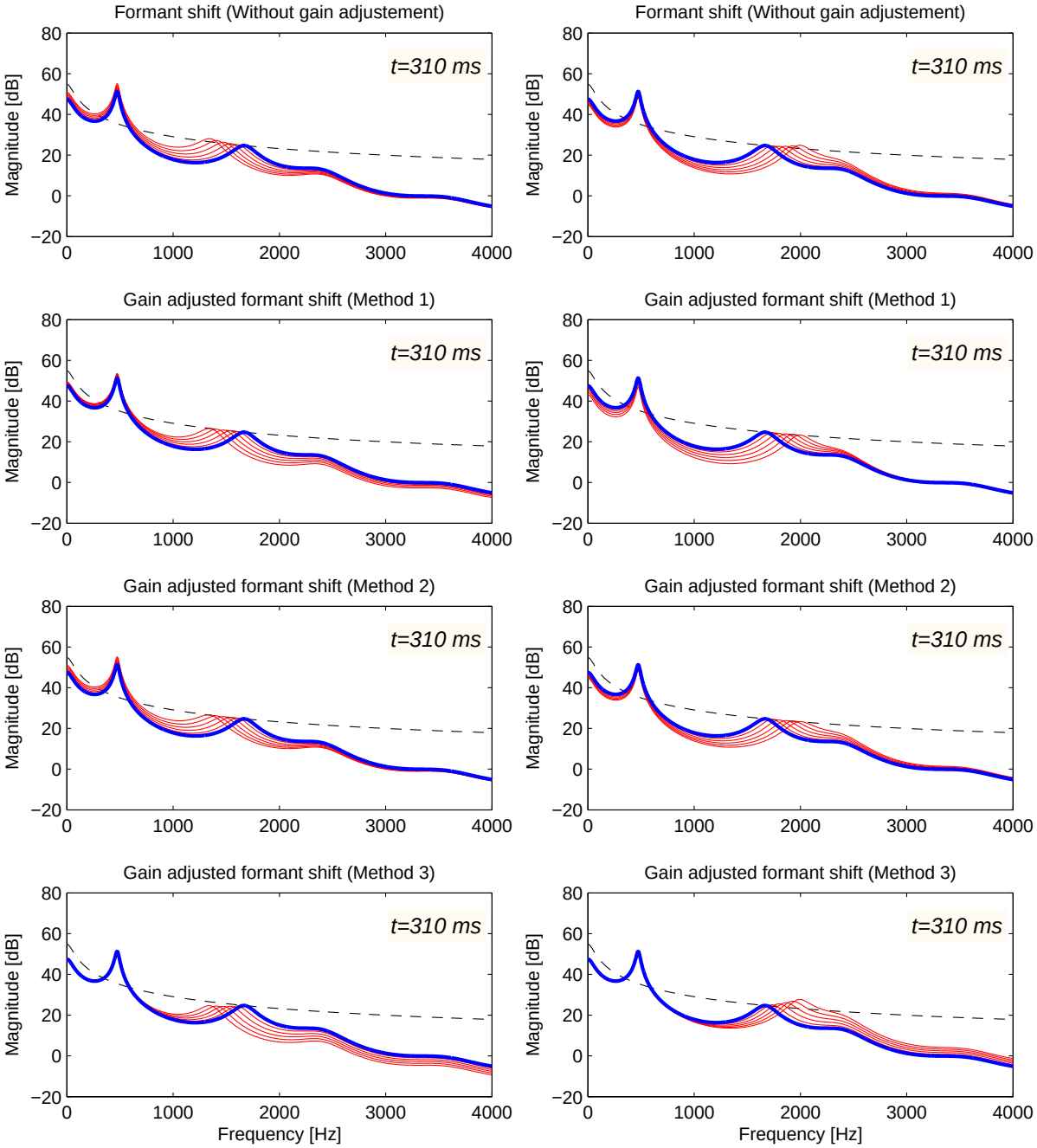
We have studied these methods in order to make the formant shift sound as natural as possible. This is important because it will affect how subjects will perceive the perturbation, and hence how they will react to this perturbation. We formulated method 1 and 2 having in mind that a shifted formant peak should stay on a $-6dB/oct$ slope. These 2 methods allow us to reach this goal but, as a matter of fact, this does not exactly correspond to the "physical" properties of the vocal tract, because the overall spectrum is flatter for vowels with a low f_1, than for vowels having a high f_1, which means that the $-6dB/oct$ is moving up and down depending on the frequency of the first formant. This is taken into account in the third Method.



(a) f_1 down shift

(b) f_1 up shift

Figure A.1: Gain adaptation for an $f_1$shift



(a) f_2 down shift

(b) f_2 up shift

Figure A.2: Gain adaptation for an f_2 shift

Bibliography

[1] G. Fant. *Acoustic Theory of Speech Production*. Mouton, The Hague, 1960. 7, 20, 36, 52, 65

[2] James L. Flanagan. *Speech Analysis, Synthesis, and Perception*. Springer-Verlag, Berlin, Germany, second edition, 1972. 7

[3] R. L. Freyman, R. K. Clifton, and R. Y. Litovsky. Dynamic processes in the precedence effect . *Acoustical Society of America Journal*, 90:874–884, August 1991. 20

[4] John F. Houde and Michael I. Jordan. Sensorimotor Adaptation in Speech Production. *Science*, 279(5354):1213–1216, 1998. 1, 2

[5] John F. Houde and Michael I. Jordan. Sensorimotor Adaptation of Speech I: Compensation and Adaptation. *J Speech Lang Hear Res*, 45(2):295–310, 2002. ix, 1, 3

[6] J. Makhoul. Linear prediction: A tutorial review. *Proceedings of the IEEE*, 63:561–580, 1975. 27

[7] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 1992. 29

[8] David W. Purcell and Kevin G. Munhall. Compensation following real-time manipulation of formants in isolated vowels. *The Journal of the Acoustical Society of America*, 119(4):2288–2297, 2006. 3

[9] L. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals*. Prentice-Hall, Englewood Cliffs, NJ, 1978. 25, 26, 27

[10] Kenneth Stevens. *Acoustic Phonetics*. MIT Press, Cambridge, 1998. ix, 7, 8, 9

[11] Gautam K. Vallabha and Betty Tuller. Systematic errors in the formant analysis of steady-state vowels. *Speech Commun.*, 38(1):141–160, 2002. 31

[12] D. Wolpert, Z. Ghahramani, and M. Jordan. Are arm trajectories planned in kinematic or dynamic coordinates, 1995. 1, 3, 4

[13] Kun Xia and Carol Espy-Wilson. A new strategy of formant tracking based on dynamic programming. *ICSLP-2000*, 3:55–58, 2000. 36, 38

Index