Diploma Thesis

Enhancement of spectral envelope modeling in HMM-based speech synthesis

Florian Krebs

Signal Processing and Speech Communication Laboratory
Graz University of Technology
Head: O. Univ.-Prof. Dipl.-Ing. Dr. techn. Gernot Kubin

Assessor: O. Univ.-Prof. Dipl.-Ing. Dr. techn. Gernot Kubin
Supervisor: Dipl.-Ing. Dr. sc. ETH Harald Romsdorfer

Graz, September 2010

Abstract

Hidden Markov Model (HMM)-based text-to-speech synthesis systems have grown in popularity over the last years, as they are very flexible in generating speech with various speaker characteristics, emotions and speaking styles. HMM-based text-to-speech synthesis is also often referred to as statistical parametric speech synthesis. To create a voice, statistical parametric models are built from a speech corpus. Using these models, an arbitrary text can be converted into a speech parameter sequence that fulfills a maximum likelihood criterion.

Due to the statistical processing, certain characteristics of a parameter sequence get lost. To recover these characteristics, variance based features in the parameter generation process are investigated in this thesis, and a new *roughness* based feature is proposed that describes the presence of fast variations of a cepstral time sequence. The values of both the variance and roughness based features are significantly smaller in standard HMM-based synthetic speech than in natural speech. A method to increase the roughness of a parameter sequence is described and was tested in a listening test. It was found that the roughness criterion reduces temporal over-smoothing, but also introduces audible discontinuities.

All the algorithms have been developed and tested using the HMM-based Speech Synthesis System (HTS) released by the Nagoya Institute of Technology.

Kurzfassung

Hidden Markov Model (HMM)-basierte Sprachsynthesesysteme haben in den letzten Jahren an Beliebtheit gewonnen, da sie eine einfache Methode bieten, Sprache mit verschiedensten Sprechercharakteristiken, emotionalen Färbungen oder Sprechstilen zu erzeugen. HMM-basierte Sprachsynthesesysteme werden oft auch als statistische parametrische Sprachsynthesesysteme bezeichnet. Um eine Stimme zu generieren, werden statistische parametrische Modelle aus einem Sprachkorpus gebildet. Mithilfe dieser trainierten Modelle können mittels einer Maximum Likelihood Schätzung Sprachparametersequenzen aus einem beliebigen Text gewonnen werden.

Bei der statistischen Modellierung des Trainingskorpus gehen bestimmte Eigenschaften der Sprachparametersequenzen verloren, was zu einer Verschlechterung der Sprachqualität führt. In dieser Diplomarbeit soll der Einfluss von Varianz-basierten Merkmalen auf die Sprachqualität untersucht werden, als auch ein neues *Rauheits*-Merkmal eingeführt werden, das schnelle Variationen einer Parameterkurve beschreibt. Sowohl die Varianz- als auch die Rauheitswerte von Sprache, die mit einem Standard-HMM System generiert wurde, sind signifikant geringer als bei natürlicher Sprache. Eine Methode zur Erhöhung der Rauheit wird vorgestellt und in einem Hörtest getestet. Es stellte sich heraus, dass das eingeführte Rauheitskriterium zwar das Problem einer übermäßigen Glättung der Parameterkurven behebt, aber zusätzlich hörbare Artefakte erzeugt, die die Sprachqualität beeinträchtigen.

Alle Versuche und Tests in dieser Diplomarbeit wurden mit dem HMM-based Speech Synthesis System (HTS) durchgeführt, das vom Nagoya Institute of Technology entwickelt wurde.

Acknowledgments

This thesis was written at the Signal Processing and Speech Communication Laboratory (SPSC) at Graz University of Technology in the year 2010.

I want to thank my supervisor Harald Romsdorfer, who has always been very open to new ideas and shared a lot of time and his rich experiences with me. Furthermore, I want to express my gratitude to the members of the institute (Stefan Petrik and Robert Peharz) for their inputs and interesting discussions, Markus Köberl and Andres Lässer for the infrastructural support and their indestructible faith in Linux, the colleagues from the DSP-lab and the IEM for all the lunch- and coffee-breaks and finally my friends and flatmates for the wonderful time beyond the working hours.

This thesis would not have been possible without the support of these people.

Special thanks go to my parents Karin and Stefan for all the love and for always encouraging me to do the things in life that I really love.

Graz, September 2010 Florian Krebs

Contents

List of Abbreviations

CDHMM	Continuous Density Hidden Markov Models
F0	Fundamental frequency
GR	Global Roughness
GV	Global variance
HMM	Hidden Markov Model
HNM	Harmonic plus noise model
HSMM	Hidden Semi-Markov Model
HTS	Hidden Markov Model-based text-to-speech system
IFFT	inverse Fourier transform
IIR	Infinite impulse response
LMA	Log magnitude approximation
LPC	Linear prediction coding
LR	Local Roughness
LV	Local variance
MFCC	Mel-frequency cepstral coefficient
MLSA	Mel log spectrum approximation
MSD-HMM	Multi-space distribution Hidden Markov Model
PDF	Probability density function

Chapter 1

Introduction

Speech is one of the most important and complex interaction forms between humans. Since the 2nd half of the 18th century people have been trying to create synthetic speech. The first attempts were made in 1773 by Ch. G. Kratzenstein and Wolfgang von Kempelen, who generated voice-like sounds with a mechanical model using resonance tubes. In the 1950ies an electrical model was introduced to produce speech by passing an electric source signal through a filter. Since 1970 the development of speech synthesis has been closely associated with the raise of the computer technology. Computers have made it possible to concatenate stored words or shorter segments, and the first systems were developed to convert text to speech.

Today's speech synthesis systems can be categorized into rule based and data based synthesis:

Rule based synthesis relies on rules developed by linguists to generate speech parameter without using samples of human speech. This permits great flexibility on one hand but - as the generation of speech is a very complex procedure - it is very difficult to find optimal rules to make speech sound natural on the other hand.

Data based synthesis systems concatenate acoustical units to generate speech. With the increasing memory and processing power of computers in the last decades, they have more and more outperformed rule-based approaches.

Furthermore, the data based synthesis systems can be divided into unit-selection and statistical parametric speech synthesis systems:

In 1992, the ATR ν-talk system [SKIM92] was published and was the first to show the effectiveness of automatic selection of appropriate units [BZT07]. This so-called *unit-selection approach* is still dominant in speech synthesis. The quality of the generated speech is directly related to the quality and amount of the recordings of the speech database. This facilitates very natural-sounding speech, but also limits the output speech to the speaking style and voice of the original recordings. To obtain different speakers and speaking styles, recordings of huge databases are required, which is very time and money consuming. IBM's stylistic synthesis [EAB+04] is one good example for such a system. One of the probably best known unit selection systems is the Festival Speech Synthesis System [TBC98] developed at the University of Edinburgh.

Regarding the flexibility in generating speech variations, *statistical parametric speech synthesis* produces promising results and has grown in popularity over the last few years. In direct contrast to the selection of actual instances of speech from a database, statistical parametric speech synthesis might be most simply described as generating the *average* of some sets of similar-sounding speech segments [BZT07]. Statistics are obtained from a database and fed into generative statistical models. According to these statistics, new speech parameter can be generated. Compared to unit-selection systems, parametric statistical systems still lack naturalness of the generated speech. Important details of the speech waveform get lost due to the averaging in the training step, and the models have to be very complex to represent natural-sounding speech.

The term *parametric* means that a parametric representation of the speech waveform is used, not the waveform itself. The speech signal is split into a filter signal, which represents the resonances of the vocal tract, and a source signal, which represents the excitation signal produced by the glottis. Using a parametric representation facilitates the modification of the voice characteristics, the speaking style or the emotional expression of the generated speech. The synthesis system can be applied to various languages with little modification and has a relatively small footprint [BZT07].

Of course, there have also been proposed hybrid speech synthesis systems that unify the advantages of both unit-selection and statistical parametric systems [ZTB09]. E.g., in [YZTW07], the authors take speech spectrum vectors from a codebook to resolve the over-smoothing problem of the statistical modeling.

In the Blizzard Challenge of the recent years, where a common database is provided for participants to build a synthetic voice, Hidden Markov Model (HMM)-based synthesis systems have proved to be the most preferred (MOS score) and most understandable (WER score) systems among statistical parametric speech synthesis systems, even though its naturalness is still far from natural speech [BZT07, ZTB09]. The success of the HMMs can be explained by the following reasons [TT07]: 1) the HMM is capable of modeling a time sequence of speech parameters as it has been widely used in speech synthesis, 2) many techniques, that have originally been developed for HMM-based speech recognition, can be applied, 3) the HMM is mathematically tractable and hence facilitates modifications of the speech parameters in order to change the voice characteristics.

1.1 Motivation

As I am working as a sound engineer in the film business, I very often encounter the problem of recording voices in noisy environments. It happens regularly that the recorded sound cannot be used and has to be exchanged in the post-production, using a very time-consuming procedure called automatic dialog replacement (ADR). This implies finding the right speaker and recording new samples of the speech that synchronize with the image. If clean speech could be generated to exchange the noise-polluted parts, a lot of time and money could be saved. This very complex task requires high quality speech synthesis that allows mimicking the actor's speaking style, emotion and voice characteristics. At the current state of the art, we are still far from realizing such a system. In this thesis, I want to find out where the problems in current systems are and to contribute a small step in the direction of the holy grail of speech synthesis, which is to produce speech that is indistinguishable from natural speech.

1.2 Goal

In the case where only few training data is available and the objective is to generate speech with a great variety of emotions, styles and speakers, a parametric statistical speech system performs best. The HMM-based speech synthesis system (HTS) [YMKK99] is an open source framework for HMM training and synthesis, which is widely used among researchers. Within the limited time of this thesis I want to achieve the following aims:

- Understand and document the speaker dependent demo scripts provided with HTS 2.1.1
- Analyze the weaknesses of the system and review different solutions that have been proposed so far.
- Develop a new approach, implement and evaluate it

1.3 Organization of the thesis

This thesis describes different algorithms to enhance the generation of natural-sounding speech using the Hidden Markov Model-based Speech Synthesis System (HTS).

Chapter 2 reviews some theoretical aspects that are important to understand this thesis. The source-filter model of speech production is introduced and two methods are described to extract the spectral envelope of a speech signal: linear prediction coding (LPC) and the cepstral analysis. In the previous versions of the HTS, a Mel Logarithmic Spectrum Approximation (MLSA) filter converted the cepstrum back to the speech spectrum and is shortly described. In the current version, the authors propose the integration of STRAIGHT (Speech Transformation and Representation using Adaptive Interpolation of weiGHTed spectrum [Kaw97]) as a synthesis filter. An overview of different vocoding techniques is given with a focus on STRAIGHT.

Chapter 3 describes the system architecture of a typical HMM-based Speech Synthesis system. The parameter generation algorithm is derived and known drawbacks are discussed.

Chapter 4 describes the HTS 2.1.1 system. Drawbacks of the system are identified and described.

In chapter 5, variance characteristics of a speech signal, which have been removed during the modeling process, are investigated. A roughness based feature is proposed to describe the presence of high frequencies in a parameter trajectory.

A listening test was conducted to evaluate the different approaches. The test design and the results are presented in chapter 6.

Chapter 2

Theoretical Background

2.1 Parametric models of speech

It is very difficult to find a mathematically tractable parametric model of speech that describes exactly the real anatomy and physiology of the human speech production system. What we need to find is a simple mathematical representation that describes well the essential properties of the speech signal.

2.1.1 Source-filter model vs. Harmonic+Noise

The source-filter model

In the source-filter model (Fig. 2.1), the speech signal is split into two parts: the excitation source and the vocal tract filter, see also [VHH98].

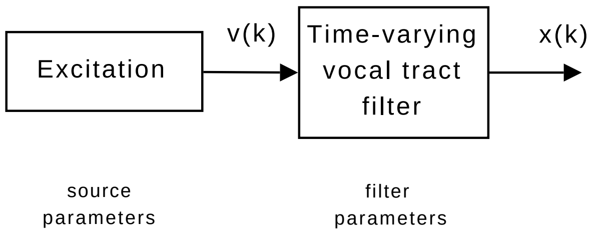

Figure 2.1: digital source-filter model

The excitation source The source parameters of the speech model represent the excitation of the lungs and the vocal chords. Four different excitation patterns can be described:

- **Impulse trains** (for voiced sounds, e.g., vowels) are produced by periodically opening and closing the glottis while exhaling, leading to a quasi-periodic oscillation determining the fundamental frequency F0.

- **Noise** (for unvoiced sounds, e.g., fricatives) is produced by air turbulences in the larynx and pharynx, while exhaling through the open glottis.

- **Single impulses** (for unvoiced sounds, e.g., plosives) are produced by exhaling with open glottis. The lips are closed to increase the pressure in the vocal tract and then suddenly opened.

- **Silence** between other excitation patterns is important to create certain sounds (e.g., plosives).

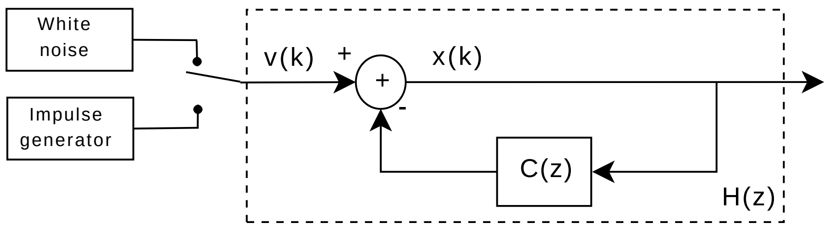

Figure 2.2: All-pole vocal tract model

The filter The human vocal tract is the cavity where the excitation that is produced at the larynx is filtered and includes the pharynx, the oral and the nasal cavities. It can be simplified by a set of tubes, that has open ends (the lips and the nasals) and a closed end (the glottis), forming a body where resonances can occur due to reflections at both ends. These resonances correspond to standing waves inside the tube and therefore depend on the extension of the vocal tract. The resonances appear at certain frequencies called *formants.* The average length of the vocal tract of an adult man is about 17 cm, which corresponds to a standing wave with a fundamental frequency of about 500 Hz. While talking, we continuously change the volumes of our vocal tract (with the tongue or the lips), which means the corresponding resonance filter is time-varying. Nevertheless, the filter can be assumed to be quasi-stationary, when restricting our analysis to periods of about 5-10 ms, leading to a minimum sampling rate of 100 Hz for the filter coefficients. In the system described in the later sections (see chapter 4) a sampling rate of 200 Hz is used.

Describing a model structure that perfectly models the vocal tract, leads to very complex filters with a lot of zeros and poles [VHH98]. As we want to encode and decode with our parametric model, a filter has to be found, that can be inverted without any stability problems. A minimum-phase system exactly has this property, as by definition all its zeros are located inside the unit circle. Any system can be split into a minimum-phase system and an allpass, and as the minimum-phase part includes all the relevant information about the formants, the allpass can be neglected in this case. The remaining zeros of the minimum-phase system can be approximated by a series of poles, such that an all-pole filter can be designed that sufficiently models the formant structure of speech as seen in 2.2. The vocal tract filter $H(z)$ is represented by a feedback-loop with the filter coefficients (poles) given in $C(z)$ [VHH98].

The harmonic plus noise model (HNM)

The harmonic plus noise model was originally developed in 1987 by Daniel W. Griffin at the MIT [Gri87] as an approach to reduce the “buzziness” of a vocoder.

The HNM assumes the speech signal to be a sum of two parts:

- The **harmonic part**, represented by a sum of sinusoids of multiples of the fundamental frequency (deterministic part)
- The **noise part**, using an all-pole filter excited by random white Gaussian noise (stochastic part)

When synthesizing voiced sounds, frequencies beyond a *maximum voiced frequency* are generated by the fundamental frequency sinusoid and its harmonics, while frequencies above the threshhold are represented by noise modulated by the envelope. This makes the synthesis of voiced segments sound more natural. The HNM has been integrated into the HTS system in [Hem06] and reported to improve the naturalness of the synthesized speech.

2.1.2 Autoregressive model - linear prediction coding (LPC)

The vocal tract can be interpreted as an ensemble of various acoustic resonators, that produce decaying exponentials after being excited. These decaying exponentials are predictable when no new excitation signal occurs and can be modeled by a linear filter with the coefficients a_k. The fact that neighboring samples in speech are correlated, can be exploited for speech coding.

An autoregressive filter (Fig. 2.3) can be designed with an equivalent structure as the source-filter model of the speech production model in Fig. 2.2. When the filter coefficients a_k are chosen to match the vocal tract filter coefficients c_k, the prediction error signal $d(k)$ represents the excitation $v(k)$ of the vocal tract. The LPC filter coefficients a_k can be found by minimizing the error between the predicted speech signal and the actual speech signal.

The speech signal $x(k)$ at time frame k can finally be described by the LPC coefficients a_k and the residual $d(k)$, where

$$d(k) = x(k) - \sum_{i=1}^{n} a_i x(k-i)$$

with n denoting the number of past samples used for prediction.

The number of coefficients n determines the accuracy of the model, and hence the size of the residual.

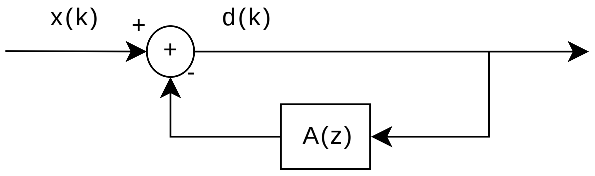

Figure 2.3: Digital AR filter

2.1.3 Exponential model - cepstral analysis

Another way of splitting the speech signal into its source and filter component is the use of the cepstrum. This idea arises from the fact that, when looking at the spectrum of a speech signal, two parts can be distinguished: a fast varying part representing the excitation source, and a slow varying part representing the envelope of the curve. The envelope can be interpreted as a filter shaping the excitation signal, therefore separating the source signal from the filter signal is equivalent of separating the high frequency components from the low frequency components (envelope) of the spectrum.

The cepstrum of a speech signal is computed as follows (Fig. 2.4) [Zoe02]):

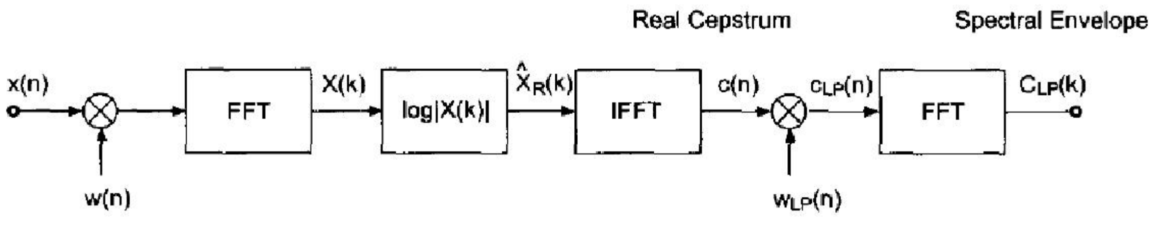

Figure 2.4: Spectral envelope computation by cepstrum analysis [Zoe02]

First, the discrete Fourier transform $X(k)$ of the time signal $x(n)$ is calculated to yield the spectrum of the signal:

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{kn} = |X(k)|e^{j\varphi_x(k)} \tag{2.1}$$

Next, the complex natural logarithm is taken:

$$\hat{X}(k) = logX(k) = log|X(k)| + j\varphi_x(k) \tag{2.2}$$

In many applications it is sufficient to use the real cepstrum [VHH98], so only the real part $\hat{X}_R$ of $\hat{X}(k)$ is used.

$$\hat{X}_R(k) = log|X(k)| \tag{2.3}$$

The advantage of taking the logarithm of the spectrum lies in the fact that all the multiplications are converted into additions. Assuming the spectral magnitude of a speech signal $X(e^{j\omega})$ to be a product of the magnitude of the excitation signal $V(e^{j\omega})$ and the filter frequency response $H(e^{j\omega})$ in the frequency domain

$$|X(e^{j\omega})| = |V(e^{j\omega})| \times |H(e^{j\omega})|, \tag{2.4}$$

then applying the logarithm yields

$$log|X(e^{j\omega})| = log|V(e^{j\omega})| + log|H(e^{j\omega})| \tag{2.5}$$

This sum is mathematically easier to handle than a product because the signal stays linear; a sum of two linear systems is always linear, which is not true for the product. In linear signals, additive terms can be separated, each term can be modified by a function separately, and the results can be summed up again (principle of superposition). Furthermore, the log has the nice property of attaching more importance to lower frequencies, which goes well with our hearing system. To evaluate the rate of fluctuations of the spectrum, another Fourier transformation is used to yield the cepstral coefficients $c_x(m)$ of the signal x. Normally, the inverse Fourier transform (IFFT) is used that differs from the FFT only by a sign change and the factor $\frac{1}{N}$:

$$c_x(m) = \frac{1}{N}\sum_{k=0}^{N-1} \hat{X}_R(k)W_N^{-km} \tag{2.6}$$

As the slow variations of log $|X(e^{j\omega)}|$ are assigned to the filter component $|H(e^{j\omega})|$ and the rapid ones to the source $|V(e^{j\omega})|$, one can simply separate them by using a "highpass window" and a "lowpass window" respectively. This is called "liftering" in analogy to the filtering operation in the frequency domain . To define the cut-off "quefrency" (from cepstral frequency) between source and filter, the periodicity of the cepstrum can be analyzed. Due to the fact that the quefrency is a time variable, the cepstrum shows periodicities whenever the excitation is periodic. Everything below the minimal periodic quefrency can be assigned to the spectral envelope [Zoe02].

In speech *synthesis* we want to reconstruct the spectral envelope given the cepstral coefficients $c(m)$. This is done by reverting the steps explained in Fig. 2.4. It has to be stated that the real cepstrum is not invertible, i.e. $x[n]$ cannot be recovered from $c_x[n]$, because the real cepstrum is calculated only from the magnitude $|X(k)|$ of the Fourier transform. Only assuming a zero-phase, $x[n]$ can be calculated from $c_x[n]$. Taking the FFT, the exponential and the IFFT of Eq. 2.6 yields

$$x(n) = IFFT[exp\{FFT[c_x(m)]\}] \tag{2.7}$$

To describe the frequency response of the spectral envelope $H(e^{j\omega})$ in terms of the cepstral representation $c_h(m)$ we calculate

$$H(e^{j\omega}) = exp \sum_{m=0}^{M} c_h(m) W_N^{km} \tag{2.8}$$

This representation of the spectral envelope's filter response $H(e^{j\omega})$ is referred to as "the exponential model" [TZ09a]. It cannot be directly modeled by a filter, as it is not a rational function, and hence has to be approximated (see 2.2.1).

2.2 Vocoding approaches

The vocoder synthesizes speech by filtering an excitation signal with a vocal tract filter impulse response as shown in figure 2.2. The most simple approach uses white noise for the excitation of unvoiced segments and a periodic pulse train for voiced segments. In reality, voiced segments also consist of a noisy part that is not modeled in the simple approach. Replacing noise-like energy in the original spectrum with periodic energy in the synthetic spectrum leads to a perception of "buzziness“, which is the main vocoding problem.

2.2.1 Cepstral vocoding using the LMA/MLSA filter

The spectrum of a speech signal is calculated from the cepstral coefficients by solving Eq. 2.8. Nevertheless, it is difficult to implement an ideal filter having this exact transfer function, because it is not rational. The log magnitude approximation (LMA) filter approximates Eq. 2.8 using an infinite impulse response (IIR) filter design [TKI95a]. However, the spectrum is not reconstructed perfectly and there is hearable degradation in speech quality because of this approximation.

A Mel log spectral approximation (MLSA) filter was proposed 1983 by Imai et al. [Ima83] and has been frequently applied in speech synthesis, when using a Mel scale frequency warped spectrum.

2.2.2 STRAIGHT vocoding

The STRAIGHT *(Speech Transformation and Representation using Adaptive Interpolation of weiGHTed spectrum)* is a tool for manipulating, analyzing and synthesizing voices, based on the source-filter model [Kaw97, KEF01, Kaw06]. It is very popular along speech synthesis research groups and has gained a lot of awards. E.g., a STRAIGHT based TTS system won the first place in the Blizzard challenge reported at INTERSPEECH 2005 [ZT05]. Basically, STRAIGHT is a channel vocoder, that represents the signal as a source signal and a frequency band dependent filter. Three parameters are used to represent the speech signal:

- the fundamental frequency (F0) with voicing information $f_0(t)$:
- STRAIGHT spectrogram $P(\omega, t)$
- aperiodicity map $A(\omega, t)$

The STRAIGHT parameters

Mixed excitation STRAIGHT uses mixed excitation consisting of a weighted sum of a pulse train with phase manipulation and Gaussian noise (Fig. 2.5).

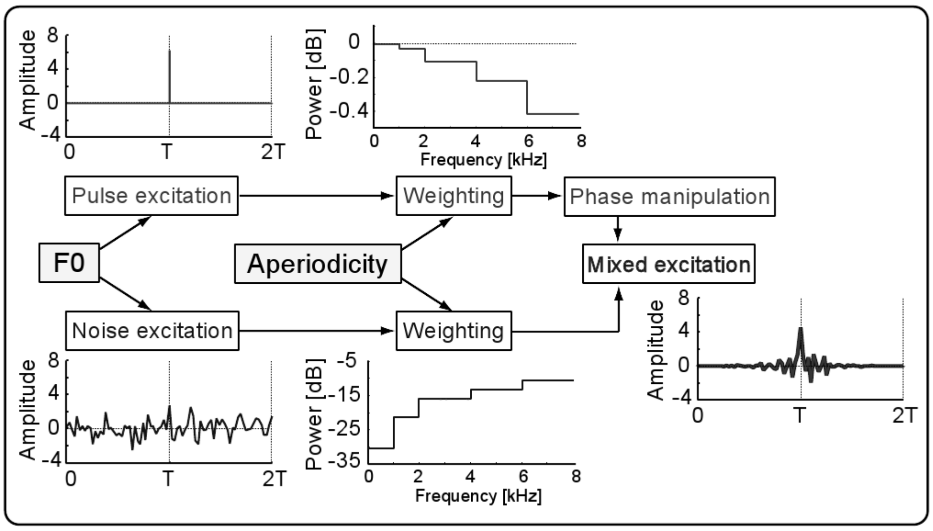

Figure 2.5: Excitation signal generation in STRAIGHT [TZ09a]

Jitter (the time difference between two adjacent pairs of impulses) and Shimmer (the amplitude difference between two adjacent impulses) are also implicitly implemented. The frequency domain aperiodicity measure controls the spectral shape of the noise and a time domain concentration measure defines the temporal envelope of the noise [KEF01]. Randomly chosen group delays in higher frequency regions are also used to reduce the perceived "buzziness" of the signal.

STRAIGHT spectrum When looking at voiced speech, the periodic excitation of a set of resonators by a pulse train can also be seen as a sampling operation. A periodic signal $s(t) = s(t + n\tau_0)$ with a fundamental period τ_0 provides information about the produced spectrum for every τ_0 in the time domain and every $f_0 = \frac{1}{\tau_0}$ in the frequency domain [Kaw97]. As the transfer function of the vocal tract is not band-limited, aliasing errors might be introduced when the sampling frequency (i.e., the pulse train frequency) is less than two times the highest frequency of the vocal tract transfer function.

Spectral distortions can also be caused by fundamental frequency estimation errors. Temporal periodicity occurs due to phase interference between adjacent components [Kaw06]. The periodic interferences in both the time and frequency domain can be seen in the left panel of Fig. 2.6.

STRAIGHT uses two steps to solve this problem.

1. Remove temporal periodicity by using a complementary set of windows
2. Inverse filtering and spectral smoothing in a spline space

The resulting STRAIGHT spectrum is shown in the right panel of Fig. 2.6.

As the STRAIGHT spectrum claims to be F0-independent, higher cepstral coefficients can be calculated without having the problem of interferences between cepstrum and F0 in the higher coefficients. Using the STRAIGHT-cepstrum, 39 cepstral coefficients can be used instead of 13 when using the conventional cepstrum.

Aperiodicity map Speech sounds are not strictly periodic. Excitation and filter component are fluctuating in frequency and amplitude, due to the movements of the articulators and to the varying fundamental frequency. The energy of inharmonic frequencies normalized

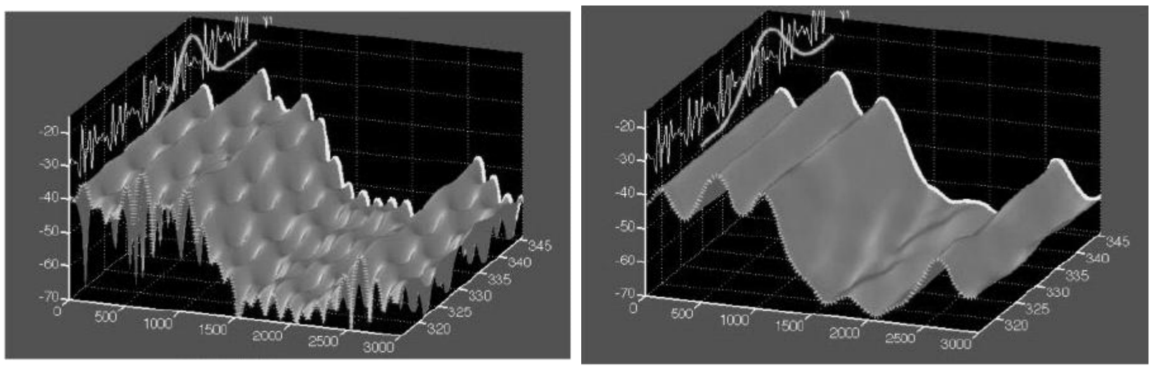

Figure 2.6: Estimated spectra of Japanese vowel /a/. The three-dimensional plot has a frequency axis (left to right in Hz), a time axis (front to back in ms) and a relative level axis (vertical in dB). Left panel shows the spectrogram calculated using an isometric Gaussian window. The right panel shows the STRAIGHT spectrogram. (Taken from [Kaw06])

by the total energy provides a good measure of aperiodicity. In order to reliably model the aperiodicity, it is averaged on five frequency sub-bands (i.e.,0-1, 1-2, 2-4, 4-6, 6-8 KHz).

Tandem-STRAIGHT

Tandem-STRAIGHT [KMT+08] was released in 2007 as a reformulation of STRAIGHT, which is based on nonlinear transformations and many coefficients for tuning, making a theoretical analysis of the system intractable. The new approach eliminates ad-hoc parameter tuning and the heavy demand on computational power, from which STRAIGHT has suffered in the past. It also introduces a new power-spectrum estimation method called TANDEM, that eliminates periodic temporal fluctuations while preserving the quality of the current STRAIGHT version.

Chapter 3

HMM-Based Speech Synthesis

The HMM-based synthesis is among the most preferred methods of all the statistical parametric synthesis techniques, proved by a high number of entries for the Blizzard Challenge 2009 [OWT09].

3.1 System architecture

3.1.1 Overview

Fig. 3.1 shows the basic structure of a HMM-based speech synthesis system. The speech signal is split into excitation and spectral parameters according to the source-filter model (2.1.1). Due to a transcription code provided by a label file, the speech data is further divided into phonetic units. Each phonetic unit is statistically modeled by a HMM.

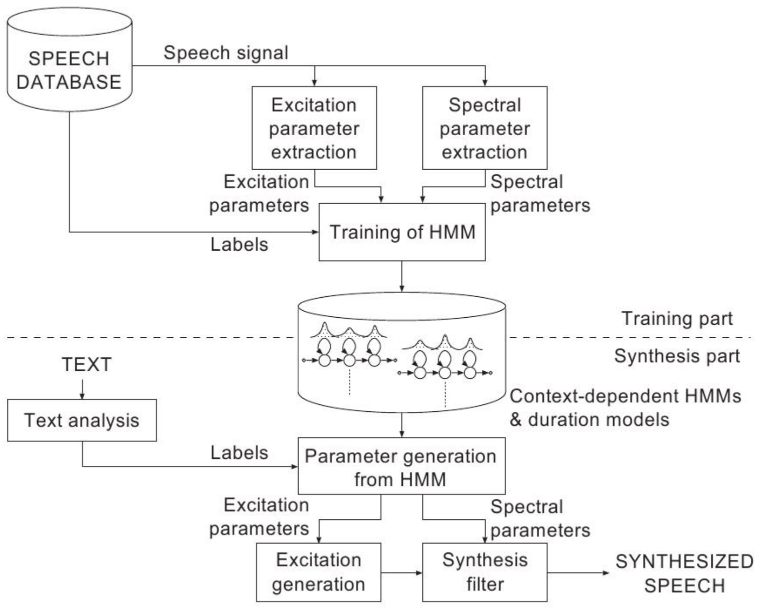

Figure 3.1: HMM-based speech synthesis system (HTS) [ZON+09]

The label file contains information about the sequence of the phonetic units and prosodic features, such as rhythm, stress, intonation and contextual factors (full-context modeling, [Yam06]), e.g.:

- the current and the two preceding and succeeding phonemes
- the number of morae in a sentence (the mora determines the weight of a syllable, its stress and timing)

- the position of the breath group in a sentence (a breath group is a part of an utterance that is spoken within a single expiration)
- the number of morae in the preceding, current and succeeding breath group
- etc.

All these features are listed in the label file resulting in complex model names like
l^f-ax+p=aa@1_ 2/A:1_0_3/B:0-0-2@1-2&5-6#3-4$2-3!1-1 [...]

In the synthesis step, the waveform of a speech segment is generated from the trained models. A textanalyzer generates a label file from the text, which lists the models that are needed to synthesize the phrase. First, it has to be checked if all the models of the target utterance are available in the training database. If not, the unseen models are built according to the question trees (3.2), that have been set up in the training step. When all the models are available, a composite HMM is created by simply concatenating the full-context models in respect to the label file. The state sequence is determined maximizing the output probabilities of the state duration models. Next, the most likely parameter sequences of spectrum and excitation can be calculated from the HMMs given the state sequence. Finally, the excitation and cepstral parameters are fed into the synthesis filter to generate the speech waveform.

3.1.2 Acoustic modeling

An appropriate parameter representation of the speech signal has to be defined. This representation consists of several features, e.g., the fundamental frequency F0 and the spectrum represented by the cepstral coefficients. The feature vectors consist of the "static" features and their delta and delta-delta values, referred to as "dynamic" features. They are organized in stochastically independent data streams, as seen in Fig. 3.2. The spectral parameters and their dynamic features are assigned to one stream, whereas the $\log F0$ and its dynamic features belong to separate streams.

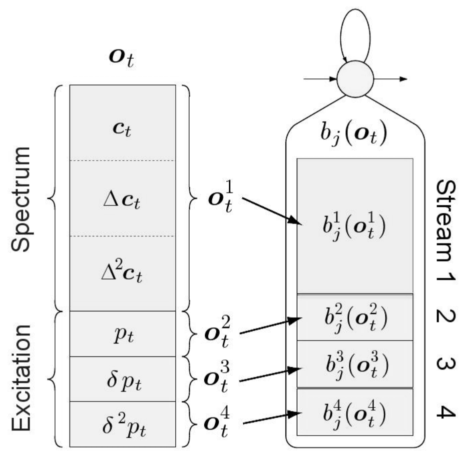

Figure 3.2: Multi-stream HMM structure [TZ09b]

The HMMs are mostly left-to-right models using 5-7 emitting states without skip. The minimum state duration corresponds to the frameshift. When using 7 emitting states and a frameshift of $5ms$, the minimum duration of a phoneme is $7 \times 5ms = 35ms$. This can be a problem when very short phonemes (e.g., plosives) are to be generated.

3.1.3 Duration modeling

In conventional HMMs, the state duration is only defined by the self-transition probability a_{ii}. The probability of being n frames in state i becomes

$$p_i(n) = (a_{ii})^{n-1}(1 - a_{ii}) \tag{3.1}$$

This exponential state duration probability does not model the state duration satisfactorily [Yam06]. To gain better control over the temporal aspect of the generated speech, Hidden Semi-Markov Models (HSMMs) [Mur02] are used in the current HTS system. HSMMs are HMMs that use an explicit duration model to represent the state duration (Fig. 3.3).

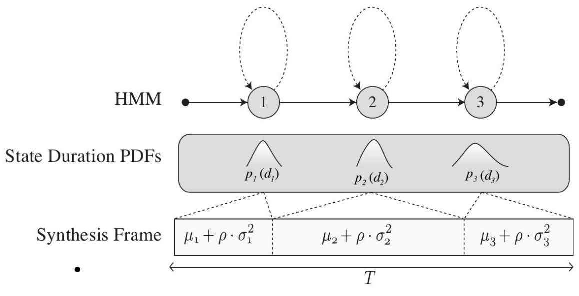
HMM
1
2
3
State Duration PDFs
$p_1(d_1)$
$p_2(d_2)$
$p_3(d_3)$
Synthesis Frame
$\mu_1+\rho\cdot\sigma_1^2$
$\mu_2+\rho\cdot\sigma_2^2$
$\mu_3+\rho\cdot\sigma_3^2$
T

Figure 3.3: Duration synthesis [Yam06]

3.1.4 F0 modeling

As the F0 is not defined in unvoiced regions, it cannot be modeled by conventional HMMs. Therefore, multi-space distribution HMMs (MSD-HMM) have been introduced [TMMK02] to model the varying dimensions of F0, which is one-dimensional for voiced regions and zero-dimensional for unvoiced regions.

3.2 Context clustering

The statistical properties of a phoneme vary a lot, depending on contextual factors. Contextual factors include prosodic and linguistic features, such as the position of the phoneme, the stress, the number of morae or the position of breath groups. Ideally, all the different realizations of speech units would be available in a database, and in the synthesis step one would only have to select the compatible ones. As it is practically not possible to record all possible combinations of contextual factors, it is necessary to group similar instances together and share properties within those groups. This is called ”parameter tying”. To find out which instances to group together, a decision-tree-based context clustering algorithm [Yam06] or a data-driven clustering approach is applied

In decision-tree-based context clustering, each node of the tree has a context, related question that can be answered with “yes” or “no” (Fig. 3.4), e.g., “R-silence ?” (”is the previous phoneme a silence ?”). Several questions are concatenated until the final leaf node is reached. Each leaf node represents a cluster of instances that have some context features in common - as they had the same answers to the same questions - and is assigned to a state output distribution that is shared among all the instances belonging to this leaf node. The question at each node is chosen to maximize the local likelihood of the training data. Assuming we would only have one cluster for the whole training corpus, the whole corpus would be modeled by one single PDF. The likelihood of generating the training data with our

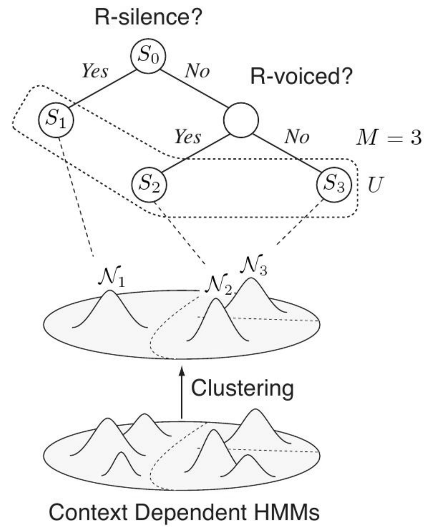

Figure 3.4: An example of a decision tree [Yam06]

given PDF will be rather small. Now the training data is divided into two clusters, each one modeled by a single Gaussian. Cluster C_1 is modeled by PDF_1 and cluster C_2 is modeled by PDF_2. Then the likelihood of generating the training data assigned to C_1 will be increased, because the more specific PDF_1 is used to generate C_1. The questions are chosen in a way such that the log likelihood increase is maximized, when dividing the data into clusters. The clustering process is repeated until the increase in log likelihood or the occupancy of the leaf node falls beyond a certain threshhold.

Beside the decision-tree based clustering, there is also a data-driven approach for clustering of the speech corpus. Starting with an own cluster for each model, the number of clusters is reduced by merging clusters with the smallest distance of the model parameters. Data-driven clustering has the disadvantage that unseen models are difficult to relate to seen model parameters. Normally, the model parameters of a model with lower accuracy is assigned to the unseen model that has the same central phoneme.

The parameters of a cluster are obtained by averaging over all the instances assigned to that cluster. This is illustrated in Fig. 3.5 showing all the realizations of a parameter curve belonging to one leaf node of a decision tree and the resulting average. The magnitude and center-frequency of the formants are not clearly preserved, resulting in muffled-sounding speech due to the over-smoothing.

One possible solution to this problem is to increase the model complexity and accuracy augmenting the number of leaf nodes. Using preciser models, the clusters can be made smaller and more specific.

3.3 Parameter generation

In this section, it is described how to find the optimal parameter sequence $\mathbf{O}$ of length T, given a trained model λ. We are interested in the speech parameter vector sequence

$$\mathbf{O} = (\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_T) \tag{3.2}$$

that maximizes $P(\mathbf{O}|\lambda, T)$:

$$\hat{\mathbf{O}} = \arg\max_{\mathbf{O}} P(\mathbf{O}|\lambda, T) \tag{3.3}$$

This implies summing up all the emission probabilities of all possible state sequences and mixture sequences $\mathbf{Q} = (q_1, i_1), (q_2, i_2), ...(q_T, i_T)$ with q_t the state and i_t the mixture assigned to time t (see B.6). As the computational cost increases exponentially with T, the parameter generation is restricted to the most likely state and mixture sequence $\hat{\mathbf{Q}}$ determined by the Viterbi algorithm:

$$\hat{\mathbf{O}} \simeq \arg\max_{\mathbf{O}} \max_{\mathbf{Q}} P(\mathbf{O}, \mathbf{Q}|\lambda, T) = \arg\max_{\mathbf{O}} P(\mathbf{O}|\hat{\mathbf{Q}}, \lambda) \tag{3.4}$$

Given the state q_t and mixture i_t at a certain time t, the probability of emitting $\mathbf{o}_t$ can be written as (see B.1):

$$P(\mathbf{o}_t|\lambda, (q_t, i_t)) = w_{q_t,i_t} N(\mathbf{o}_t; \boldsymbol{\mu}_{q_t,i_t}, \boldsymbol{\Sigma}_{q_t,i_t}) \tag{3.5}$$

Using B.2 the log of $P(\mathbf{o}_t|\lambda, (q_t, i_t))$ can be written as

$$\log P(\mathbf{o}_t|\lambda, (q_t, i_t)) = \log\left[\frac{w_{q_t,i_t}}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}_{q_t,i_t}|}}\right] + \left[-\frac{1}{2}(\mathbf{o}_t - \boldsymbol{\mu}_{q_t,i_t})^\top \boldsymbol{\Sigma}^{-1}_{q_t,i_t} (\mathbf{o}_t - \boldsymbol{\mu}_{q_t,i_t})\right] \tag{3.6}$$

Using the most likely state and mixture sequence $\hat{\mathbf{Q}}$ it can be written

$$\begin{aligned} \log P(\mathbf{O}|\hat{\mathbf{Q}}, \lambda) &= \sum_{t=1}^{T} \log P(\mathbf{o}_t|\lambda, (q_t, i_t)) = K - \frac{1}{2}\sum_{t=1}^{T}[(\mathbf{o}_t - \boldsymbol{\mu}_{q_t,i_t})^\top \boldsymbol{\Sigma}^{-1}_{q_t,i_t} (\mathbf{o}_t - \boldsymbol{\mu}_{q_t,i_t})] \\ &= K - \frac{1}{2}(\mathbf{O} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{O} - \boldsymbol{\mu}) = K' - \frac{1}{2}\mathbf{O}^\top \boldsymbol{\Sigma}^{-1}\mathbf{O} + \mathbf{O}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \end{aligned} \tag{3.7}$$

where

- K and K' are constants that do not depend on $\mathbf{o}_t$
- D is the dimensionality of the static feature vector
- $\boldsymbol{\mu} = [\boldsymbol{\mu}^\top_{q_1,i_1}, \boldsymbol{\mu}^\top_{q_2,i_2}, ..., \boldsymbol{\mu}^\top_{q_T,i_T}]^\top$ is a $3D \times T$ matrix of one static and two dynamic mean vectors, associated with state q_t and mixture i_t.
- $\boldsymbol{\Sigma}^{-1} = diag[\boldsymbol{\Sigma}^{-1}_{q_1,i_1}, \boldsymbol{\Sigma}^{-1}_{q_2,i_2}, ..., \boldsymbol{\Sigma}^{-1}_{q_T,i_T}]$ is a $3D \times (3DT)$ matrix of static and dynamic covariance matrices, associated with state q_t and mixture i_t. .

In order to maximize $P(\mathbf{O}|\hat{\mathbf{Q}}, \lambda)$ in respect to $\mathbf{O}$, the first derivative is set to zero. If the observation vector $\mathbf{O}$ only includes static features, we can write $\mathbf{O} = \mathbf{C}$ with

$$\mathbf{C} = (\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_T) \tag{3.8}$$

c_t is the static feature vector at time t. Derivating 3.6 in respect to $\mathbf{C}$ yields

$$\frac{\partial(\log P(\mathbf{O}|\hat{\mathbf{Q}}, \lambda))}{\partial \mathbf{C}} = \frac{\partial(\log P(\mathbf{C}|\hat{\mathbf{Q}}, \lambda))}{\partial \mathbf{C}} = -\boldsymbol{\Sigma}^{-1}\mathbf{C} + \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \overset{!}{=} 0 \tag{3.9}$$

For $\mathbf{C} = \boldsymbol{\mu}$ the equation is true and $\log P(\mathbf{O}|\hat{\mathbf{Q}}, \lambda)$ has a maximum. But problems occur at the state borders, when the parameter vector jumps from the mean vector of the current state to the mean vector of the next state. These jumps lead to discontinuities in the generated parameter curve and causes artifacts.

When also taking into account dynamic features, these problems can be solved [TKI95b]. The dynamic features are obtained by a linear combination of the static feature vectors of

several frames around the current frame.
By setting $\Delta^0\mathbf{c}_t = \mathbf{c}_t$, $\Delta^1\mathbf{c}_t = \Delta\mathbf{c}_t$, and $\Delta^2\mathbf{c}_t = \Delta\Delta\mathbf{c}_t$, the general form $\Delta^{(n)}\mathbf{c}_t$ is defined as

$$\Delta^{(n)}\mathbf{c}_t = \sum_{\tau=-L}^{L} w_{t+\tau}^{(n)}\mathbf{c}_t \qquad n = 0, 1, 2 \tag{3.10}$$

The new observation vector $\mathbf{o}_t$ now is defined as

$$\mathbf{o}_t = [\mathbf{c}_t, \Delta\mathbf{c}_t, \Delta^2\mathbf{c}_t] \tag{3.11}$$

with the relation

$$\mathbf{O} = \mathbf{W}\mathbf{C}, \tag{3.12}$$

where $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_T]^\top$ is a $3DT \times 3DT$ window matrix that extends the static feature vector sequence $\mathbf{C}$ to the observation vector sequence $\mathbf{O}$, consisting of static and dynamic features. The more interested reader is referred to [Yam06].

Combining 3.7 and 3.12 we get

$$\log P(\mathbf{O}|\hat{\mathbf{Q}}, \lambda) = \log P(\mathbf{W}\mathbf{C}|\hat{\mathbf{Q}}, \lambda) = K' - \frac{1}{2}(\mathbf{W}\mathbf{C})^\top\mathbf{\Sigma}^{-1}\mathbf{W}\mathbf{C} + (\mathbf{W}\mathbf{C})^\top\mathbf{\Sigma}^{-1}\boldsymbol{\mu} \tag{3.13}$$

Computing the first derivative and setting it equal to zero yields:

$$\frac{\partial(\log P(\mathbf{W}\mathbf{C}|\hat{\mathbf{Q}}, \lambda))}{\partial\mathbf{C}} = (\mathbf{W}^\top\mathbf{\Sigma}^{-1}\mathbf{W})\mathbf{C} - \mathbf{W}^\top\mathbf{\Sigma}^{-1}\boldsymbol{\mu} \overset{!}{=} 0 \tag{3.14}$$

This equation is very often written as [Yam06, TKI95b]

$$\mathbf{R}\mathbf{C} = \mathbf{r} \tag{3.15}$$

where

$$\mathbf{R} = \mathbf{W}^\top\mathbf{\Sigma}^{-1}\mathbf{W} \tag{3.16}$$

is a $TD \times TD$ matrix and

$$\mathbf{r} = \mathbf{W}^\top\mathbf{\Sigma}^{-1}\boldsymbol{\mu} \tag{3.17}$$

is a TD-dimensional vector. To directly solve Eq. 3.16 using only the most likely state and mixture sequence $\hat{\mathbf{Q}}$, $O(T^2D^3)$ operations are needed. When $\mathbf{\Sigma}$, $\Delta\mathbf{\Sigma}$ and $\Delta^2\mathbf{\Sigma}$ are diagonal, the complexity of the algorithm becomes $O(T^2D)$ [TKI95b].

There also exist other approaches to find $P(\mathbf{O}|\lambda)$ without the constraint of a given most likely state sequence [TYM$^+$00]. In this thesis I concentrate on the mostly used one algorithm, as described above.

3.4 Known drawbacks

Compared to unit-selection approaches, HMM-based speech synthesis still sounds less natural. The following main factors can be identified to degrade the quality of a HMM-based synthesis system [BZT07] [ZTB09]:

3.4.1 Vocoder

The generated speech sounds *buzzy* since a vocoder technique is used. In the older approaches [YMKK99], the excitation of voiced sounds were modeled by a simple periodic pulse-train. In the past years, a lot of different high-quality vocoders have been proposed to alleviate this problem, such as STRAIGHT [Kaw06], the harmonic plus noise model (HNM) [Sty00], the pitch synchronous residual codebook [TATG09] or the SVLN model [LDR10].

3.4.2 Over-smoothing

When using HMMs for modeling speech, many assumptions are made that do not hold for real speech. E.g., constant statistics within an HMM state or statistical independence of the features are assumed. To perfectly model real speech data, a high number of parameters would have to be estimated, requiring huge training databases. A trade-off has to be found between the accuracy of the model and the number of parameters that have to be estimated. Poor acoustical modeling leads to over-smoothing.

Statistical averaging in the modeling step improves the robustness against data sparseness. However this averaging very often leads to over-smoothed parameter trajectories and muffled-sounding speech. Over-smoothing can be categorized into two types: over-smoothing in the time domain and over-smoothing in the frequency domain [YZTW07, MZW08].

Over-smoothing in the frequency domain is the main factor that degrades the quality of synthesized speech and it is generally caused by training algorithm accuracy problems [MZW08]. The most used training algorithm for HMMs is the maximum likelihood estimation (MLE). This can be considered as an inconsistency between the training algorithm of the HMMs and its application [WW06]: The aim of HMM-based speech synthesis is to create speech vectors that are as close as possible to natural speech. The HMM training based on the MLE on the other hand, tries to find model parameters that fit best to the average of a set of speech vectors. This is adequate for speech recognition purposes, as we only want to find the right cluster that belongs to one of many realizations of a phonetic unit (e.g., a word). But it will obviously fail in synthesis, where we want to generate many different realizations out of our few models (providing only the mean value and the variance of a feature). Over-smoothing in the frequency domain is shown in Fig.3.5, where all the realizations of the spectrum in one leaf node and the resulting average are plotted. The formants of the average signal are blurred and have a lower magnitude. Increasing the number of leaf nodes in the decision tree clustering step, increases the accuracy and leads to more expressive formants, but also increases the number of parameters that have to be estimated from the database.

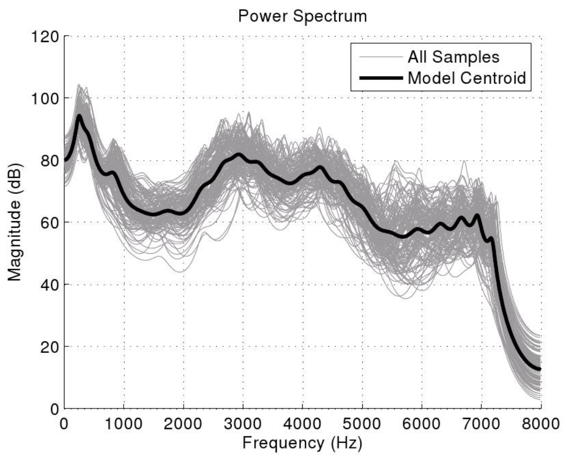

Figure 3.5: Over-smoothing problem in a typical decision tree leaf-node [YQS09]

Wu and Wang proposed a minimum generation error (MGE) training, which estimates the model parameter to minimize an error measure between training data and the generated speech parameters [WW06]. The over-smoothing in the frequency domain can be alleviated by the MGE leading to an improvement of the speech quality.

Methods like the Global Variance based parameter generation sharpen the formants, leading to a more intelligible and natural-sounding speech [TT07, TY09], 5.1.1.

Another approach uses real speech data from the training corpus to generate parameters. Drugman et al. use a pitch synchronous residual codebook [TATG09] to model the excitation, and Jian Yu et al. developed a HMM-based TTS system using discrete HMMs, based on speech vectors from a codebook [YZTW07].

Over-smoothing in the time domain (Fig. 5.1) causes parameter sequences that are too smooth to carry sufficient detailed information. This is often the case when using a low number of states and mixtures. The results are long state durations and poor temporal resolution, because each state is only modeled by one probability distribution. Furthermore, constraints between static and dynamic features are defined in the feature extraction and in the parameter generation algorithm, but are not taken into account during the training process. Once calculated, the dynamic features are trained independently from the static ones and are treated like the other static features. This inconsistency was fixed by imposing a trajectory model with explicit relationships between static and dynamic features into the HMM training [TZK03, ZTK04, HKT06].

A new method proposed in this thesis is the roughness based parameter generation. It increases the temporal details of a generated curve and hence reduces the over-smoothing in the time domain (see 5.2).

Chapter 4

Description of the HTK-based Speech Synthesis System (HTS)

The HMM-based Speech Synthesis System (HTS) was released in 2002 by the HTS working group to provide a research and development platform for statistical parametric speech synthesis. Since then it has regularly been updated, according to the newest developments and has grown in popularity over the years. It is released as a free patch to the HTK toolkit [YEG$^+$09]. The current version contains two demo scripts to construct speaker-dependent systems (English and Japanese) and a demo script to train a speaker-adaptation system (English). The following explanations are based on the speaker-dependent HTS-demo CMU-ARCTIC-SLT STRAIGHT 2.1.1 (English) that was released on May 14, 2010.

The basic structure of HTS is shown in Fig. 3.1; details are explained in the following section.

Forced alignment In order to compare natural and generated data, the generated parameter sequences are force-aligned with the natural data. The most likely state duration sequence of the natural parameter curves was extracted using the Viterbi algorithm (HVite). This state duration sequence was fed into the parameter generation algorithm.

4.1 Training Corpus and Model topology

4.1.1 The voice database

The English demo script uses the CMU ARCTIC databases [KB03] for training the models. They contain 1132 utterances from out-of-copyright texts, spoken by a male US English speaker. The database was fully phonetically labelled by the CMU Sphinx using the FestVox based labelling scripts. The utterances were recorded with 16 kHz sampling rate and 16 bit resolution. The phoneme set (of the basic English language) has 41 elements consisting of the 39 phonemes from CMUDICT, plus the reduced vowel schwa /AX/ and the pause symbol /PAU/. The number of possible diphones is 41 x 41 = 1.680 and the number of possible triphones is 41 x 41 x 41 = 68.921, but not all possible diphones/triphones appear in real speech. The triphone coverage of the database is said to be 13,7 % of all possible triphones .

The database provides one audio file (.raw) and one utterance file (.utt) per training-phrase. Beyond the transcription of the audio file, the utterance file also contains segmental and suprasegmental information, like the position of the monophones and diphones within the phrase, the fundamental frequency and the accentuation.

20 sentences from the training corpus are put in a test pool and therefore are not available as training data. Using full-context models (3.1.1), 38085 different models occur in the training database. To test the speech synthesis system, 20 sentences from the test pool and 20 sentences of "Alice's Adventures in Wonderland" - written 1865 by the English author Charles Lutwidge Dodgson - are to be synthesized. These target sentences contain 2824

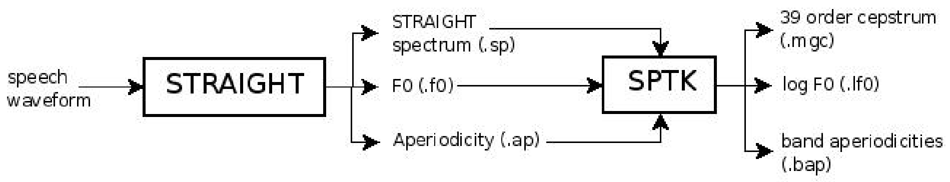

Figure 4.1: Feature extraction; SPTK [IK09] is the Speech signal processing toolkit

different full-context models. Four models of the target sentences also appear in the training database, the remaining 2820 are unseen.

4.1.2 Speech features

The demo script first extracts STRAIGHT spectrum, F0 and aperiodicities using STRAIGHT v40 007c. The spectrum is converted into a generalized 39 order cepstrum, F0 is converted into $\log F0$ and the aperiodicity is converted into five band aperiodicities (Fig. 4.1).

Vocal tract modeling The conventional cepstral feature vector, as derived from the Short-Time Fourier Transform (STFT) of a signal, is usually restricted to an order of about 13, since the higher order coefficients are distorted by the fundamental frequency. STRAIGHT (2.2.2) carries out a F0-adaptive spectral analysis, that removes all effects of the fundamental frequency from the shortterm spectrum. Using the STRAIGHT spectrum instead of the STFT spectrum, the order of the cepstral feature vector can be increased from 13 to 39 [IMN+02].

Frequency warping The demo scripts allow frequency warping using the Mel scale, yielding Mel-frequency cepstral coefficients (MFCCs). The frequency bands represented by the MFCCs are equally spaced on the Mel scale, in contrast to the conventional cepstral features that are equally spaced using a linear frequency scale. The Mel scale approximates the human auditory system better than a linear frequency scale.

4.1.3 Model topology

The models hold the following properties:

- full-context models (see 3.1.1)
- 7 emitting states
- left-to-right models without skip
- HSMM with explicit duration models (see 3.1.3)
- the observation vector is divided into five stochastically independent data streams (see 3.2): 1-spectrum, 2-$\log F0$, 3-$\Delta \log F0$, 4-$\Delta\Delta \log F0$ and 5-bap
- the covariance matrix is diagonal, assuming there is no correlation between the features

4.2 Training

In the training stage, the models are trained using the features that have been extracted in the previous section. An overview of the training process is given in Fig. 4.2

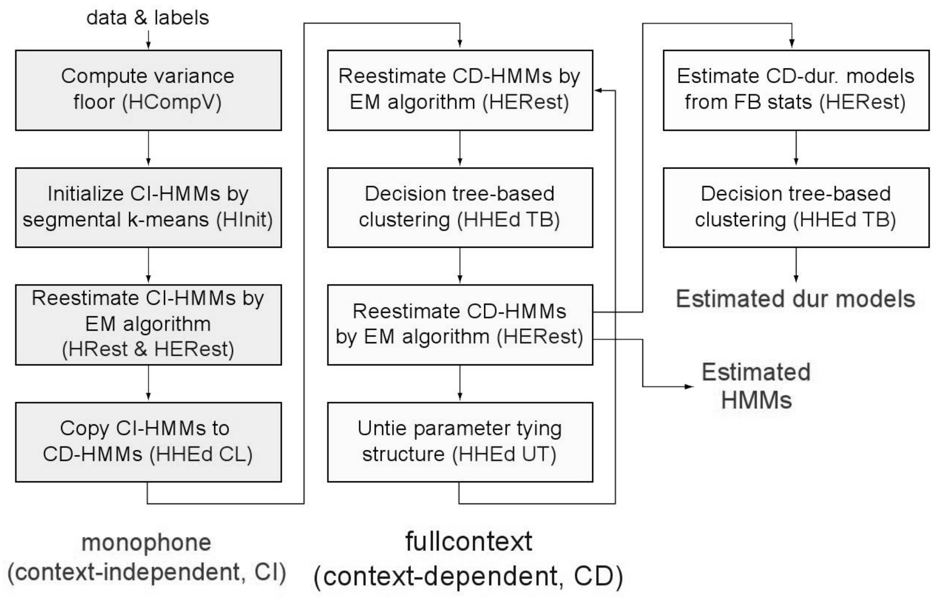
data & labels
Compute variance floor (HCompV)
Initialize CI-HMMs by segmental k-means (HInit)
Reestimate CI-HMMs by EM algorithm (HRest & HERest)
Copy CI-HMMs to CD-HMMs (HHEd CL)
Reestimate CD-HMMs by EM algorithm (HERest)
Decision tree-based clustering (HHEd TB)
Reestimate CD-HMMs by EM algorithm (HERest)
Untie parameter tying structure (HHEd UT)
Estimate CD-dur. models from FB stats (HERest)
Decision tree-based clustering (HHEd TB)
Estimated dur models
Estimated HMMs
monophone (context-independent, CI)
fullcontext (context-dependent, CD)

Figure 4.2: Training process of the HTS-demo [TZ09a]

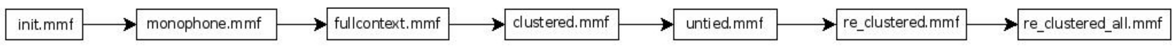

Figure 4.3: Sequence of the generated model files in the demo script

Compute variance floor (HCompV)

Defining the exact borders of the phonemes and states within a training sentence is called segmentation. Starting with states having a uniform length (flat start), the likelihood - of a parameter sequence being generated by the current model parameters (see B.1) - is maximized by changing the state borders iteratively. To do this, an initialization of the models is needed: For each stream, a global variance and a mean value are calculated using the whole training corpus. Then the same variance floor is assigned to each of the segments. The segmentation performance could be improved, if manually labeled data was available as an initial model, which is not feasible in most of the cases.

When training large model sets from limited data, setting a floor is often necessary to prevent variances being badly underestimated through lack of data [YEG$^+$09]. When the same global variance is assigned to all the segments, the segments, that are not seen in the training data and therefore are not subsequently re-estimated, remain with the global variance.

Initialize context-independent HMMs by segmental k-means (HInit)

As the CMU ARCTIC database already provides a label file with the phoneme borders, this information can be used as an initial model. Within the given phoneme borders only the state borders have yet to be determined. With the initial HMM parameters the most likely state sequence of given training utterance can be determined using the Viterbi algorithm (see B.1). Using this state sequence, model parameters can be re-estimated and a new most likely state sequence can be found. This is done iteratively until no further increase in likelihood is obtained.

If the states are modeled by multiple mixture components, each training vector emitted by a state is associated with the mixture component with the highest likelihood. The number of vectors associated with each component can then be used to estimate the mixture weights. The initial segmentation is obtained by clustering each state with a k-means clustering algorithm (k denotes the number of mixture components).

Re-estimate context-independent HMMs by the EM algorithm (HRest & HERest)

HRest also updates the HMM parameters like HInit, but instead of Viterbi training it uses Baum-Welch re-estimation. Viterbi training makes a hard decision about which state has generated which observation vector. In contrast, the Baum-Welch algorithm makes a soft decision - it provides the probability of being in each state at a certain time frame. In real speech there are no hard boundaries between the phonemes and using soft decisions often yields better results [YEG$^+$09].

HERest simultaneously updates all of the HMMs in a system using all of the training data in one single iteration (embedded re-estimation) applying the Baum-Welch algorithm. All the phonemes are joined together in the right order (given by the label file) to create a single composite HMM per training utterance. HERest estimates the new model parameter and writes out an updated modelfile.

Copy context-independent HMMs to context-dependent HMMs (HHEd CL)

Until now, the system has only trained monophone models, providing 41 models and their statistics (mean and variances). Now the full-context models are introduced. Each monophone model is assigned to the full-context model (over all the database contains 38085 such full-context models) with the corresponding central phoneme, and the statistics are copied to the full-context model file. At this moment, each full-context model sharing the same central phone has the same model parameters.

Re-estimate context-dependent HMMs by the EM algorithm (HERest)

The parameters of the full-context HMMs are updated. A statistics file shows the occupation count for each state of each model, providing information on how much training material was available for each state.

Decision tree-based clustering (HHEd TB)

At first, states with a low occupation count are removed. Then all possible phonetic questions are loaded. Now those questions have to be selected, that maximize the likelihood when dividing the set into clusters. Finally, each cluster is tied together as a macro and saved to a file with the decision trees. The decision trees are used later to map the unseen models onto the clusters. As spectrum and excitation have different context dependencies, the decision trees are built separately. Tree-based clustering reduces the number of model parameters to about 0.3 %, e.g., 38085 output PDFs of state 6 are reduced to 133 clusters (and 133 PDFs).

Re-estimate clustered context-dependent HMMs by the EM algorithm (HERest)

The parameters of the clustered full-context HMMs are updated.

Untie parameter tying structure (HHEd UT)

The state transition probabilities and the stream weights have been tied over all the states. These tyings are reversed in this step. After untying, re-estimation and tree-based clustering is applied again, to finally yield the estimated HMMs.

4.3 Synthesis

In the synthesis stage, the speech waveform is generated according to the label file and the trained models (Fig. 4.4).

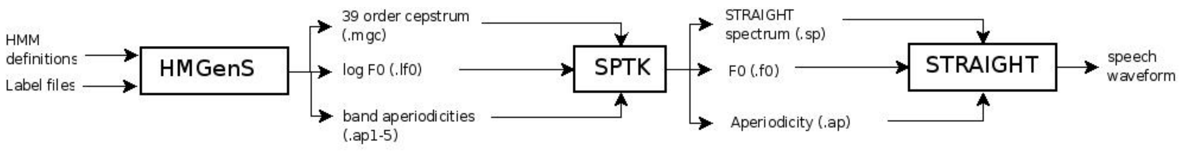

Figure 4.4: Synthesis stage; SPTK is the Speech signal processing toolkit [IK09]

Make unseen models

Unseen models are models that have not been detected in the training database. The database can never cover all possible full-context models. In our example, 2824 models are needed to generate the parameters for the 40 sentences that are to be synthesized. Only 4 models out of these 2824 are seen in the training data, which corresponds to 0.0014%. The remaining 2820 unseen models have to be generated: Using the decision trees from the training part, the unseen models can be mapped onto a cluster of similar linguistic properties. Every unseen model is assigned to the macro of the underlying cluster. E.g., the second state of the cepstrum of the unseen quinphone "n^d-ow+n=z" is mapped onto a cluster containing also the quinphones "iy^g-ow+pau=x", "ae^g-ax+n=iy", "n^d-ax+k=aa", "iy^d-ax+p=y".

4.4 Found problems

4.4.1 Vocoder

STRAIGHT does not provide a perfect representation of a speech signal. The encoding of a waveform into STRAIGHT spectrum, F0 and aperiodicities is a lossy procedure. For this reason, the output of the speech synthesis system cannot sound better than the waveform obtained by natural parameter curves passing through the STRAIGHT filter.

4.4.2 Accuracy of acoustic modeling

In the following section, the amount of training data that is available in relation to the number of estimated model parameters is examined. The *mean occupation count* is a measure indicating how many samples from the training data are available to estimate the parameters of the PDF describing one state. A low occupation count causes models that have no or only few training data associated with. This can lead to poor modeling quality, where clicks and crackles can occur when no suitable PDF for a model can be estimated. In this case, the complexity of the models should be reduced having less parameters to estimate. This can be done by using less states, less mixtures or simpler models, such as diphones instead of triphones. In contrast, when the occupation count is too high, the complexity of the models should be increased to achieve the maximum quality from the database. The ideal occupation count has to be found, facilitating accurate synthesis at one hand and low data sparseness on the other hand.

The number of leaf nodes for the decision tree based state tying is a good parameter to gain control over the modeling accuracy. Using the full-context model definition from 3.1.1, 38085 different models are present in our database. Each model has 7 emitting states, yielding $7 \times 38085 = 266595$ different PDFs for each feature when using one mixture per state. This means 266595 mean and variance values have to be estimated from the training data. Using the training corpus described in 4.1.1 and a frameshift of 5ms, 666634 frames (and feature vectors) are extracted. Estimate the 266595 free parameters from the 666634 feature vectors, a mean occupation count of $666634/266595 \approx 2.5$ is obtained.

In the proposed demo script, these 266595 PDFs are divided into 803 clusters for the cepstrum, 2084 clusters for F0, 950 clusters for the aperiodicities and 505 clusters for the duration using decision tree based state tying. The resulting mean occupation count after the clustering is 830, 320, 701 and 1320 respectively. A perceptional comparison of different leaf node numbers is given in the following audio examples.

4.4.3 Over-smoothing

Over-smoothing in the time- and frequency domain occurs due to the reasons explained in 3.4.2 and can be seen in Fig. 5.1 and 5.3 respectively. The following section describes two approaches to reduce the over-smoothing problem in both time- and frequency domain.

Audio examples

Audio example 01 (CD-track 01)	Natural speech ($16kHz, 16bit$)
Audio example 02 (CD-track 02)	Speech generated with STRAIGHT from natural parameters
Audio example 03 (CD-track 03)	Speech generated using 1943 leaf nodes for spectrum, 6361 for $\log F0$, 2410 for aperiodicities and 1128 for duration (occupation count = 343 / 105 / 277 / 591)
Audio example 04 (CD-track 04)	Speech generated using 803 leaf nodes for spectrum, 2084 for $\log F0$, 950 for aperiodicities and 505 for duration (occupation count = 830 / 320 / 701 / 1320)
Audio example 05 (CD-track 05)	Speech generated using 401 leaf nodes for spectrum, 901 for $\log F0$, 461 for aperiodicities and 250 for duration (occupation count = 1662 / 740 / 1446 / 2666)
Audio example 18 (CD-track 18)	Utterance synthesized with GV and synthesized state durations using one mixture
Audio example 19 (CD-track 19)	Utterance synthesized with GV and synthesized state durations using two mixtures

Chapter 5

Variance and Roughness based parameter generation

One of the mayor problems of HMM-based synthesis is the perceived "muffleness" of the generated speech. Fig. 5.1 yields a comparison between the third mel cepstral coefficient of an utterance extracted from natural speech and the curve generated by the HTS system. The generated curve is much smoother than the natural one. The parameter generation algorithm reproduces well the means of the models but fails at reproducing the full range of possible values specified by the variance. The gray area shows the standard deviation of a parameter per state according to the HMM definitions. That means that 68.2% of all the instances from the training data have a MFCC value that lies within the gray area and 31.8% of all the values lie outside. This is obviously not true for the generated curve, as it always sticks very close to the mean value of the state (dotted line).

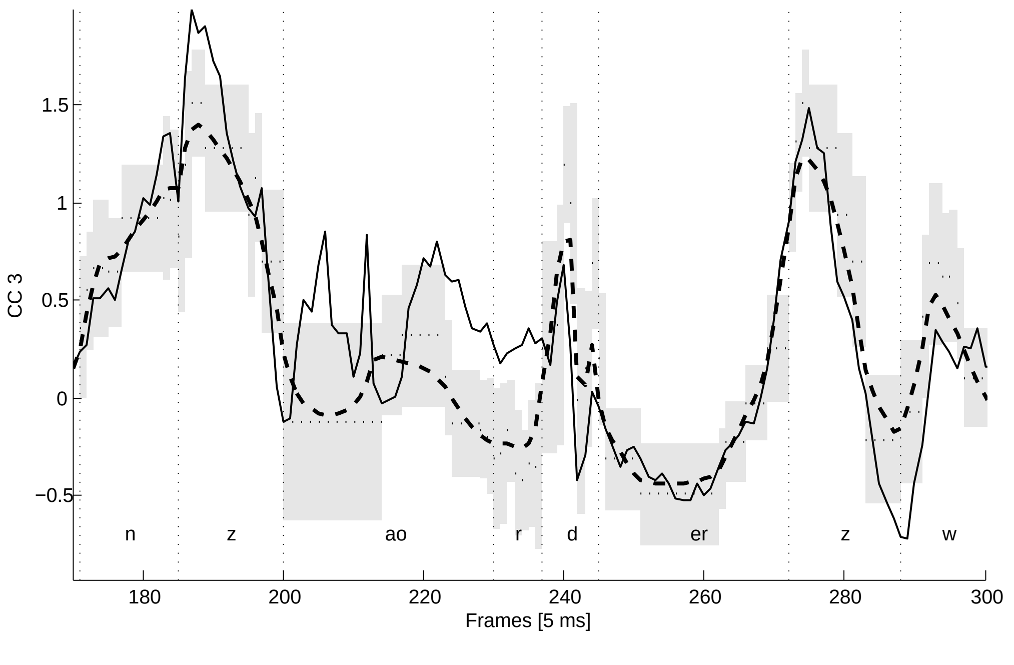

Figure 5.1: 3rd MFCC; Gray area: standard deviation of each state obtained from the HMM model definition; Solid line: natural speech; Dashed line: synthesized speech

Several approaches have been proposed to solve this problem. The following chapter describes my research to alleviate the over-smoothing problems by introducing features that describe the variance (see 5.1) and the ripple (see 5.2) of a parameter trajectory. Preserving these features in the parameter generation stage should work against an excessive smoothing of the curve. I concentrated mainly on the parameters describing the spectral envelope of speech - that is the MFCCs - because I think the unsatisfying generation of the spectral envelope plays a mayor role in the degradation of the quality in the current HTS version.

5.1 Variance based parameter generation

The variance of a parameter x is the expected quadratic distance from its mean value μ.

$$V(x) = E\{(x-\mu)^2\} \tag{5.1}$$

$$\mu = \frac{1}{T}\sum_{t=1}^{T} x(t) \tag{5.2}$$

A high variance of a feature obtained from a trained model is caused by two factors:

1. "Inter-item" variance: The items - meaning states in this context - corresponding to the same model in the training data are very different from each other. The model accuracy is too low and many different instances are mapped onto the same model. This case also occurs when training samples from different speakers are used.

2. "Intra-item" variance: The samples within one state, which are used to train the same probability density function, vary a lot. This might be the case when dealing with spectral features of plosives like e.g., "t" or "p".

A high inter-item variance of a training corpus is not desired in speech synthesis, because the averaging between many different realizations of one state can lead to a loss of details in the feature curve. Additionally, a high inter-item variance makes the mean value of the corresponding state less reliable. In speech recognition, we are interested in integrating the inter-item variance into our models, yielding more robust models against the different realizations of speech.

This forms a contrast to speech synthesis, where the intra-item variance is something that we *do* want to include into our models. Unfortunately, synthesizing the full details of a feature curve, given only one PDF per state, is an unsolvable problem.

The concept of inter- and intra-item variance can also be extended using phoneme-items instead of state-items:

1. "Inter-item" variance: The phoneme based inter-item variance describes the diversity of the realizations belonging to the same phoneme in the training corpus. The characteristics of a phoneme vary depending on the context, the speaker and the speaking style. Depending on the modeling accuracy and the cluster size, the PDFs are built from a different number of realizations of the phoneme in the database. Increasing the model-complexity - e.g., by using quintphones instead of triphones - or increasing the number of leaf nodes in the clustering step will reduce the interphoneme-variance.

2. "Intra-item" variance: The phoneme based intra-*phoneme* describes the variation of a parameter inside one realization of a phoneme. The intra-item variance depends mainly on the type of the phoneme, e.g., plosives like "t" or "p" have a higher intra-item variance than vowels.

Table 5.1 shows the mean values and variances of the 2nd MFCC of six phonemes within one utterance (generated and natural). It can be stated that:

- While the mean value of the natural speech has been well preserved by the parameter generation algorithm, the (intra-item) variance has been reduced.

Table 5.1: intra-item mean and variances of the 2nd MFCC (examples)

phoneme	$mean_{gen}$	var_{gen}	$mean_{nat}$	var_{nat}
t	0.4587	0.0123	0.7489	0.1710
l	1.6767	0.4254	1.8343	0.5007
iy	2.4852	0.0003	2.7955	0.0059
ow	2.9284	0.0139	3.0409	0.0205
b	1.9635	0.1536	1.7909	0.4153
ey	2.6937	0.0287	2.8070	0.0450

- As the values are obtained from only one realization of the corresponding phoneme, the given values represent intra-item values.

Table 5.2 shows mean and variance values taken from the trained models (monophone models and full-context models). The values represent all realizations of one phoneme in the whole database and hence include intra- and inter-item mean and variances. They have been calculated by taking the mean of all the state parameters belonging to the phoneme. Note that Tab. 5.1 and Table 5.2 cannot directly compared, because of two reasons: Firstly, the states in the upper case have different durations and secondly, Table 5.1 describes only the intra-item values. Nevertheless it can be noted that:

- With increasing accuracy of the model (from monophone to full-context), the variance decreases. Using more accurate models, less realizations are associated to one model and the inter-item variance decreases.
- The variances given by the trained models (Table 5.2) include intra- and inter-item variance of the phonemes and therefore have higher values.

Table 5.2: intra- and inter-item mean and variances of the 2nd MFCC (examples)

phoneme	$mean_{mono}$	var_{mono}	$mean_{full}$	var_{full}
t	1.0443	0.9129	0.3546	0.2321
l	2.0805	0.1924	1.7766	0.1994
iy	2.2655	0.2057	2.5025	0.0571
ow	2.8303	0.1258	2.9192	0.0568
b	1.4918	0.3676	1.9030	0.1081
ey	2.3467	0.3289	2.7521	0.0615

5.1.1 Global Variance

Toda and Tokuda 2007 proposed a new parameter generation algorithm based on a Global Variance (GV) criterion [TT07]. The Global Variance $v(\mathbf{c}, d)$ is defined as the variance of the trajectory $\mathbf{c}(d)$ of a feature d over a whole utterance of length T with t being the time frame index and $\bar{c}(d)$ denoting the mean value of the $d-th$ feature over a whole utterance.:

$$v(\mathbf{c}, d) = \frac{1}{T}\sum_{t=1}^{T}(c_t(d) - \bar{c}(d))^2 \tag{5.3}$$

$$\bar{c}(d) = \frac{1}{T}\sum_{t=1}^{T} c_t(d) \tag{5.4}$$

This variance could also be called ”sentence based intra-item variance” in the terminology described above. It cannot be compared to the (state based intra-and inter-) variance that is obtained from the trained models.

The GV of synthetic speech is lower than the GV of natural speech, caused by the MLE based parameter generation algorithm (over-smoothing problem, see 3.4.2). In the approach described below, the GV is adjusted to match the GV of natural speech.

Implementation

The following description is based on the implementation in the HTS-demo CMU-ARCTIC-SLT STRAIGHT 2.1.1 script.

Training

1. The GV of each utterance from the training data of each feature is calculated.
2. A 1-emitting-state HMM for each utterance is created: The names of the GV models are the names of the first monophone model of the corresponding utterance. The mean value of the emitting PDF is the mean value of the GV of all the utterances beginning with the same monophone.
3. The monophone models are cloned to yield full-context models. Then the models are clustered using questions that concern the whole utterance, like ”What is the number of phrases/words/syllables in the utterance ?” In the demo script, 1047 models (from 1047 utterances in the database) are divided into 2 clusters for cepstrum and aperiodicities and into 6 clusters for $\log F0$. The corresponding decision trees are saved.
4. Re-estimating the models yields the final GV models.

Synthesis

1. For each utterance to be generated, take the first model and load the corresponding mean and variance value of the GV model. Unseen models can be made using the decision trees from the training step.
2. Calculate an initial estimate for the parameter vector **c** using only the HMM (acoustic) models.
3. Calculate the Global Variance of the generated sequence according to Eq. 5.3.
4. Calculate the square root of the ratio of the calculated GV $\mu_{GV,generated}$ and the target GV $\mu_{GV,models}$ (from the GV models) for every feature d:

$$ratio(d) = \sqrt{\frac{\mu_{GV,models}(d)}{\mu_{GV,generated}(d)}} \tag{5.5}$$

5. Expand each frame t of the generated parameter vector **c** according to *ratio*. Silence or pause segments are not expanded.

$$c(d,t) = ratio(d) \times (c(d,t) - \mu_c(d)) + \mu_c(d) \tag{5.6}$$

   $\mu_c(d)$ denotes the mean value of feature d of the generated sequence $\mathbf{c}(d)$.

6. In the conventional parameter generation algorithm (see 3.3), the likelihood $P(\mathbf{o}|\hat{\mathbf{Q}}, \lambda)$ is maximized to find the most probable parameter sequence $\mathbf{o}$, given the HMM parameters λ and the most likely state and mixture sequence $\hat{\mathbf{Q}}$. Using the GV criterion as an additional constraint, the likelihood $P(v(\mathbf{c})|\lambda_v)$ of the GV $v(\mathbf{c})$ using the GV model λ_v also has to be taken into account. This is done by maximizing $P(\mathbf{o}|\lambda, \lambda_v)$, the product of the two likelihoods:

$$P(\mathbf{o}|\lambda, \lambda_v) = P(\mathbf{o}|\hat{\mathbf{Q}}, \lambda)^{\omega} P(v(\mathbf{c})|\lambda_v) \tag{5.7}$$

where ω is a weight for controlling the balance between the two likelihoods.

Using the gradient ascent algorithm, a local maximum of the two likelihoods can be found iteratively [TT07, TY09].

Results

Fig. 5.2 shows a time sequence of the 3rd mel cepstral coefficient extracted from natural speech in comparison to a generated trajectory with and without using the Global Variance criterion.

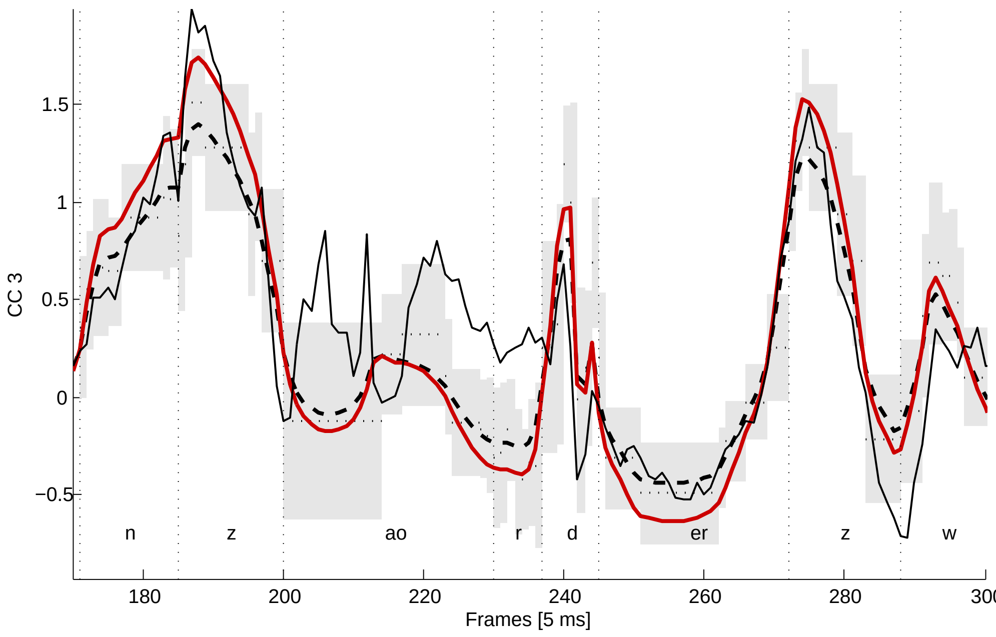

Figure 5.2: 3rd MFCC; Gray area: standard deviation of each state obtain from the HMM definition; Dashed line: generated without GV; Red solid line: generated with GV; Thin solid line: Natural speech

Fig. 5.3 shows the STRAIGHT spectrum of natural and generated speech with and without Global Variance adjustment over the time .The Global Variance of the parameter curve was increased while preserving the main characteristics of the curve (e.g., formant positions). The increase in GV sharpens the formants and hence leads to clearer speech.

Audio examples

Audio example 06 (CD-track 06)	Utterance synthesized without adjustments
Audio example 07 (CD-track 07)	Utterance synthesized with adjusted GV

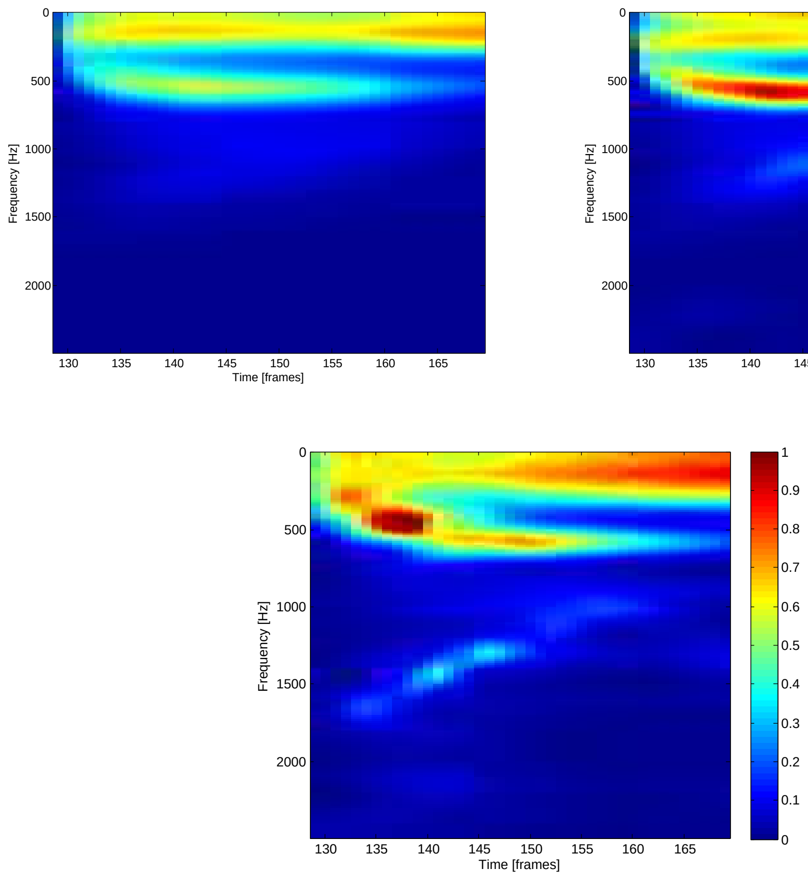

Figure 5.3: STRAIGHT spectrum of vowel ”ow”. Upper left: synthesized without GV; Upper right: synthesized with GV; Bottom: original speech

5.1.2 Local Variance

In analogy to the Global Variance, a Local Variance algorithm is investigated. The idea is to reconstruct the phoneme based “intra-item” variance of natural speech. Therefore, the term “Local Variance” (LV) is introduced that describes the variance of a parameter within one realization of a phoneme.

The LV $v(d, p)$ of a phoneme p and a feature d is calculated as follows:

$$v(d,p) = \sum_{t=start_p}^{end_p} (c(d,t) - \bar{c}_p(d))^2 \tag{5.8}$$

$$\bar{c}_p(d) = \frac{1}{T_p} \sum_{t=start_p}^{end_p} c(d,t) \tag{5.9}$$

with $start_p$ and end_p denoting the start- and endframe of the phoneme p, T_p the number of frames belonging to the phoneme, $c(d, t)$ the value of the feature d at time-frame t and $\bar{c}_p(d))$ the mean value of feature d within the phoneme.

The LV of a natural and a generated utterance is shown in Fig. 5.4. As expected, the variance within plosive phonemes like “p“, ”t“ or ”d“ is higher than the variance of other phonemes. To test if an adjusted LV increases the naturalness of the generated utterance, the LV of a natural utterance is extracted and a generated utterance is converted to match

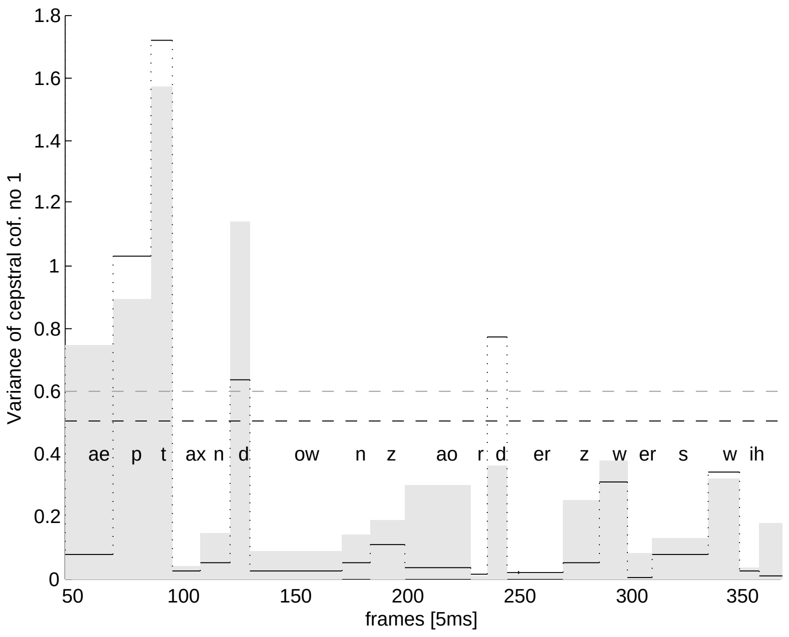

Figure 5.4: Black solid line: Phoneme based variance of the synthesized utterance; Grey area: State-wise variance of the natural utterance; Black dashed line: Global Variance of the synthesized utterance; Grey dashed line: Global Variance of the natural utterance

the natural LV. Obviously, the LV of the natural parameter sequence is not given in reality and would have to be obtained by a Local Variance model.

To modify the LV of a parameter curve, the variance ratio between generated LV $v_{gen}(d,p)$ and natural LV $v_{nat}(d,p)$ is calculated analog to Eq. 5.5:

$$ratio(d,p) = \sqrt{\frac{v_{nat}(d,p)}{v_{gen}(d,p)}} \tag{5.10}$$

Then the parameter curve can be extended using

$$c(d,t) = ratio(d,p) \times (c(d,t) - \mu_{c,p}(d)) + \mu_{c,p}(d) \tag{5.11}$$

with $t_{p_{start}}...t_{p_{end}}$ denoting the frames belonging to the phoneme p, and $\mu_{c,p}(d)$ being the mean value of the parameter curve $c(d)$ within the phoneme p.

The ratio of a 3rd MFCC feature curve is shown in Fig. 5.5.

Modification with unsteady variance ratio

The generated curve can be expanded to match the target LV according to Eq. 5.6. To prevent artifacts at the phoneme borders due to the unsteady ratio curve, linear interpolation over 3 frames was applied at each border. A LV-adjusted 3rd MFCC trajectory is shown in Fig. 5.6.

Fig. 5.6 indicates that there are still discontinuities at the phoneme borders.

Modification with smooth variance ratio

To avoid the discontinuities described above two measures are taken:

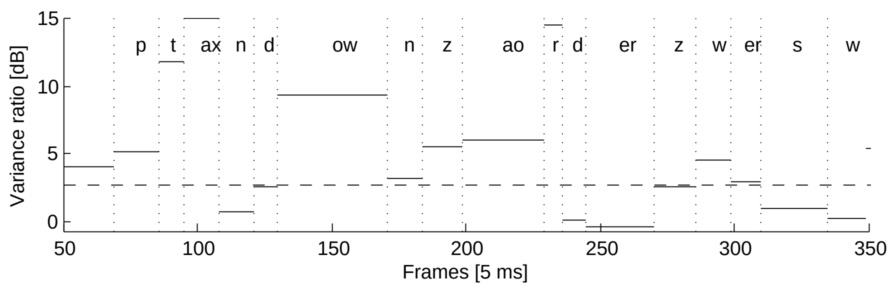

Figure 5.5: Variance ratio of the 3rd MFCC; Dashed line: Global Variance of the synthesized utterance

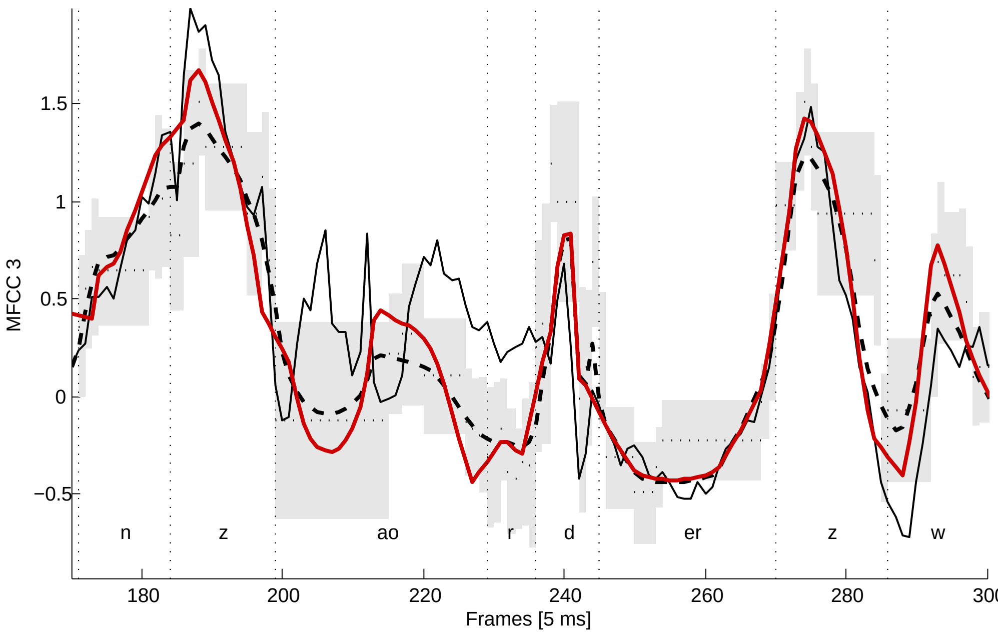

Figure 5.6: 3rd MFCC; Gray area: standard deviation of each state obtain from the HMM definition; Dashed line: generated without LV; Red solid line: generated with LV; Thin solid line: Natural speech

- The ratios of the corresponding central frame of each phoneme are linearly interpolated to get a smoother trajectory, see Fig. 5.8. This avoids an unsteady ratio curve.
- All phoneme based variances have been calculated using the mean value of the corresponding phoneme. This is a problem when modifying the parameter curve according to Eq. 5.11, because the mean values $\mu_{c,p}(d)$ jump at the phoneme borders. To circumvent this problem, we introduce auxiliary mean values $\hat{\mu}_c(d,t)$, that lay on a line interpolating the parameter value $c(d, t_{p_{start}})$ with the value $c(d, t_{p_{end}})$.

Using these values, the LV of the parameter curve can be adjusted using:

$$c(d,t) = ratio_{interpolated}(d,p,t) \times (c(d,t) - \hat{\mu}_c(d,t)) + \hat{\mu}_c(d,t) \tag{5.12}$$

$$\hat{\mu}_c(d,t)) = c(d,t_{p_{start}}) + \frac{t - t_{p_{start}}}{T_p} \times [c(d,t_{p_{end}}) - c(d,t_{p_{start}})] \tag{5.13}$$

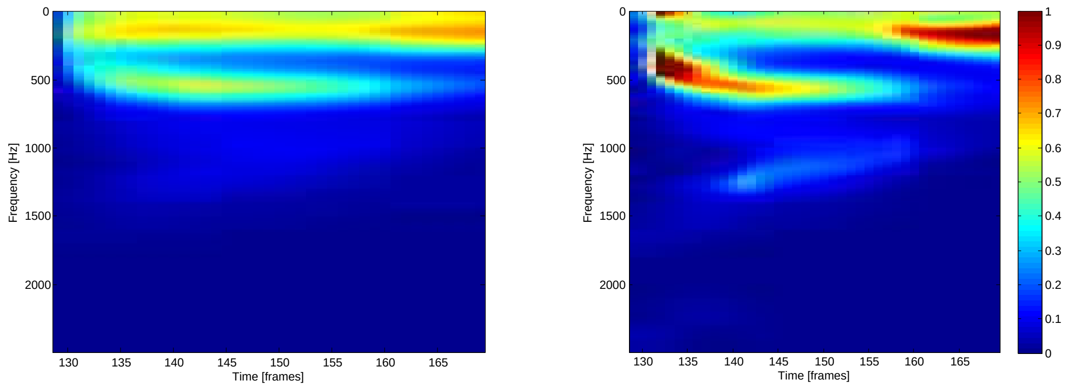

Figure 5.7: STRAIGHT spectrum of vowel "ow"; Left: synthesized without LV; Right: synthesized with LV

with $t = t_{p_{start}}...t_{p_{end}}$ denoting the frames corresponding to the phoneme p and T_p the number of frames of p.

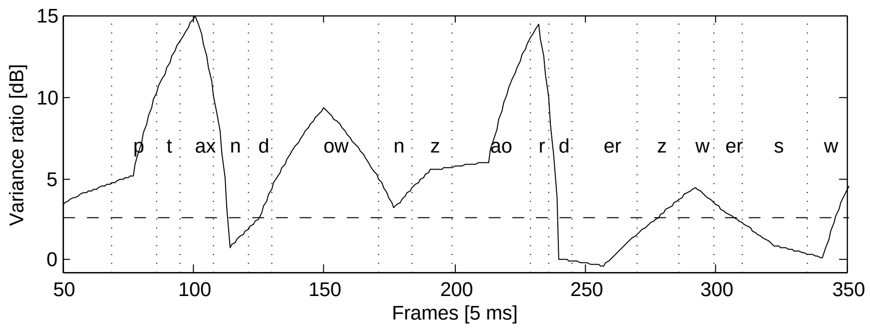

Figure 5.8: Variance ratio of the 3rd MFCC; Dashed line: Global Variance of the synthesized utterance

The problem of the discontinuities could be solved using the proposed approach. Fig. 5.9 shows the smooth parameter trajectory with adjusted LV. Fig. 5.10 shows that the second formant at 1000 Hz of the vowel "ow" is amplified and sharpened. However, it has to be stated that the natural Local Variance was assumed to be given, which is not true in reality.

Audio examples

Audio example 08 (CD-track 08)	Utterance synthesized with adjusted LV (unsteady variance ratio)
Audio example 09 (CD-track 09)	Utterance synthesized with adjusted LV (smooth variance ratio and global mean value)

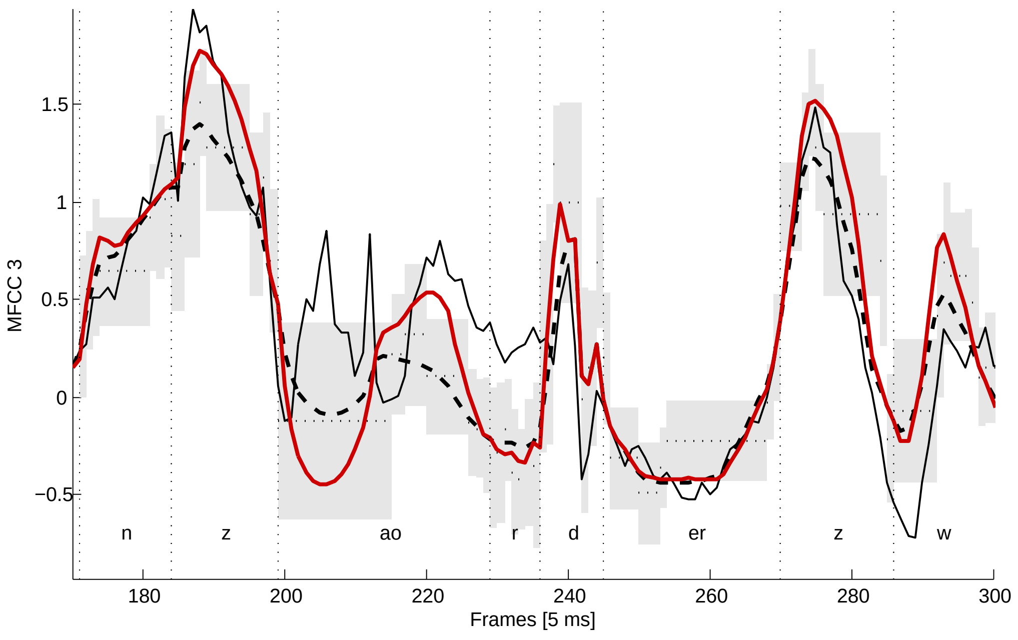

Figure 5.9: 3rd MFCC; Gray area: standard deviation of each state obtain from the HMM definition; Dashed line: generated without LV; Red solid line: generated with smooth LV; Thin solid line: Natural speech

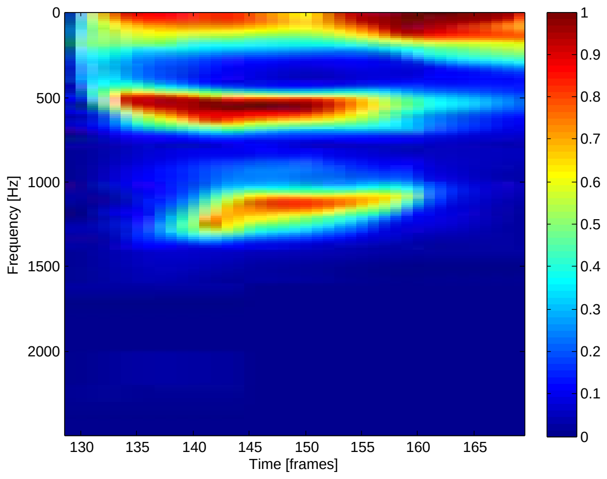

Figure 5.10: STRAIGHT spectrum of vowel ”ow”; synthesized with smooth LV

5.2 Roughness based parameter generation

In the previous section, the variance properties of a generated feature sequence was investigated. In Fig. 5.10 and 5.3 it can be seen that the variance adjustment is a good approach to alleviate the oversmoothing problem in the frequency domain, as it sharpens the formants. However when looking at the time domain, the variance-adjusted curves still look very smoothed compared to the natural ones (see Fig. 5.2).

Fig. 5.11 yields two sinus signals that share the same variance (amplitude), but differ in frequency. When only adjusting the variances, one aspect of a curve is neglected: the frequency dimension. In this section, the frequency components that are present in a parameter curve are tested for their influence on the speech quality. Here, the term "frequency domain" is used for the Fourier transformation of all parameter time sequences, including sequences of MFCCs.

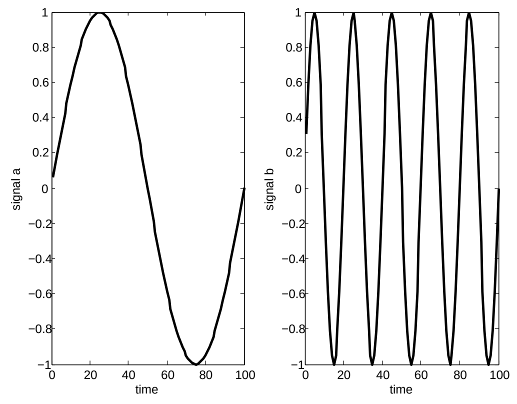

Figure 5.11: Two signals with the same variance

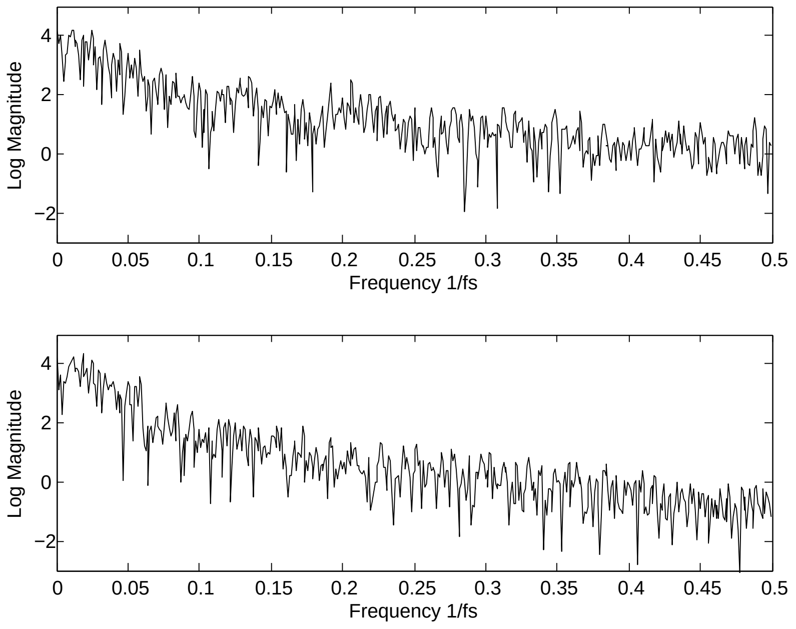

Figure 5.12: Spectrum of the 3rd MFCC sequence; Top: natural speech; Bottom: generated speech (with GV); Sampling frequency fs = 200 Hz (with a frameshift of 5 ms)

Fig. 5.12 shows the Fourier Transform of a MFCC time sequence. While the generated curve is similar to the natural curve in the low frequency bands, the natural one contains larger high frequency components - that cannot be modeled manipulating the variance only.

Due to the statistical modeling with unsatisfying model accuracy, spectral details, represented by the higher frequency components of a feature curve, get lost. In order to investigate the acoustical relevance of higher frequency bands in a feature curve, a "roughness" criterion is introduced in the following.

5.2.1 Definitions

In metrology, the terms *waviness* and *roughness* are used to describe a mechanical surfaces. This is illustrated in Fig. 5.13.

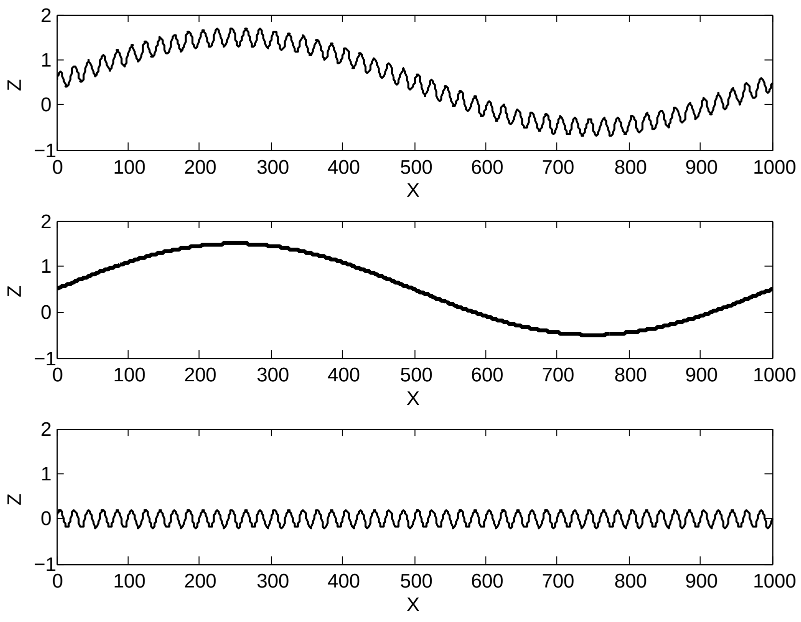

Figure 5.13: Top: (primary) profile of a surface; Center: profile of waviness; Bottom: profile of roughness

To distinguish between waviness and roughness, a limiting frequency f_r is defined. The waviness refers to the frequency components below f_r of a measured profile, whereas the roughness determines the frequency components above. This mechanical model can be transferred to the speech parameter curves. Assuming the waviness of a feature curve has been reconstructed satisfactorily using the GV adaptation, acoustical details could be added by manipulating the roughness of a feature curve.

There are many different roughness parameters in literature and I will concentrate on the most common one, that is the arithmetic average or center line average R_a. It is calculated by summing up the signal components after a highpass filtering with cut-off frequency f_r:

$$r = \frac{1}{N-2K} \sum_{k=K}^{N-K-1} |X[k]| \tag{5.14}$$

where $|X[k]|$ is the magnitude of the k-th Fourier coefficient of the (real-valued) time signal x[t], N is the FFT length, and $K = N \times \frac{f_r}{f_s}$ is a number between 0 and N/2, corresponding to the cut-off frequency f_r of the highpass filter. Note that if we sum up only the frequency components from N/4 to N/2, this corresponds to a highpass filtering with a cut-off frequency of $f_s/4$ (f_s denotes the sampling frequency), if x[t] is real.

5.2.2 Global Roughness

Similar to the definition of the Global Variance, a Global Roughness (GR) is defined as the roughness of a feature vector over a whole utterance.

Calculation of the Global Roughness

The roughness is calculated with Eq. 5.14 using the Fourier Transform of a whole utterance. The output after the highpass is shown in Fig. 5.14.

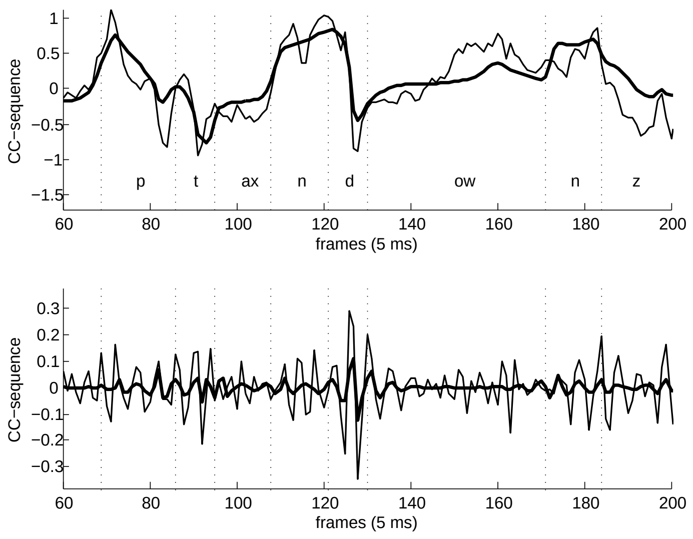

Figure 5.14: 3rd cepstral coefficient sequence. Thin line: natural speech; Thick line: generated speech (GV); Top: untouched signal; Bottom: signal after Hipass ($f_r/f_s = 0.225$)

In Fig. 5.15, the Global Roughness of the cepstrum of an utterance is plotted in respect to the MFCC order. The roughness was normalized by the variance of the feature vector, to take into account the different range of the values. The normalized roughness increases with the coefficient order, which is caused by the fact that higher order cepstral coefficients fluctuate more than the lower order coefficients.

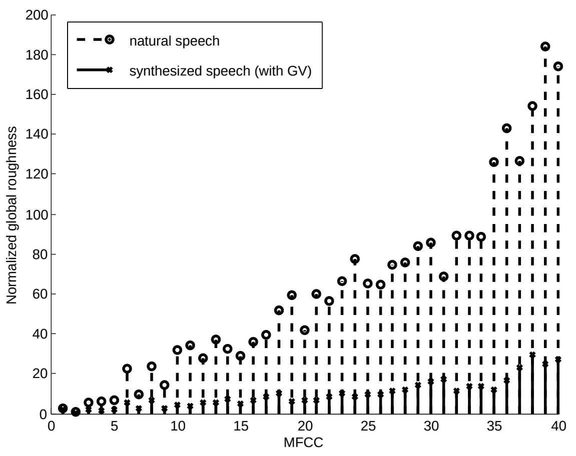

Figure 5.15: Global roughness normalized by the feature variance ($f_r/f_s = 0.225$)

Adjustment of the Global Roughness

The ratio between the Global Roughness of the natural and the generated utterance is calculated for each MFCC :

$$ratio = \frac{r_{nat}}{r_{gv}} \tag{5.15}$$

The spectral bins corresponding to a frequency $f > f_r$ are multiplied by this ratio. To prevent distortions between adjacent frequency bins at f_r, a linear fade-in is used. The spectral boosting is shown in Fig. 5.16.

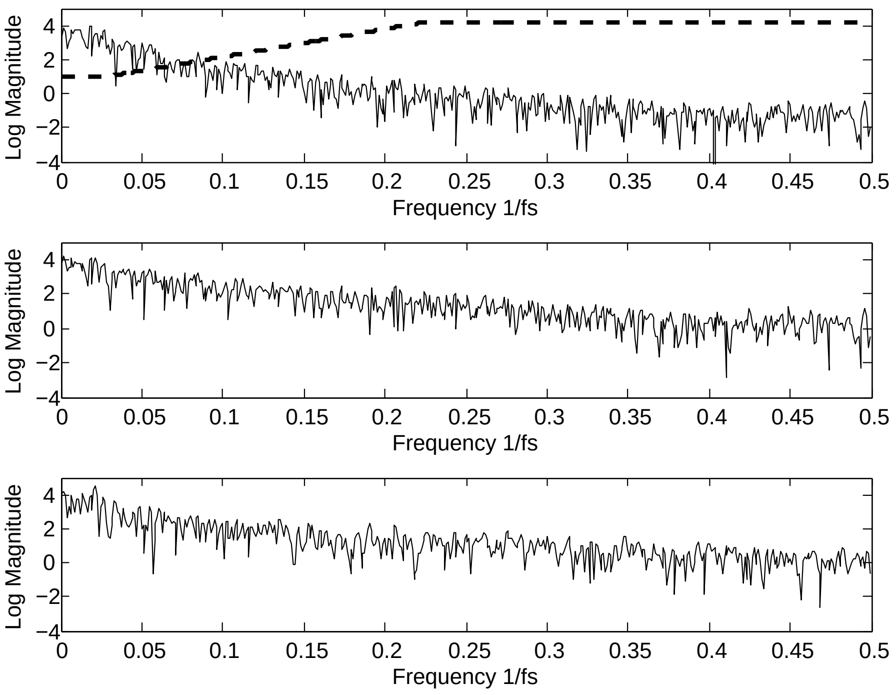

Figure 5.16: Spectrum of the 3rd MFCC of an utterance. Dashed line: spectral boosting function; Top: generated signal; Center: generated signal with boosted spectrum; Bottom: original speech

The GR-boosting is applied to the GV-adjusted parameter curve, because the GV adjustment was found to improve the speech quality by sharpening the formants (see 5.1.1) and hence provides a better initial signal for the GR adjustment. Fig. 5.17 and 5.18 show that the GR adjustment sharpens the edges of a parameter curve as they contain a high magnitude of high frequency components.

5.2.3 Local Roughness

Using the short time Fourier transform (STFT) a framewise (local) roughness (LR) is defined and investigated.

Calculation of the Local Roughness

The discrete signal x is divided into overlapping blocks. The Fourier transform of a block can be interpreted as a snapshot of the spectrum at the center time frame m of the block. Each block is multiplied with a Hanning window in the time domain to reduce spectral leakage.

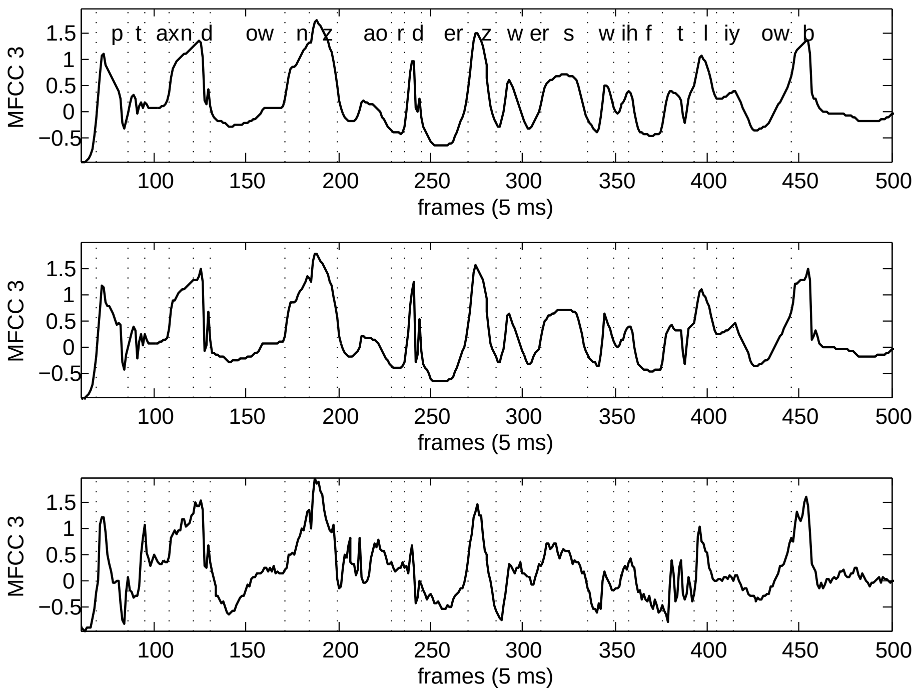

Figure 5.17: 3rd MFCC. Top: generated signal (GV); Center: generated signal (GV + GR); Bottom: original speech

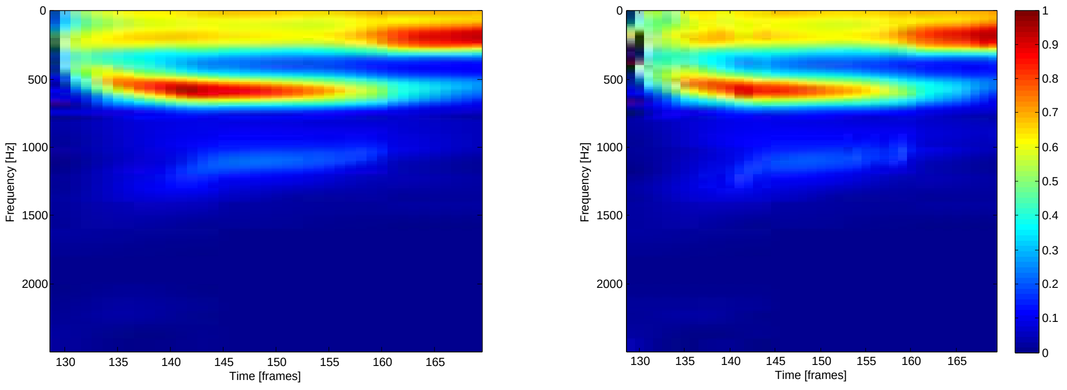

Figure 5.18: STRAIGHT spectrum of vowel "ow"; Left: synthesized without GR; Right: synthesized with GR

$$X(m,k) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\frac{2\pi}{N}kn} \tag{5.16}$$

m denotes the center position of the window inside of x and therefore defines the time frame associated with the spectrum $X(m,k)$. x is the causal, real, discrete time signal, k the number of the Fourier coefficient, n the time frame index, N the FFT length, and w is the window function that defines the size of the block. To facilitate an inverse short time Fourier transform (ISTFT) after having adjusted the roughness, the height of the window has to be scaled to fulfill

$$\sum_{n=-\infty}^{\infty} w[n] = 1 \tag{5.17}$$

Combining Eq. 5.14 and Eq. 5.16 yields the short time roughness $r(m,d)$:

$$r(m) = \frac{1}{N-2K} \sum_{k=K}^{N-K-1} \Big| \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\frac{k}{N}n} \Big| \tag{5.18}$$

with $x(n)$ denoting the value of the feature vector at time frame n and $K = N \times \frac{f_r}{f_s}$ is the Fourier coefficient representing the cut-off frequency f_r of the highpass filter.

In the following, the Local Roughness is determined using a window size of 16 samples and $f_r = 0.225/f_s$.

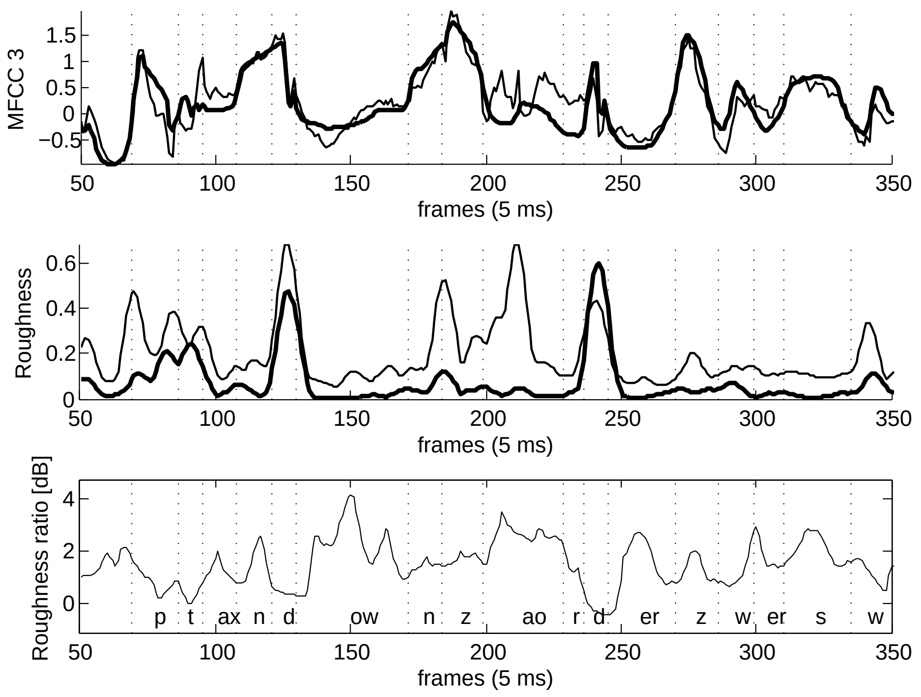

Figure 5.19: Top: 3rd MFCC sequence (thin line: natural speech, thick line generated speech (GV)); Center: roughness (thin line: natural speech, thick line generated speech (GV)); Bottom: LR ratio

The biggest deviations from natural speech occur at vowels (see peaks of the LR ratio at "ow", "ao", "er"). These are also the longest phonemes. As we have only a restricted number of states (here 7 emitting states), the low roughness value is caused by a lacking model accuracy. E.g., the vowel "ow" occupies 40 frames. While five states only occupy one frame each, the remaining 25 frames belong to two states that are modeled by a single Gaussian distribution each. This evidently results in a loss of acoustic details.

Adjustment of the Local Roughness

To find out, if the correct roughness correlates with the perception of natural-sounding speech, the roughness values of a generated utterance are manipulated to match the roughness values of the same utterance spoken by a human speaker. The ratio between the roughness of an utterance with natural speech and the same utterance with generated speech (with GV) is determined for each frame m by

$$ratio(m) = \frac{r_{nat}(m)}{r_{gv}(m)} \tag{5.19}$$

and plotted in Fig. 5.19.

The feature vector is transformed into the frequency domain using the STFT (Eq. 5.16). The frequency bins k of $X(k)$ corresponding to the frequencies above f_r are multiplied with the ratio determined in Eq. 5.19 accordingly to yield $\hat{X}$:

$$\hat{X}(m,k) = X(m,k) \times ratio(m) \tag{5.20}$$

with $k = K...(N-K-1)$. An ISTFT (see Eq. 5.21) re-transforms the manipulated spectrum $\hat{X}$ back to the time domain:

$$x(n) = \frac{1}{L} \sum_{k=-\infty}^{\infty} \sum_{m=1}^{L} \hat{X}(m,k) e^{-j\frac{2\pi}{N}kn} \tag{5.21}$$

with L denoting the length of the time window w. Due to the condition stated in 5.17, the blocks can be overlapped and added without scaling problems.

The adjustment of the LR is repeated until the error between the natural LR and the generated LR is minimized. This yields Fig. 5.20.

The roughness of the generated curve (Fig. 5.20, 2nd from top) matches the roughness profile of the natural speech. Fig. 5.21 shows that the spectrum is less smooth while preserving the positions of the formants.

Audio examples

Audio example 10 (CD-track 10)	Utterance synthesized with natural GR
Audio example 11 (CD-track 11)	Utterance synthesized with natural LR

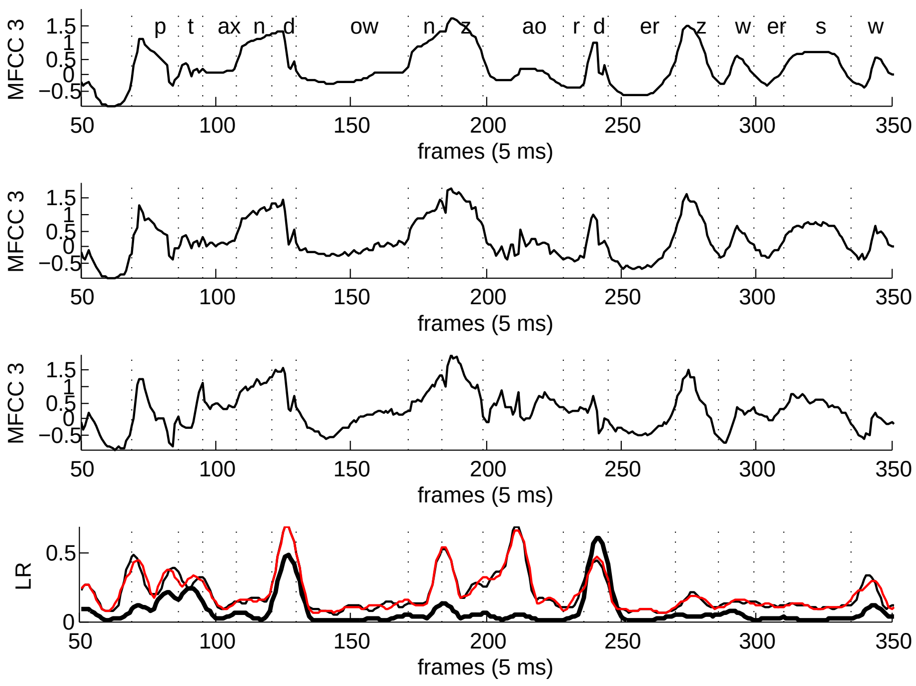

Figure 5.20: 3rd MFCC sequences (from top to bottom); 1st: generated with GV; 2nd: generated with GV and LR; 3rd: natural speech; 4th: roughness (thick solid black line: only GV; thin solid black line: natural speech; red line: GV and LR

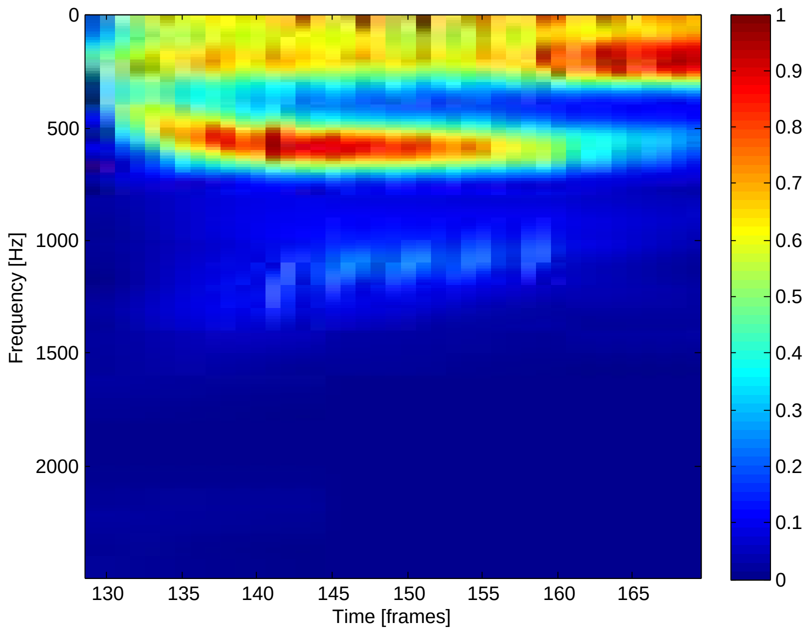

Figure 5.21: STRAIGHT spectrum of vowel ”ow”; synthesized with LR

Chapter 6

Experiments

A listening test provides an evaluation of the approaches described in the previous section. The test is supposed to find out if the variance and roughness based features are relevant for the quality of the synthesized speech. The variance based features alleviate the over-smoothing problem in the frequency domain, whereas the roughness based parameter generation increases the temporal details of a curve and reduces its temporal over-smoothing by boosting higher frequencies of the parameter curve. The first part of this section describes the design of the test, the second part the testing procedure and the last part presents the results.

6.1 Description of the experimental setup

14 people, among them 10 men and 4 woman, participated in the test. 7 of them can be regarded as "expert listeners", as they have been studying electrical engineering-sound engineering at the Technical University Graz and the University of Music and dramatic Arts or they have been working at the Signal Processing and Speech Communication Laboratory for several years. The participants were between 23 and 53 years old and performed the tasks at home or in their offices using headphones. The duration of the test was varying, as an individual number of tests could be freely chosen by the participants. Most of them made 10-15 of the overall 20 test cycles. Each of the tests lasted about 3 minutes. Over a webpage the participants could download 20 folders, each of them containing 6 audio examples of the same utterance. The 20 utterances were taken from the CMU ARCTIC databases [KB03](b520 - b539). The six versions of each utterances were randomly titled by a number from 1 to 6. The participants should bring the 6 examples into the order, such that the most preferred example is number 1 and the least preferred example is number 6.

The audio examples were produced by the STRAIGHT synthesis filter using the F0 and aperiodicities generated by the HTS-2.1.1 demo script. The state duration sequence was taken from a natural recording of the utterance. All the STRAIGHT parameters (F0, aperiodicities and spectrum) were generating using the same natural state duration sequence. The STRAIGHT spectrum was obtained using different approaches:

- Extraction from natural speech
- Synthesis with the HTS-2.1.1 demo script without Global Variance adaption
- Synthesis with the HTS-2.1.1 demo script with GV adaption (see 5.1.1). The GV of the synthesized cepstrum was calculated and boosted to match the GV of the natural cepstrum.
- Synthesis with the HTS-2.1.1 demo script with LV adaption (with unsteady LV ratio, see 5.1.2). The variance of the synthesized cepstrum was calculated phoneme-wise and boosted to match the variance of the corresponding phonemes of the natural cepstrum.

- Synthesis with the HTS-2.1.1 demo script with GV and GR adaption (see 5.2.2). The GR of the synthesized cepstrum (GV adapted) was calculated and boosted to match the GR of the natural cepstrum.
- Synthesis with the HTS-2.1.1 demo script with GV and LR adaption (see 5.2.3). The roughness of the synthesized cepstrum (GV adapted) was calculated frame-wise and boosted to match the roughness of the corresponding frames of the natural cepstrum.

6.2 Comparison of Variance- and Roughness-based parameter generation

The GV based parameter generation alleviates the over-smoothing in the frequency domain. Fig. 5.3 shows that the formants of the GV adapted spectrum are more expressive and sharper than without GV. This is also true for the LV adapted signal. The frequency valleys are equally expressive as in the global case, but the phoneme-wise adjustment introduces distortions because the relations between phonemes are disturbed and the parameter curve jumps at phoneme borders. In the roughness cases, the roughness was increased by boosting high frequency components in the spectrum of a feature curve. This implies that only frequency components can be amplified that are already present in the signal. High frequencies mostly appear at sharp edges of the curve. When boosting these high frequencies, mainly those regions are amplified, where jumps appear. This can be seen in Fig. 5.17. Adjusting the LR of a parameter curve also affects the more steady regions of the signal, as it can be seen in Fig. 5.20.

To compare the approaches described above, the error between the generated signal and the same signal produced by a human speaker is plotted in Fig. 6.1 for the first 15 cepstral coefficients of the cepstrum. The error is calculated by summing up the Euclidean distances of each synthesized frame from the corresponding natural frame. It is normalized by the error of the generated signal without variance and roughness adjustments. The signal without adjustments (no GV, no RN) has the lowest error, as it was generated using the maximum likelihood criterion that minimizes the error. Adjusting the GV, the error of the coefficients 2-4 decreases, but the overall error increases. All the other approaches augment the error.

6.3 Perceptual evaluation

Each audio example was assigned a number from 6 (best quality) to 1 (worst quality). For each of the 6 approaches, the mean value of all the assigned numbers was calculated to yield the final preference score.

The listening test revealed that the Global Variance criterion produces the best results among the investigated approaches. It clearly enhances the quality of the generated speech, sharpening the formant regions while preserving the relations within the generated speech signal.

All the other approaches to modify the variance locally or to adjust the roughness of the signal lead to degradation of the speech quality.

However, it can be noted that the result of the listening test does not correspond with the Euclidean error measure as shown in Fig. 6.1.

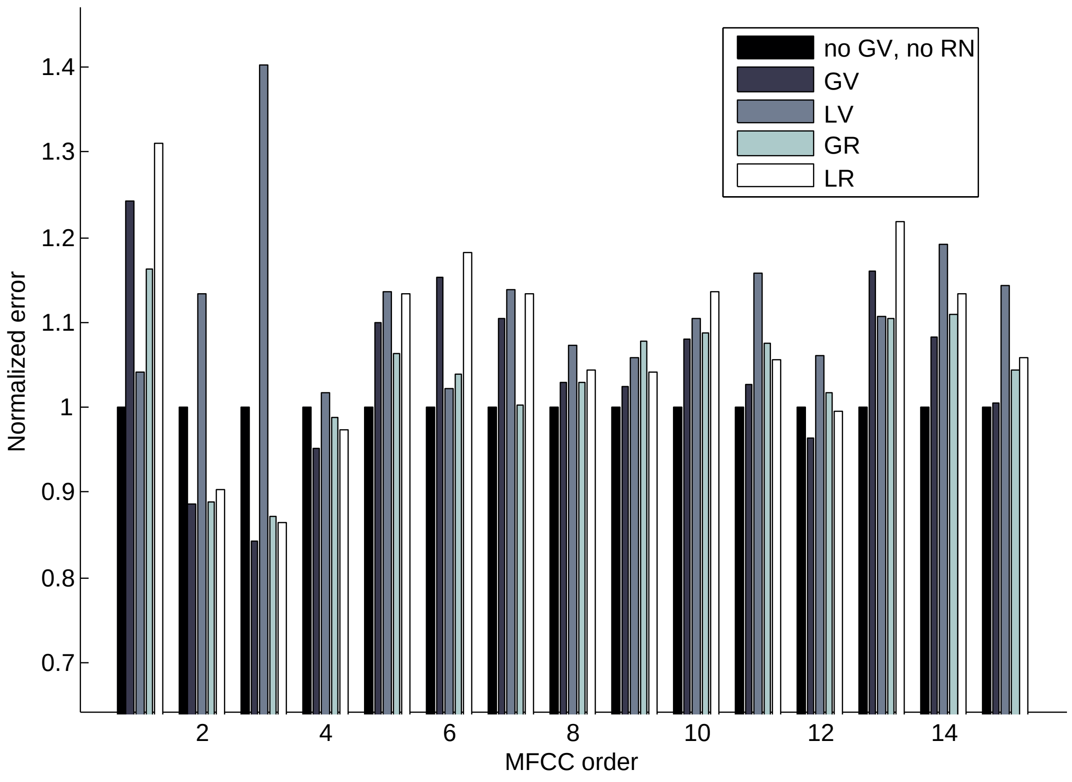

Figure 6.1: Error (Euclidean distance) between synthesized and natural signal, normalized by the error of the synthesized signal without any adjustments (no GV, no RN)

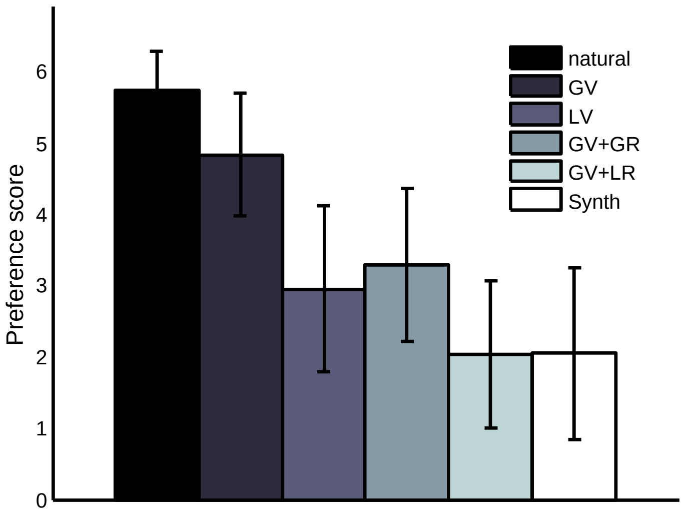

Figure 6.2: Results of the listening test

Audio examples

The following examples, are taken from the listening test.

"So far as flags were concerned, they were beyond all jurisdiction"

Audio example 12 (CD-track 12)	Utterance synthesized with natural cepstrum
Audio example 13 (CD-track 13)	Utterance synthesized with adjusted GV
Audio example 14 (CD-track 14)	Utterance synthesized with adjusted LV
Audio example 15 (CD-track 15)	Utterance synthesized with adjusted GR
Audio example 16 (CD-track 16)	Utterance synthesized with adjusted LR
Audio example 17 (CD-track 17)	Utterance synthesized without adjustments

Chapter 7

Conclusions

In this work, different methods for improving the generation of parameter curves (especially the spectral envelope) in Hidden Markov Model-based speech synthesis were investigated and evaluated in terms of their influence on the speech quality. For this purpose, the HMM-based speech synthesis system (HTS) based on the Hidden Markov Model Toolkit (HTK) was chosen as a software framework, as it is a very flexible research tool offering a variety of algorithms related to HMMs.

Taking a closer look at the HTS system, the over-smoothing problem was identified as the main factor to degrade the quality of the generated speech. The maximum likelihood criterion in the parameter generation algorithm results in parameter trajectories that are close to the mean vector sequences of the HMMs, and hence fails at producing natural parameter curves. The statistical modeling removes certain characteristics of the parameter trajectories that are relevant for the speech quality. Re-introducing these characteristics leads to an enhancement of the speech quality. This thesis investigates variance and roughness features.

Inspired by a paper by Tomoki Toda and Keiichi Tokuda from 2007 about a speech parameter generation algorithm considering a Global Variance criterion [TT07], I firstly focused on the variance features and extended the approach to adjust not only the variance over a whole utterance, but also over a single phoneme.

Secondly, a new feature named "roughness" was proposed to describe the ripple of a cepstral parameter curve. The roughness is a measure for the presence of high frequency components in a parameter trajectory and was found to be higher in natural cepstral sequences than in the generated ones. When phonemes with long state durations (e.g. vowels) are modeled, temporal details of the cepstrum get lost due to the low number of available model parameters. The proposed approach reintroduces spectral details, amplifying the frequency components beyond a certain cut-off frequency in order to improve the naturalness of the generated speech.

To compare and evaluate the approaches described above, a listening tests was conducted. 10 participants had to evaluate 6 audio examples of 20 different sentences by assigning them a preference score between 6 (best quality) and 1 (worst quality). The tests revealed that the Global Variance adaption is most effective method to increase the speech quality. All the other attempts (Local Variance, Global Roughness and Local Roughness) restored the variance or roughness features that had been removed during the modeling process, but did not further upgrade the speech quality.

In the case of the roughness based features, two reasons have been identified to explain the inferior speech quality: 1) over-smoothing in the time domain has a smaller impact on the perceived quality than over-smoothing in the frequency domain [MZW08], 2) the local and the global roughness adjustments cause audible discontinuities because of the exaggerated emphasis of the signal flanks.

In the case of the Local Variance, the speech quality degraded because of unsteady signal properties at the phoneme borders. An improved local variance adaption method using a smoothed Local Variance-ratio was proposed and was found to remove some of the artifacts, but was not evaluated in the listening test.

While doing my research, a lot of new questions and ideas arose. In the limited time of this diploma thesis, I had to stop at a certain point and leave some questions for future investigations. In the following I want to document some of the ideas that occurred in the course of my thesis.

One big problem still is the lack of modeling accuracy of long state durations. It happens regularly (especially with vowels) that one state has a duration of 10 or more frames and is modeled by one probability density function. The inverse problem exists when the duration of very short phonemes (especially fricatives) is smaller than the number of states of the HMMs and no skip is used. The time- and the frequency resolution would increase if each pdf modeled more or less the same amount of frames. HMMs with bigger amount of states and skip could alleviate these problems.

Secondly, the steps of Thomas Drugman and Jian Yu [YZTW07, TATG09] could be followed further and a combination of statistical parametric synthesis and unit selection could be implemented. Every state could not only be represented by one probability density function but by a set of parameters describing an actual instance from the training corpus. If there is more than one instance in the training data associated to the same model, the longest entry would be taken. Different durations of this instance could be generated by compacting or expanding the parameter sequence. For unseen models the training sequence would have to be found that is as close as possible to the most likely parameter curve.

Finally, other machine learning approaches, such as neural networks or support vector machines, could be applied to generate the spectral envelope. It should be investigated, if other statistical methods are equally sensitive to the over-smoothing problem as HMMs are.

Appendix A

Mathematical formulations used in this thesis

A.0.1 Joint probability

$$P(A \cap B) = P(A, B) = P(AB) \tag{A.1}$$

A.0.2 Conditional probability

$$P(A|B) = \frac{P(AB)}{P(B)} \tag{A.2}$$

A.0.3 Bayes theorem

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)} \tag{A.3}$$

Appendix B
The Hidden Markov Model

The theoretical background of the HMMs was firstly described in a series of statistical papers by Leonard E. Baum et al. in the late 1960s. In the mid-1970s HMMs were successfully applied to speech recognition and later in the 1980s they were found to be very useful analyzing biological sequences such as the DNA.

In this thesis, Continuous Density Hidden Markov Models (CDHMM) are applied to speech synthesis. The related basic mathematical formulations are explained in this section.

A HMM consists of N states, that are sequentially visited (left-to-right model without skip) and is defined by a set of three parameters $\lambda = \{\mathbf{A}, \mathbf{B}, \mathbf{\Pi}\}$. A HMM has always one initial state, one or more emitting states and an ending state.

1. The transition probability matrix **A**
   A transition from state i to state j occurs with a probability of $a_{ij} = P(q_t = S_j | q_{t-1} = S_i)$. All the transition probabilities of a HMM can be defined by the $N \times N$ transition probability matrix **A**.

2. The state output probability matrix **B**
   The state i outputs a certain feature value with an output probability b_i. As we assume the feature value to be a continuous quantity (CDHMM), each feature is assigned a probability density function. These PDFs usually are represented by a sum of multivariate Gaussian distributions. The probability that the state i produces the observation vector $\mathbf{o}_t$ at time t can be written as

   $$b_i(\mathbf{o}_t) = \sum_{m=1}^{M} w_{im} N(\mathbf{o}_t; \boldsymbol{\mu}_{im}, \mathbf{\Sigma}_{im}) \tag{B.1}$$

   with the definition of a Gaussian distribution

   $$N(\mathbf{o}_t; \boldsymbol{\mu}_{im}, \mathbf{\Sigma}_{im}) = \frac{1}{\sqrt{(2\pi)^D |\mathbf{\Sigma}_{im}|}} e^{-\frac{1}{2}(\mathbf{o}_t - \boldsymbol{\mu}_{im})^\top \mathbf{\Sigma}_{im}^{-1} (\mathbf{o}_t - \boldsymbol{\mu}_{im})} \tag{B.2}$$

   M denotes the number of mixtures and D is the dimensionality of the feature space. w_{im} is the weight, $\boldsymbol{\mu}_{im}$ the D-dimensional mean vector and $\mathbf{\Sigma}_{im}$ the $D \times D$ covariance matrix of mixture m and state i.

3. The initial probability vector $\mathbf{\Pi}$

   $$\mathbf{\Pi} = \{\pi_i\}_{i=1}^{N} \tag{B.3}$$

   where π_i is the probability of the model to initially be in state i. In the left-to right models, the initial state of a HMM is always the first (initial) state.

B.1 Basic problems and algorithms

Three main problems appear when dealing with HMMs and are discussed in this section. A more detailed description can be found in [**?**].

What is the likelihood of an observation sequence O, given the model λ (Evaluation problem) ?

Let us assume we observe the feature sequence **O** with

$$\mathbf{O} = [\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_T] \tag{B.4}$$

$\mathbf{o}_t$ denoting the feature vector extracted at time t.

How probable is it, that the observation sequence **O** was produced by a given HMM λ ? This depends on two factors:

1. The output probabilities $\mathbf{b}_i$ of the observation vector $\mathbf{o}_t$ at the state i
2. The possible (hidden) state sequences Q that can be emitted by the HMM λ

$$Q = [q_{t=1}, q_{t=2}, ...q_{t=T}] \tag{B.5}$$

with q_t is the state assigned to the time frame t. If we sum up all the possible state sequences that could occur, multiply them by the observation probability $\mathbf{b}_i$ of the corresponding state i, we get a simple but very impractical solution:

$$P(\mathbf{O}|\lambda) = \sum_{allQ} P(\mathbf{O}, Q|\lambda) \tag{B.6}$$

Using Bayes' rule (Eq. A.0.3) the term $P(\mathbf{O}, Q|\lambda)$ can be split into two parts:

$$P(\mathbf{O}, Q|\lambda) = P(\mathbf{O}|Q, \lambda)P(Q|\lambda) \tag{B.7}$$

The first term $P(\mathbf{O}|Q, \lambda)$ can be calculated multiplying the observation probabilities b_{qt} for all the time frames t, where q_t is the state corresponding to time frame t:

$$P(\mathbf{O}|Q, \lambda) = b_{q1}(\mathbf{o}_1) \times b_{q2}(\mathbf{o}_2) \times ... \times b_{qT}(\mathbf{o}_T) \tag{B.8}$$

The second term $P(Q|\lambda)$ is the probability of the state sequence Q given the HMM λ and can be calculated by multiplying the state transition probabilities between two adjacent time frames in the sequence Q:

$$P(Q|\lambda) = a_{q1} \times a_{q1q2} \times ... \times a_{qT-1qT} a_{qTN}) \tag{B.9}$$

Combining Eq. B.8 and Eq. B.9 yields:

$$P(\mathbf{O}|\lambda) = \sum_{all\ Q} a_{q1} b_{q1}(\mathbf{o}_1) a_{q1q2} b_{q2}(\mathbf{o}_2) ... a_{qT-1qT} a_{qTN} b_{qT}(\mathbf{o}_T) \tag{B.10}$$

The demand in computational power of solving equation B.10 increases exponentially with the length of the observation sequence. A method to calculate $P(\mathbf{O}|\lambda)$ more efficiently is the *forward-algorithm.*

The forward-algorithm The problem of solving equation B.10 directly is that all the state sequences are treated independently. Even if two state sequences differ only in the last state, the probability of both of them is calculated from the first to the last state. It would be a lot more efficient to reuse the results from other state sequences that have equal states up to a certain frame. This is done in the forward-algorithm, where probabilities called forward probabilities $\alpha_t(i)$ are saved for each state i and each step t in the sequence (going from left to right) and are reused for the next step. The forward probability is defined as the probability of being in state i at time t and having emitted the observation sequence $\mathbf{O}_1^t = [\mathbf{o}_1, \mathbf{o}_2, ...\mathbf{o}_t]$. To obtain the final likelihood $P(\mathbf{O}|\lambda)$, only the forward probabilities at time T have to be summed up (ignoring the initial and ending state, that are not emitting any observations):

$$P(\mathbf{O}|\lambda) = \sum_{i=2}^{N-1} \alpha_T(i) a_{iN} \tag{B.11}$$

What is the most probable state sequence $\hat{Q}$, given the model λ (Decoding problem)

The true state sequence Q, given the observation vector sequence $\mathbf{O}$, can rarely be determined with 100% security, because in most cases the same observation vector can be produced by more than one state sequence. If we still want to find out the underlying state sequence, a criterion has to be defined, how to chose the right state sequence. In our case this will be the sequence with the highest a-posteriori probability:

$$\hat{Q} = \arg\max_{Q} P(Q|\mathbf{O}, \lambda) \tag{B.12}$$

An efficient method to solve this problem is the Viterbi-algorithm.

The Viterbi algorithm The Viterbi algorithm is a recursive algorithm that calculates for each time step t (from beginning t=1 to the end of the sequence t=T) and each state j:

- the probability of the most likely state sequence that ends in the state j at timestep t and has emitted $\mathbf{O}_{t=1...t}$:

  $$\delta_t(j) = \max_{i}[\delta_{t-1}(i)a_{ij}]\mathbf{b}_j(\mathbf{o}_t) \tag{B.13}$$

  i is the state assigned to the time frame $t-1$.

- the most probable precursor state of the actual state j:

  $$\Psi_t(j) = \arg\max_{i}[\delta_{t-1}(i)a_{ij}] \tag{B.14}$$

When the recursion has ended, the most probable state sequence can be determined in taking all the most probable precursor states of the last (most probable) ending state. This procedure is known as *backtracing*.

What are the most suitable model parameters to model the training data (Training problem) ?

The most difficult problem is the training problem (also referred to as learning or estimation problem), where the model parameters $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$ are estimated from training data $\mathbf{O}$ in a manner such that the model describes the training data as good as possible. In other words we have to find the model parameter $\hat{\lambda}$ that maximize the likelihood of the training data $P(\mathbf{O}|\lambda)$.

$$\hat{\lambda} = \arg\max_{\lambda} P(\mathbf{O}|\lambda)$$

Up to now, there is no known analytical way to obtain $\hat{\lambda}$ in a closed form, so iterative algorithms have to be used to find at least a local maximum. The most popular among these algorithms is the so called *Baum-Welch-* or *Forward-backward algorithm.* In order to explain it, we first have a look at the backward-algorithm:

The backward-algorithm According to the forward-probability $\alpha_t(i)$, a so-called backward probability $\beta_t(i)$ can be defined. $\beta_t(i)$ is defined as the probability of being in state i at time t when the *next* observation vectors are $\mathbf{O}_{t+1}^{T} = \mathbf{o}_{t+1} \times \mathbf{o}_{t+2} \times \mathbf{o}_{t+3}...\mathbf{o}_T$. That means we calculate the probability of a state sequence starting from the last state and save intermediate results like in the forward algorithm. These intermediate results can again be shared among state sequences that have the same ending states to reduce complexity.

The Baum-Welch algorithm The Baum-Welch- or forward-backward algorithm is an iterative algorithm that was introduced in the early 70ies by Leonard E. Baum and Lloyd R. Welch. To describe the algorithm, two auxiliary probabilities have to be defined:

- $\gamma_t(i)$ is the probability of being in state i at time t given the observation sequence $\mathbf{O}$ and the model parameters λ. Using Bayes theorem (equation A.0.3) this can be also written as:
$$\gamma_t(i) = P(q_t = i|\mathbf{O}, \lambda) = \frac{P(\mathbf{O}, q_t = i|\lambda)}{P(\mathbf{O}|\lambda)} \tag{B.15}$$
with $1 \leq t \leq T, 1 < i < N$
The numerator of B.15 represents the probability of all the state sequences that are in state i at time t and emit the observation sequence $\mathbf{O}$, whereas the denominator stands for the probability of *all* the state sequences that emit $\mathbf{O}$. The numerator can also be expressed as the product of $\alpha_t(i)$ (covering the states before i) and $\beta_t(i)$ (covering the states after i):
$$\gamma_t(i) = \frac{\alpha_t(i) \times \beta_t(i)}{P(\mathbf{O}|\lambda)} \tag{B.16}$$

- $\xi_t(i,j)$ is the probability of a transition from state i to j given the observation sequence $\mathbf{O}$ and the model parameters λ. Using Bayes' rule again, this can be also written as:
$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j|\mathbf{O}, \lambda) = \frac{P(\mathbf{O}, q_t = i, q_{t+1} = j|\lambda)}{P(\mathbf{O}|\lambda)} \tag{B.17}$$
with $1 \leq t \leq T, 1 < i,j < N$
The numerator of B.17 represents the probability of all the state sequences that have a transition from state i to j at time t and emit the observation sequence $\mathbf{O}$, whereas the denominator stands for the probability of *all* the state sequences that emit $\mathbf{O}$. The numerator can be expressed in terms of the forward- and backward-probability as follows:
$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{P(\mathbf{O}|\lambda)} \tag{B.18}$$

Summing up these auxiliary probabilities $\gamma_t(i)$ and $\xi_t(i,j)$ over the whole observation sequence, yields very useful information:

$\sum_{t=1}^{T} \gamma_t(i)$ can be interpreted as the expected number of times that state i is visited in the sequence.

$\sum_{t=1}^{T-1} \xi_t(i,j)$ can be interpreted as the expected number of transitions from state i to state j that occur in the sequence.
From these two sums the parameters $\hat{\lambda} = (\hat{A}, \hat{B})$ of a HMM can be estimated:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)} = \frac{expected\, number\, of\, transitions\, from\, i\, to\, j}{expected\, number\, of\, times\, state\, i\, has\, been\, visited} \tag{B.19}$$

$$\hat{b}_j(o_k) = \frac{\sum_{all\, t\, with\, o_t = o_k} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} = \frac{expected\, number\, of\, times\, j\, is\, visited\, when\, observation\, o_t = o_k}{expected\, number\, of\, times\, j\, is\, visited} \tag{B.20}$$

These steps are iterated until the parameters converge and the maximum likelihood is reached. It has to be stated that it is very important to have good initial model parameters because the Baum-Welch algorithm (like the other iterative algorithms) only finds a local maximum of the likelihood.

Appendix C
CD index of audio examples

All generated audio examples (except no. 18 and 19) have been forced aligned to natural speech to yield better comparing. As this thesis concentrates on modifying the spectral envelope, the other STRAIGHT parameters (aperiodicities, F0) are synthesized without any adaptions.

CD-index	**description**
Audio example 01 (CD-track 01)	Natural speech ($16kHz, 16bit$)
Audio example 02 (CD-track 02)	Speech generated with STRAIGHT from natural parameters
Audio example 03 (CD-track 03)	Speech generated using 1943 leaf nodes for spectrum, 6361 for $\log F0$, 2410 for aperiodicities and 1128 for duration (occupation count = 343 / 105 / 277 / 591)
Audio example 04 (CD-track 04)	Speech generated using 803 leaf nodes for spectrum, 2084 for $\log F0$, 950 for aperiodicities and 505 for duration (occupation count = 830 / 320 / 701 / 1320)
Audio example 05 (CD-track 05)	Speech generated using 401 leaf nodes for spectrum, 901 for $\log F0$, 461 for aperiodicities and 250 for duration (occupation count = 1662 / 740 / 1446 / 2666)
Audio example 06 (CD-track 06)	Utterance synthesized without adjustments
Audio example 07 (CD-track 07)	Utterance synthesized with adjusted GV
Audio example 08 (CD-track 08)	Utterance synthesized with adjusted LV (unsteady variance ratio)
Audio example 09 (CD-track 09)	Utterance synthesized with adjusted LV (smooth variance ratio and auxiliary mean value)
Audio example 10 (CD-track 10)	Utterance synthesized with adjusted GR
Audio example 11 (CD-track 11)	Utterance synthesized with adjusted LR
Audio example 12 (CD-track 12)	Utterance synthesized with natural cepstrum
Audio example 13 (CD-track 13)	Utterance synthesized with adjusted GV
Audio example 14 (CD-track 14)	Utterance synthesized with adjusted LV
Audio example 15 (CD-track 15)	Utterance synthesized with adjusted GR
Audio example 16 (CD-track 16)	Utterance synthesized with adjusted LR
Audio example 17 (CD-track 17)	Utterance synthesized without adjustments
Audio example 18 (CD-track 18)	Utterance synthesized with GV and synthesized state durations using one mixture
Audio example 19 (CD-track 19)	Utterance synthesized with GV and synthesized state durations using two mixtures

Bibliography

[BZT07] Alan W. Black, Heiga Zen, and Keiichi Tokuda. Statistical parametric speech synthesis. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Honolulu, April 2007.

[EAB+04] E. Eide, A.Aaron, R. Bakis, W. Hamza, M. Picheny, and J. Pitrelli. A corpus-based approach to AHEM expressive speech synthesis authors. 2004.

[Gri87] Daniel W. Griffin. Multi-band excitation Vocoder. *RLE Technical Report*, 1987.

[Hem06] Coralie Hemptinne. Integration of the harmonic plus noise model (HNM) into the hidden markov model-based speech synthesis system (HTS). Master's thesis, IDIAP, 2006.

[HKT06] Zen. H., Tokuda K., and Kitamura T. Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic feature vector sequences. *Comput. Speech Language 21*, pages 153–173, 2006.

[IK09] Satoshi Imai and Takao Kobayashi. Speech Signal Processing Toolkit (SPTK). 2009.

[Ima83] Satoshi Imai. Analysis synthesis on the mel frequency scale. *ICASSP Boston*, 1983.

[IMN+02] T. Irino, Y. Minami, T. Nakatani, M. Tsuzaki, and H. Tagawa. Evaluation of a speech recognition/generation method based on hmm and straight. In *Seventh International Conference on Spoken Language Processing*, 2002.

[Kaw97] Hideki Kawahara. Speech representation and transformation using adaptive interpolation of weighted spectrum: vocoder revisited. *IEEE*, 1997.

[Kaw06] Hideki Kawahara. STRAIGHT, exploitation of the other aspect of VOCODER: Perceptually isomorphic decomposition of speech sounds. *Acoustical science and technology 27*, pages 349–353, 2006.

[KB03] John Kominek and Alan W. Black. CMU ARCTIC databases for speech synthesis. 2003.

[KEF01] Hideki Kawahara, Jo Estill, and Osamu Fujimura. Aperiodicity extraction and control using mixed mode excitation and group delay manipulation for a high quality speech analysis, modification and synthesis system STRAIGHT. *International Workshop on Models and Analysis of Vocal Emissions for Biomedical Applications (MAVEBA)*, 2001.

[KMT$^+$08] H. Kawahara, M. Morise, T. Takahashi, R. Nisimura, T. Irino, and H. Banno. TANDEM-STRAIGHT: A temporally stable power spectral representation for periodic signals and applications to interference-free spectrum, F0, and aperiodicity estimation. *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2008.

[LDR10] Pierre Lanchantin, Gilles Degottex, and Xavier Rodet. a HMM-based speech synthesis system using a new glottal source and vocal-tract separation method. 2010.

[Mur02] Kevin P. Murphy. Hidden semi-Markov models (HSMMs). 2002.

[MZW08] Huibin Jia Meng Zhang, Jianhua Tao and Xia Wang. Improving HMM based speech synthesis by reducing over-smoothing problems. *International Symposium on Chinese spoken language processing (ISCSLP)*, 2008.

[OWT09] Keiichiro Oura, Yi-Jian Wu, and Keiichi Tokuda. Overview of NIT HMM-based speech synthesis system for Blizzard Challenge 2009. 2009.

[SKIM92] Y. Sagisaki, N. Kaiki, N. Iwahashi, and K. Mimura. ATR ν-TALK speech synthesis system. pages 483–486, 1992.

[Sty00] Y. Stylianou. On the implementation of the harmonic plus noise model for concatenative speech synthesis. In *IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS SPEECH AND SIGNAL PROCESSING*, volume 2. IEEE; 1999, 2000.

[TATG09] Drugman T., Moinet A., Dutoit T., and Wilfart G. Using a pitch-synchronous residual codebook for hybrid HMM/frame selection speech synthesis. *Proc. ICASSP*, pages 3793–3796, 2009.

[TBC98] Paul Taylor, Alan W. Black, and Richard Caley. The architecture of the festival speech synthesis system. Edinburgh, 1998.

[TKI95a] K. Tokuda, T. Kobayashi, and S. Imai. Adaptive cepstral analysis of speech. *IEEE Transactions on Speech and Audio Processing*, 3(6):481–489, 1995.

[TKI95b] Keiichi Tokuda, Takao Kobayashi, and Satoshi Imai. Speech parameter generation from HMM using dynamic features. *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1995.

[TMMK02] Keiichi Tokuda, Takashi Masuko, Noboru Miyazaki, and Takao Kobayashi. Multi-space probability distribution HMM. *IEICE Trans. Information and Systems*, pages 445–464, 2002.

[TT07] Tomoki Toda and Keiichi Tokuda. A speech parameter generation algorithm considering Global Variance for HMM-based speech synthesis. In *IEICE Transaction on Information and Systems E90-D*, pages 816–824, 2007.

[TY09] Tomoki Toda and Steve Young. Trajectory training considering global variance for HMM-based speech synthesis. 2009.

[TYM$^+$00] Keiichi Tokuda, Takayoshi Yoshimura, Takashi Masuko, Takao Kobayashi, and Tadashi Kitamura. Speech parameter generation algorithms for HMM-based speech synthesis. 2000.

[TZ09a] Keiichi Tokuda and Heiga Zen. Fundamentals and recent advances in HMM-based speech synthesis. In *Interspeech 2009*, Brighton UK, September 2009.

[TZ09b] Keiichi Tokuda and Heiga Zen. Fundamentals and recent advances in HMM-based speech synthesis. *Interspeech*, 2009.

[TZK03] K. Tokuda, H. Zen, and T. Kitamura. Trajectory modeling based on HMMs with the explicit relationship between static and dynamic features. In *Eighth European Conference on Speech Communication and Technology*. Citeseer, 2003.

[VHH98] P. Vary, U. Heute, and W. Hess. *Digitale Sprachsignalverarbeitung*. B.G. Teubner Stuttgart, 1998.

[WW06] Yi-Jian Wu and Ren-Hua Wang. Minimum generation error training for HMM-based speech synthesis. *ICASSP*, 2006.

[Yam06] Junichi Yamagishi. An introduction to HMM-based speech synthesis. 2006.

[YEG$^+$09] Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying (Andrew) Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Valtcho Valtchev, and Phil Woodland. *The HTK book*. Cambridge University Engineering Department, 2009.

[YMKK99] T. Yoshimura, K. Masuko, T. Kobayahi, and T. Kitamura. Simultaneous modelling of spectrum, pitch and duration in HMM-based speech synthesis. *Proceedings Eurospeech*, pages 2347–2350, 1999.

[YQS09] Zhi-Jie Yan, Yao Qian, and Frank K. Soong. Rich context modeling for high quality HMM-based TTS. *Interspeech Brighton*, 2009.

[YZTW07] Jian Yu, Meng Zhang, Jianhua Tao, and Xia Wang. A novel HMM-based tts system using both continuous HMMs and discrete HMMs. *ICASSP*, 2007.

[Zoe02] Udo Zoelzer. *DAFX Digital Audio Effects*. John Wiley & Sons, Inc., 2002.

[ZON$^+$09] Heiga Zen, Keiichiro Oura, Takashi Nose, Junichi Yamagishi, Shinji Sako, Tomoki Toda, Takashi Masuko, Alan W. Black, and Keiichi Tokuda. Recent development of the HMM-based speech synthesis system (HTS). 2009.

[ZT05] Heiga Zen and Tomoki Toda. An overview of nitech HMM-based speech synthesis system for Blizzard Challenge 2005. In *INTERSPEECH 2005*, Lisboa, 2005.

[ZTB09] Heiga Zen, Keiichi Tokuda, and Alan W. Black. Statistical parametric speech synthesis. *Speech Communication 51*, 2009.

[ZTK04] H. Zen, K. Tokuda, and T. Kitamura. An introduction of trajectory model into HMM-based speech synthesis. In *Fifth ISCA Workshop on Speech Synthesis*. Citeseer, 2004.