Lorenz Gutscher, BSc

Recording, Analysis, Statistical Modeling, and Synthesis of Bird Songs

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Elektrotechnik-Toningenieur

submitted to

Graz University of Technology

Supervisor

Priv.-Doz. Mag.phil. Dipl.-Ing. Dr.techn. Michael Pucher

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

Neulengbach, December 2018

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date	Signature

Abstract

In this thesis bird sounds are investigated with the help of signal processing tools. Established techniques in human speech processing and speech analysis are adapted to the specific characteristics of bird songs. One core aspect is the modeling with Hidden Markov Models (HMM). To set up statistical models, adequate methods for analysis and parameter extraction are examined to make realistic synthesis of bird sounds possible. Using the example of a budgerigar, the process of training and synthesis with the HMM-based Speech Synthesis System (HTS) is described and the results are discussed. Budgerigars have a great ability to produce complex sounds and their songs are accordingly diverse. In order to segment the recordings, an elemental breakdown of phrases is done, as well as a clustering to identify recurring elements. Label files are composed for the use with the toolkit, that contain additional context information to enhance the training and synthesis. The aim of the whole process is to offer an interface, that generates new sequences and compositions of bird songs from a user input. Finally, an objective evaluation comparing the synthesised output to the real recordings is performed.

Kurzfassung

In dieser Arbeit werden Vogellaute mit Hilfe von Techniken der Signalverarbeitung untersucht. Es wird versucht die bisherigen Methoden und Techniken im Bereich der Analyse und Synthese menschlicher Sprache an die Besonderheiten des Vogelgesangs anzupassen. Besonderes Augenmerk wird dabei auf die Modellierung mittels Hidden Markov Modellen gelegt. Hierfür werden adäquate Analysemethoden ermittelt, um statistische Modelle aufstellen zu können und eine möglichst realistische Synthese von Vogellauten zu erzeugen. Am Beispiel des Gesangs eines Wellensittichs wird der Analyse- und Syntheseprozess anhand des HMM-based Speech Synthesis System (HTS) beschrieben und die damit erzielten Ergebnisse präsentiert. Wellensittiche besitzen ein hohes Talent komplexe Laute und Klänge zu erzeugen und ihr Gesang ist dementsprechend abwechslungsreich. Durch Unterteilung einzelner Phrasen in kleinere Elemente wird eine Segmentierung in phonetische Einheiten durchgeführt. Die Elemente der Segmentierung werden in einem Clusteringverfahren gruppiert und gekennzeichnet, um diese mit dem HTS Toolkit verarbeiten zu können. Zusätzlich zu den rein akustischen Faktoren werden für das Training und die Synthese Kontextinformationen verwendet, welche in Form von Entscheidungsbäumen zu einer besseren Modellierung beitragen sollen. Mit dem erstellten Toolkit wird die Erzeugung neuer Abfolgen und Zusammenstellungen von Gesangsphrasen ermöglicht. Schlussendlich werden synthetisch erzeugte Vogellaute mit nicht für das Training verwendeten Originalaufnahmen objektiv verglichen.

Contents

List of Figures

List of Figures

List of Figures

1. Introduction

1.1. Motivation

In the last few years more attention has been drawn to the question of complexity in animal communication and it appears that there is still a lot to find out (Rothenberg et al., 2014). To know more about animal communication can also help to find out more about human speech. On the search for precursors of music and human speech, some answers may be found there with investigation of species with less complex communication systems, that still have similarities to the human one (Marler, 2001). Songbirds are - like human beings - vocal learners and need a tutor to develop more complex vocalisations, while simpler ones are inherent (Thorpe, 1958). Being able to produce realistic sounds by synthesis should give the opportunity to set up experiments with songbirds and find out more about sequencing rules of their songs or how their perception of hearing works. A complete different way to utilise a bird song synthesiser, is the use of it in areas like virtual reality, game design, film and audio productions or even animal assisted therapy (Bonada, Lachlan and Blaauw, 2016, p.1).

1.2. Tasks

The aim of this work is to create a system where synthesised bird sounds can be created on the requirements of the user. Therefore, properties of bird songs are investigated and ways to model them are presented. The HMM-based Speech Synthesis System (HTS) is used to train and model the vocalisations of a budgerigar, whereas the following areas will be covered:

- Ways to describe bird songs, their features and how to derive stochastic models of them.
- Description of the training process in connection with the HTS toolkit.
- Usage of the toolkit to perform training and synthesis of new songs.
- Evaluation of the synthesis and identification of problems that occur in the process.

Budgerigars have a great repertoire of different vocalisations, that can even include imitations of human speech (Dent et al., 1997). Their songs have large variety and especially their complex contact calls do rarely reoccur in similar way twice. On the experimental basis of a segmentation created by Daniel M. Mann (D. M. Mann, personal communication, June 12, 2018) the songs are cut into smaller units. Built on quick changes of the parameters in the audio files (amplitude, fundamental frequency,...) segmentation rules are defined and applied to recordings of one budgerigar specimen. The segmentation algorithm evolved from initial blind segmentation methods (Sharma and Mammone, 1996), whereas specific parameters were found through a comparison to manual derived segmentation of budgerigar signals based on a human visual system. The algorithm was then applied on samples of annotated speech to find the model with the lowest error rate (D. M. Mann, personal communication, June 14, 2018). To use the segmentation in conjunction with HTS there is still a lot to do. First of all, similar sounds need to be grouped together which is done by a clustering method in R (R Core Team, 2014). Before the actual clustering, the segments are divided into two groups - voiced and unvoiced sounds, so that the clustering can be done individually on each set. With the clustering a reduction of more than 30 000 segments to 11 groups of voiced (v1, v2, ... , v10, v11) and 9 groups of unvoiced (uA, uB, ..., uI) sounds is achieved. In a typical scenario the sounds to be synthesised as well as their sequence order have to be specified in a label file. Most of the additional context information can be calculated automatically, while factors like behavioural descriptions can be specified by the user.

1.3. Overview

This thesis is structured into three theoretic chapters and one more practical chapter, where the theory is applied to build a bird voice model. The theory block starts with an introduction of the anatomy of birds and helps to get a better understanding of the mechanics used for sound generation. The second chapter - Recording and Analysis of Bird Songs - deals with ways to attain the caption and storing of acoustical activities and how to process them in a meaningful manner. The last theoretical chapter will focus on the steps that are needed to get meaningful statistical model descriptions in relation with Hidden Markov Model (HMM) based synthesis. This section includes an introduction of HMMs and important tasks and methods, that will be put into practice in the experimental section later on. Finally having reached the experimental part, the HTS toolkit will be used for the training and synthesis of budgerigar sounds. To evaluate the resulting synthesis, a few recordings were retained throughout the training process, so that synthesised versions of unseen input data can be compared to the actual recording. This makes it possible to compare the way the model creates a song with the way that the budgerigar actually created it. Finally, the problems of the method are discussed, a conclusion is drawn, and future refinements are suggested.

2. Bird Anatomy and Vocalisation

In this chapter the avian anatomy that makes the characteristic bird vocalisation possible will be described. To specify the properties of the songs, a division into songs, phrases, syllables, and elements is defined. Using those elements the song structure can be analysed in a more detailed manner, which enables to obtain sequential rules. Moreover, the most common tone qualities and pitch contours are described in detail. At the end of this chapter answers to some of the difficult questions on the existence of language in animal communication are given.

2.1. Anatomy and Mechanics

The complexity of sound production differs a lot between bird species. The underlying physic mechanism of sound production is close to the mammalian one, even though the excitation signal is generated from two different organs and locations. The mechanics of bird sound production can be described with different models, whereas the following seems the most promising: Air is being pressed out from air sacks through the bronchi and syrinx, where tissues (labia) are stimulated so that they vibrate (see figure 2.1). The sound then propagates to the trachea and the larynx. In contrast to mammals, where the larynx is the primary source of sound generation, trachea and larynx operate more like a variable filter, while the syrinx produces the excitement (Mindlin and Laje, 2006, p.37). As just mentioned before, the larynx normally is not used to create sound, but there still exist a few species that use it to generate sibilant and hiss sounds (Bezzel and Prinzinger, 1990).
A different proposed model for sound generation is based on the whistle effect of air being pressed through a valve. While those different models

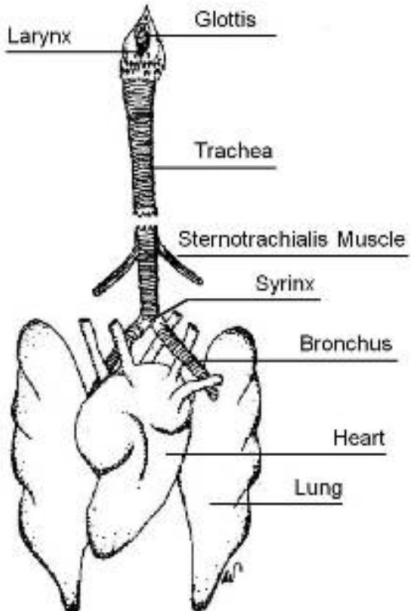


Figure 2.1.: Avian respiratory system (Figure from (Jacob, 2018))

may co-exist, investigation into the whistle model has been made without a clear confirmation of it (Mindlin and Laje, 2006, p.37).

2.2. Syrinx

Taking a deeper look at the main generator of sound, there are numerous differences in the details of the syrinx of each bird. Three main different types of syrinx can be found: the tracheo-bronchial syrinx (see figure 2.2), the tracheal syrinx and the bronchial syrinx. The main force to generate air flow comes from the respiratory muscles. The inhalation/exhalation process is done in a very quick way of about 25 cycles a second (Hummel, 2000, p.117f) and occurs between notes. This enables birds to produce long songs without noticeable pauses. In figure 2.2 (2 = inner muscles, 3 = Membrana tympaniformus lateralis, 4 = M. t. medialis, 5 = Pessulus, 8 = Bronchidesmus, 9 = Tympanum, 10 = Labium laterale, 11 = Labium mediale,

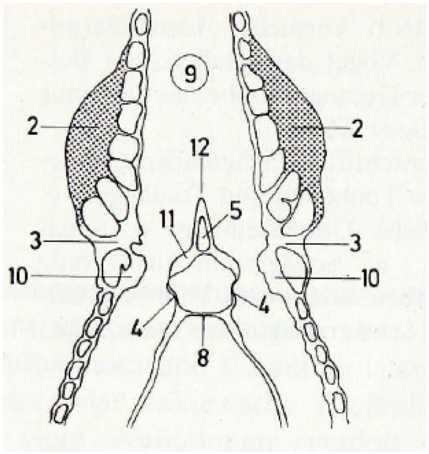


Figure 2.2.: Tracheo-bronchial syrinx (Figure from (Bezzel and Prinzinger, 1990, p.266))

12 = Membrana semilunaris) the skeleton elements can be seen, which consist of membranes and muscles surrounding the syrinx. Around those elements there is a collarbone air-sac that controls part of the pressure that is used for the airflow and sound generation. The vocal folds are mainly used for controlling the pressure by widening or closing the path for airflow (this can be done in a very accurate way by turning skeleton elements into the tube), while the membrana tympaniformis are excited to direct tonal vibrations. The sound level is dependent on the air flow's pressure, but it should be kept in mind, that different sounds may also require different air pressure to be generated (Bezzel and Prinzinger, 1990, p.266f). There are birds like the Clay-colored Robin, that are capable of producing two different tones and timbre at the same time, by using both sides of their syrinx, while others use mainly one (Mindlin and Laje, 2006, p.38f, 59) or even one after another like the brown-headed cowbird (Suthers, 2004, p.288). Neural activities control the movement of abdominal and thoracic muscles for airflow and can create complex movements that finally produce multi-modal vibrations on the membranes.

2.3. Bird Sounds

Bird sounds can be divided into two main categories:

- Vocalisation
- Sonation

While vocalisations make use of the syrinx, sonation is a non-vocal, mechanical production of sound that is intentionally used like special shaped feathers, beak or the feet (Bezzel and Prinzinger, 1990, p.269). In this chapter the focus will be on vocal sounds, that are generated by the respiratory system.
Vocalisations are produced for special purposes and have evolved out of surviving strategies and evolution. A rough division into calls and songs tries to differ between the different intentions of the two, although the differentiation is not always clear.

- **Calls** include vocal sounds, that consist of only one or few elements and for instance involve warnings and alarms. It is of great interest for a bird to know if there is a predator approaching and it is also beneficial to exchange information about food sources and social interaction. The alarm call is normally simpler in relation to bird songs but can for instance include information about the dangerousness of a predator, as well as if it attacks from land or from sky, which will be important for the bird's chosen escape strategy (Templeton, 2005). Interestingly, some alarm calls are perceived inter species as well.
- **Songs** use to be more complex than calls and consist of smaller units called syllables. Their functions include territorial boundary, attracting partners and other communication purposes. During breeding season, songs take an important part in defending territories and competing for partners, where mainly the male birds show off their skills that go hand in hand with their level of attractiveness (in the female bird's view) in terms of health condition. Neither is the production of songs limited to songbirds, nor male birds, setting the Northern Cardinal as an example. Recent studies also claim that there exist many more species where the females sing (Odom and Benedict, 2018).

Finally there is to say, that calls are assumed to be innate, whereas more complex songs are learned by the birds over time individually (Rothenberg, 2007, p.124), (Thorpe, 1958).

2.4. Bird Song Unit

As bird species have very different vocalisations, attempts to standardise the units and their names have been made. To create an initial division of units, a temporal method is sufficient, as only pauses need to be marked. Morphological methods (segmentation decisions based on specific parameter changes) segment complex units, that often are derived trough temporal methods, into smaller elements (Thompson, LeDoux and Moody, 1994). In the following the common structure of bird units and the nomenclature used is described further:
A song is the biggest unit and contains different phrases. More complex phrases can be divided into smaller units called syllables, which again can be divided with a short, temporal derived silence. The smallest units are called elements and evolve from syllables (see figure 2.3). In the following songs will be used to describe both, songs and calls, whereas only "contact" phrases relate to actual songs according to the definition above.

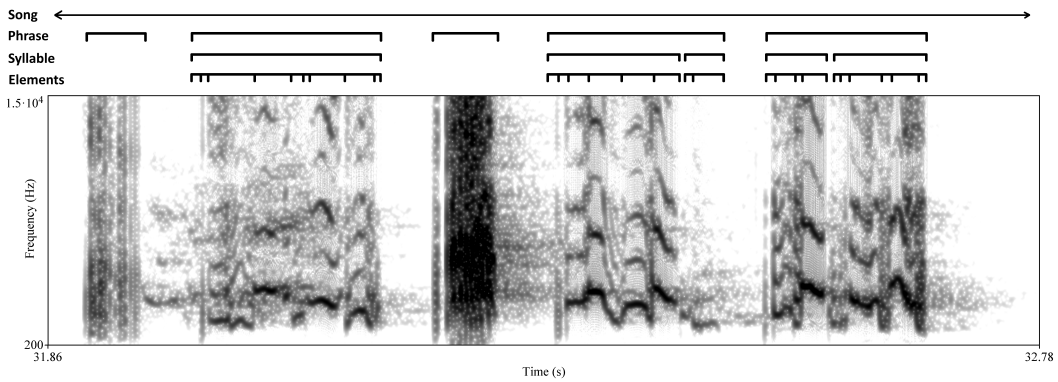


Figure 2.3.: Unit division by the example of a budgerigar song

2.5. Complexity of Song Structure and Sequencing Rules

There are birds that use to mimic hundreds of sounds (like the mocking bird) and there are birds that never really vocalise at all (like storks). Depending on the species, different repetition types of their repertoire can be distinguished. Birds without more than one sound have no variety and keep singing the same song over and over. But for the ones with a bigger repertoire, behaviour of eventual variety and immediate variety is observed. Eventual variety indicates that they repeat the same phrase for a few times and then change to another. In contrast it is referred to immediate variety, if successive phrases are diversified. Both are supposed to have sequencing rules, that arise from the previous syllable or higher order dependencies. An example for higher order dependencies are found in the Bengalese finch (Lonchura striata var. domestica) songs, where the next syllable is not only dependent of the previous one, but on more complex contexts. Even though this higher order dependencies exist, it is still possible to describe sequencing rules with first order HMMs, where each state is only dependent of the previous state. This can be done by a many-to-one state mapping, which basically means that states with higher order sequencing rules are split into different states, while each of them is again only dependent on one previous state. (Katahira et al., 2011).

2.6. Tone Qualities and Frequency Contour

The basic tone qualities that can be found in bird songs are illustrated in figure 2.4. More complex sounds evolve from those basic tones by recombination and the change of speed and repetition (Pieplow, 2017).

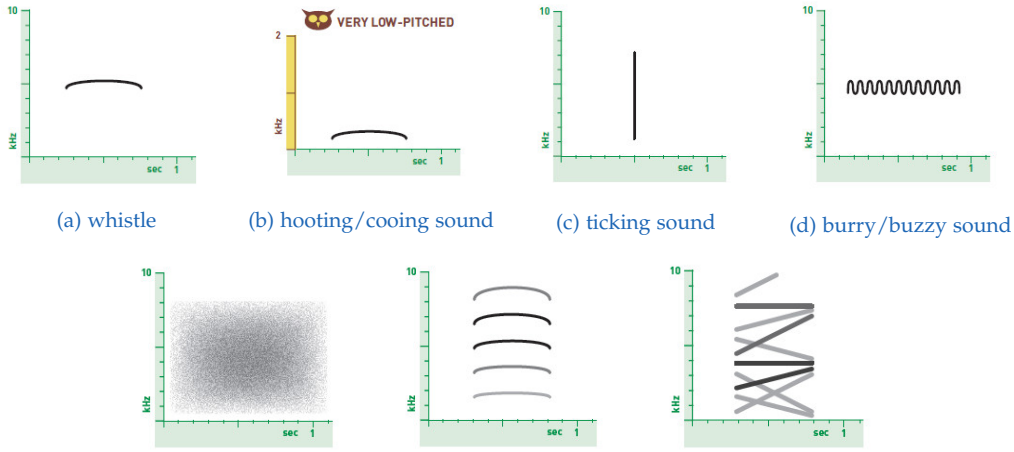
- Whistle sounds:
  Whistle sounds lack strong harmonics and have a higher fundamental frequency as hooting sounds. Birds can actively suppress harmonics by changing their vocal tract.

- Hooting and cooing sounds:
  Those sounds have a lower frequency than most of every other of their vocalisations and are typical for owls.
- Ticking sounds:
  Those sounds are very short, sharp, and broad band noises, without the appearance of a fundamental frequency.
- Burry and buzzy sounds:
  Combining a whistle sound with an extremely quick and strong periodic pitch variation (also referred to as vibrato) creates a buzzer-like sound. At least, this is how humans would describe it, as it is assumed that birds have higher temporal resolution and might hear the frequency contour more detailed.
- Noisy sounds:
  This sounds have a frequency characteristic similar to the tick sound, with the difference of a much longer duration.
- Nasal sounds:
  If the fundamental frequency has less energy compared to the harmonics, this produces a special sound - best described as nasal.
- Polyphonic sounds:
  If birds use both sides of their syrinx they can create polyphonic sounds with two independent fundamental frequencies at the same time.

Combining all those different sounds, birds can create very complex vocalisations. Furthermore, continuous pitch variations can be categorised depending on the trace of the pitch contour as illustrated in figure 2.5:

- monotone
- upslur
- downslur
- overslur
- underslur

Each bird species tends to have a characteristic sound, that results from the creative use of the patterns. These patterns sometimes help humans as well as birds themselves to distinct between closely related species.



(a) whistle (b) hooting/cooing sound (c) ticking sound (d) burry/buzzy sound

(e) noisy sound (f) nasal sounds (g) polyphonic sound

Figure 2.4.: Tone qualities (Figure from (Pieplow, 2017, p.14f))

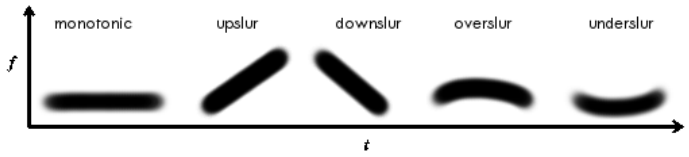


Figure 2.5.: Pitch patterns (Redrawn after (Pieplow, 2017))

2.7. Hearing Abilities of a Budgerigar

To model natural sounds, hearing abilities of budgerigars need to be considered. Studies tried to estimate the birds' critical bands and ratio, as well as their hearing range (Saunders, Rintelmann and Bock, 1979). It was shown, that their hearing ability puts special focus on frequencies around $3 - 4$ kHz, while their hearing range is limited to a bit over 8 kHz, because of their short basilar papilla (membrane) with around 2.5 mm (Manley, Schwabedissen and Gleich, 1993) to 3.7 mm (Saunders, Rintelmann and Bock, 1979, p.320). Budgerigar's temporal resolution of hearing is very high as their threshold to detect gaps in white noise signals is at a duration of roughly 2.5 ms.

2.8. Language in Vocal Communication of Singing Birds

Two concepts are often used to compare human and animal language. More complex language is required to exhibit lexicoding or lexical syntax, which describes the ability to produce new meaning through the recombination of smaller units in a semantically meaningful sentence. Less complex vocal communication systems can recombine meaningless sounds to a new song, which is called phonocoding or phonological syntax. Such recombinations can have syntactic rules as well, but the rules only describe the process of sequencing without giving the new songs a different meaning. While phonocoding is a common phenomenon in song birds and other species like whales, dolphins or bats, there is no evidence of animals that - without human interaction - construct sentences through the application of lexical syntax. Mocking birds, for example, are often mentioned to have great abilities to rearrange their song repertoire to new sequences. Their repertoires consist mainly of short imitation phrases of other birds or even other species. The bird's creative way of splitting up the phrases and combining all those pieces to new songs, sometimes even makes it difficult for scientists to distinguish the origin of sound material the bird is using. Still it is assumed, that all of the beautiful songs have the same meaning, namely representation of their identity, their social status or population membership (Marler, 2001, p.31-48).

3. Recording and Analysis of Bird Songs

To record birds, special equipment and techniques can increase the amount of useful material. This chapter gives an overview on ways to obtain discretised recordings, while paying attention to possible problems throughout the process. Having captured the recordings, ways to visualise bird songs with signal processing tools will be introduced.

3.1. Recording of Birds

Birds often vocalise in groups, which can make it very difficult - if not impossible - to record their voices separately from each other in nature. The acoustic impression of a bird swarm might be interesting as well and can be recorded rather easy, but often the recordings are followed by tasks that need investigations of single vocalisations. To achieve high quality recordings, it is desirable to get close to the singing bird, without scaring it off or causing it to stop vocalising (Budney and Grotke, 1997). The reasons why it is attempted to get as close as possible, are the presence of background noise in nature or loud environments, reverberant environments or the weak signal amplitude itself (for instance when the recording is done from far away).

3.1.1. Field Recordings

Field recordings often need windscreens or wind-shields to reduce low frequent noise (Budney and Grotke, 1997). Still it is of great advantage

to get close to the sound source, as there can be disturbing noises from aeroplanes, cars, wind, water or another competitor singing nearby (Brumm and Naguib, 2009, p.3). The person that is recording the birds can also produce noise by his or her footsteps, moving in noisy clothing or simply by breathing too loud. Early recordings were made with analogue devices or DAT recorders (Wickstrom, 1982, p.29-36), while nowadays smaller devices like hand-held recorders are preferred. Hand-held recorders have enough signal-to-noise ratio (SNR) and good models offer the opportunity to attach an external microphone to it. Headphones make it easy to control the recording and check the quality of the captured sounds directly. To reduce the problem of background noise, an external shotgun microphone is a good choice to capture more of the bird's sound. Shotgun microphones have strong directional effects and therefore reduce noise, coming from the side or back of the microphone. Another possibility to increase the quality in the recording process, is the use of a parabolic reflector. With a parabola only sound from the pointed direction is amplified making it possible to create very sterile and clear recordings even in problematic higher frequency ranges. The amount of amplification depends on the frequency of the source and increases with high frequencies. This can be useful, as high frequencies have greater loss over distance due to air dissipation. For an example of 22 inch (25.88 cm) the amplification starts at 200 Hz and increases with around 6dB/octave (Wildtronics, LLC, 2017). The outcome of a parabola recording method is rather sterile and makes it possible to hear sounds, that could barely be heard directly with human ears. The disadvantage of this method is that small parabolas do not amplify low frequencies and that the sound of a moving bird is difficult to capture exactly, as the pointing direction has to follow the source precisely. Furthermore, the sterile sound can seem unnatural in relation to gunshot microphones, that capture more reverberation.

3.1.2. Lab recordings

By lab recordings, we mean recordings being made inside houses and rooms, in contrast to recordings outside in the natural habitat. Recordings made in rooms without absorbers or other acoustic optimisations tend to have more

reverberation than outdoor recordings and might need post-processing to reduce that effect. A way to decrease reverberation in advance is a proper selection of the microphone being used, like a cardioid or shotgun microphone, placed as close to the animal as possible. To put a microphone close to a bird may change its behaviour, but with continuous training, some birds seem to get used to it very well (see figure 3.1). Another way to get

Figure 3.1.: Budgerigar recorded with a shotgun microphone in the budgerigar laboratory Vienna (Figure from (Mann, 2018))

the microphone close to the bird is the use of a backpack-like construction attached to their body. The construction consists of a microphone, a circuit board with a wireless transmitter and a battery - all held together with a harness, that has to be put around the bird to affix it. A promising example seems to be the very lightweight backpack developed by a research group at the Max Planck Institute for Ornithology (Gill et al., 2016). With the device it is not only possible to get more isolated recordings in groups of birds, but also to identify which individual actually made the sound. Also for this technique, it should be considered throughout experiments, that attaching a device to the bird's body may influence its behaviour in some way.

3.1.3. Problems and Solutions

Even if the equipment and setting is perfectly build up, there can be interfering factors. As birds are living creatures, they often jump around in cages, let their wings flutter or clap rhythms with the use of their beak. Those factors cannot be controlled and need post-processing depending on the further use of the recording.
In the following, when talking about recordings, it is supposed that these have already been discretised and sampled. In case of our experimental data, a sampling rate of 48 kHz and 16 Bit per Sample will be used, as this represents the original quality chosen for the recording and has a theoretical bandwidth of $0 - 24$ kHz, which is sufficient for frequencies in the audio spectrum. In the experiment, the analysis window of a frame is set to eight milliseconds, so that quick temporal changes can still be represented. The frame shift is set to one millisecond, therefore also short segments have enough observations to be modeled.

3.2. Feature Modeling and Extraction

Depending on what aspects to focus on, there are different methods to extract the information needed or at least describe best what is happening in a bird song. Different methods to capture adequate parameters were experimented with and will be described in the following section.

3.3. F0/LF0

The fundamental frequency, also called F0 or LF0 (logarithmic fundamental frequency) if the logarithmic representation is used, is the lowest present sinusoidal frequency of a periodic sound. A common way to describe different phrases of bird songs is to focus on the change of the F0 value over time. To detect the correct value different approaches can be taken and there is a lot of literature about the detection rate and comparison between different pitch trackers (O'Reilly and Harte, 2017). In the following section

the most common methods and their theory will be described in more detail.

- Zero crossing method:
  A straightforward method to measure F0 of a simple waveform like a sinusoid is to measure the zero crossings to get the wave period and then calculate the inverse of it in order to obtain the frequency. The method is cheap to implement, but only applicable if there is a pure waveform without noise and with only one occurring frequency.
- Autocorrelation:
  Rather than just measuring the zero crossings, many pitch detection algorithms (PDA) use a more advanced method called auto correlation to get the F0 contour. If $x(t)$ is a continuous time signal at time t and τ is the time lag between the signal and its copy, the autocorrelation is defined as

$$r_t(\tau) = \int_{-\infty}^{\infty} x(t)x(t+\tau)dt \tag{3.1}$$

  In case of discretised signals, the integral can be written as a sum in the following way

$$r_t[\tau] = \sum_{j=t+1}^{t+W-\tau} x_j x_{j+\tau} \tag{3.2}$$

  with W being the window size, t and $r_t[\tau]$ the autocorrelation with lag τ at the time index t (Cheveigné and Kawahara, 2002). The output of the formula gives the similarity between the signal and its delayed copy in a number range from 0 to 1, whereas 0 means no correlation and 1 total correlation. For a delay of $\tau = 0$ the autocorrelation is always 1. If there exist global maxima aside from $\tau = 0$, there exists a period and the fundamental frequency F0 can be calculated as $F0 = 1/\tau$ where τ is the time-lag of the maxima. Harmonic and noisy sounds make the selection of the fundamental frequency more difficult, as there exist more candidates, that could be chosen. In special cases, where single sub-harmonics of an investigated sound have higher energy than the lowest fundamental frequency, the choice for the F0 candidate is even harder. Therefore further refinements of this method improve the detection rate, for instance by taking the human perception of frequencies into consideration. In reality, there are numerous

sounds in in bird songs, with a strong first harmonic (Pieplow, 2017, p.20). Without refinements of the PDA this may lead to problems to detect the frequency contour of the fundamental (Huang, Acero and Hon, 2001, p.327f). Autocorrelation has the disadvantage of rather big analysis windows (Talkin, 1995, p.504). Its implementation within Praat, which we used in our analysis works very well together with the refinement possibilities (see chapter 5.0.2 and figure 5.3).

- Cross-correlation or modified autocorrelation:
  To avoid the reduced integration window with lag τ, the modified equation can be written as

$$r_t[\tau] = \sum_{j=t+1}^{t+W} x_j x_{j+\tau} \tag{3.3}$$

  This reduces the choice of sub-harmonics as the fundamental frequency but may also introduce some octave errors. The maximum of the modified autocorrelation no longer has to be at lag $\tau = 0$. The pitch detection algorithm called YIN (Fundamental frequency estimator for speech and music) is based on the cross-correlation (Cheveigné and Kawahara, 2002) and improvements special made for bird songs are developed in the YIN-bird version of it (O'Reilly and Harte, 2017).
- Normalised cross-correlation:
  This adaptation of the cross-correlation presents an improvement of the original function especially for fast changing signals x_j, with only little increased computational cost.

$$\phi_t[\tau] = \frac{\sum_{j=t+1}^{t+W} x_j x_{j+\tau}}{\sqrt{\sum_{j=t+1}^{t+W} x_j^2 \sum_{j=t+1}^{t+W} x_{j+\tau}^2}} \tag{3.4}$$

  RAPT (Robust Algorithm for Pitch Tracking) is an example for an algorithm that uses the normalised cross-correlation for its calculation (Talkin, 1995, p.505f). The algorithm requires the input of a minimum and maximum fundamental frequency.

After the calculation the next step is post processing, where the best candidate for the fundamental frequency is chosen or the result is combined with further methods. Advanced pitch detectors give the possibility to make rules for the choice of the best candidate. Parameters like octave-jump cost take the neighbouring values into account to reduce octave-jumps within short periods. The logarithmic frequency scale makes frequency changes more intuitive, as it is closer to human pitch perception and the musical scales used in western music, where for instance one octave higher corresponds to a doubling of the frequency (Klapuri, 2003, p.813ff).

3.4. Spectrum and Windowing

A waveform itself is not easy to interpret in regards of frequency representation and there are better ways to represent the characteristics of a signal. Waveforms contain phase information that is less important for human speech perception and is often removed in spectral analysis (Taylor, 2009, p.156). Putting together the spectra of chunks of audio consecutively a spectrogram arises, representing the frequency power spectrum over time. The process of concatenating the waveform in order to look at small chunks of audio data is done by a multiplication of the full waveform with a window function $w[n]$:

$$x[n] = w[n]s[n] \tag{3.5}$$

The simplest window is a rectangular window function.

$$w[n] = \begin{cases} 1 & \text{if } 0 \leq n \leq L-1 \\ 0 & \text{else} \end{cases} \tag{3.6}$$

Improvements to this method can be achieved by using different windows like a Hanning window

$$w[n] = \begin{cases} 0.5 - 0.5cos(2\pi N/L) & \text{if } 0 \leq n \leq L-1 \\ 0 & \text{else} \end{cases} \tag{3.7}$$

or a hamming window.

$$w[n] = \begin{cases} 0.54 - 0.46cos(2\pi N/L) & \text{if } 0 \leq n \leq L-1 \\ 0 & \text{else} \end{cases} \tag{3.8}$$

To transform the signals from time domain to the frequency domain the discrete Fourier transform (DFT) is being used in theory, while the fast-Fourier-transform is preferred in practical cases, due to its quick calculation speed (Taylor, 2009, S.342). The DFT is defined as

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N} \qquad k = 0, 1, 2, ..., N-1 \tag{3.9}$$

and its inverse by

$$x_k[n] = 1/n \sum_{k=0}^{N-1} X_k e^{j2\pi kn/N} \qquad n = 0, 1, 2, ..., N-1 \tag{3.10}$$

Having the spectrum calculated with DFT it can now be represented either by its real and complex parts or by magnitude and phase. As mentioned before, the human ear is less sensitive to phase information (Saratxaga et al., 2012) and therefore it seems obvious to use magnitude in the frequency domain. As the human ear's perception of sound amplitude is approximately logarithmic, the amplitude is normally logarithmic and by convention the log power spectrum is being used.

3.5. Cepstrum

The cepstrum is a convenient way to decouple source and filter of a sound. To convert waveforms into cepstrum the inverse DFT of the logarithmic magnitude of the DFT of a signal is being calculated.

$$c[n] = \mathcal{F}^{-1}\{log|\mathcal{F}\{x[n]\}|\} \tag{3.11}$$

With the source described as the source signal u[n] and the vocal tract (together with a radiation filter) as the signal v[n] we can get the time domain equation as

$$y[n] = u[n] \circledast v[n] \tag{3.12}$$

Undertaking a Fourier transformation this looks as follows:

$$Y(e^{jw}) = U(e^{jw})V(e^{jw}) \tag{3.13}$$

Using the logarithm of that let us split the term into a sum (Furui, 2000, p.64f)

$$log(Y(e^{jw})) = log(U(e^{jw})V(e^{jw})) \tag{3.14}$$

$$log(Y(e^{jw})) = log(U(e^{jw})) + log(V(e^{jw})) \tag{3.15}$$

Using the inverse DFT (IDFT) we now get back to the time domain representation, where we have a simple addition of source and filter components.

$$c[n] = c_u[n] + c_v[n] \tag{3.16}$$

(Taylor, 2009, p.355)

3.6. Mel-Generalized-Cepstrum (MGC)

A common approach in speech analysis in order to model speech sound is to use Mel-generalized-cepstrum analysis (Tokuda, Kobayashi et al., 1994). Figure 3.2 shows a unified view of speech analysis methods, that arise from the choice of specific values for the parameters α and γ. It can be seen, that the choice of $\alpha \neq 0$ and $-1 < \gamma < 0$ leads to the area of "Mel-Generalized Cepstral Analysis". With that method each frame consists of Mel-frequency cepstral coefficients (MFCC) of desired order. To obtain the coefficients the following steps have to be made (see figure 3.4): At first the signal has to be windowed to get frames. This can be done by using a proper windowing function like the Hamming window. Next the frames are Fourier transformed, which is in practice normally done by using the fast-Fourier transformation for reasons of computational cost. Now a Mel-scale filterbank is applied, which consists of bandpass filters, aligned in a

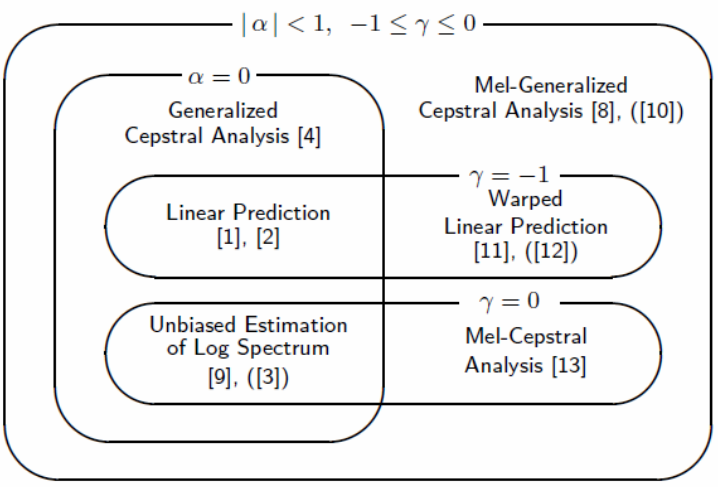


Figure 3.2.: Unified view of speech analysis methods (Figure from (Tokuda, Kobayashi et al., 1994))

non-linear fashion on the frequency axis to create equal resolution of the bands matching human hearing.

$$Mel(f) = 2595 * log_{10}(1 + \frac{f}{700}) \tag{3.17}$$

The Mel-scale describes the perceived pitch of sinusoidal tones and their relative change in frequency for human beings. In figure 3.3 the windows of the filter banks and their position on the frequency axis can be seen. After filtering the signal with the Mel-filter-bank the logarithm of the values is being taken. To decorrelate coefficients it is transformed with the discrete cosine transformation. When mentioning MGC parameters in the following, this should refer to the MFCC parameters.

3.6.1. Non-Negative Matrix Factorisation

As mentioned before, MFCC might not be the optimum solution for the analysis and modeling of bird sounds, because the Mel-scale was created to fit human perception. A different approach would be the non-negative matrix factorisation (NMF), that has been used in areas like speech and

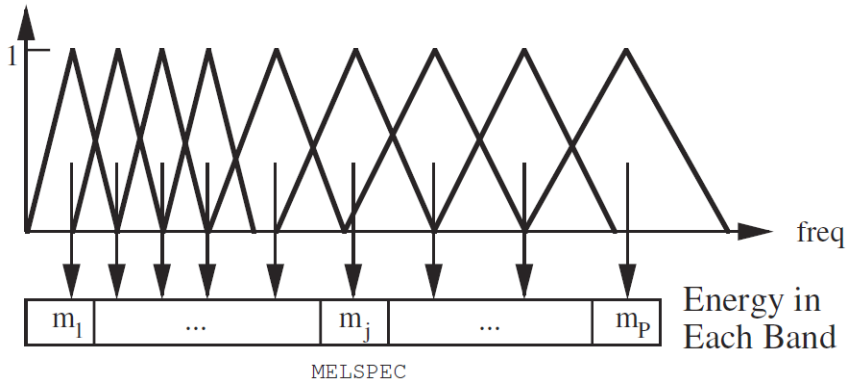


Figure 3.3.: Mel-scale filterbank as it is used in HTK (Figure from (Young et al., 2015, p.95))

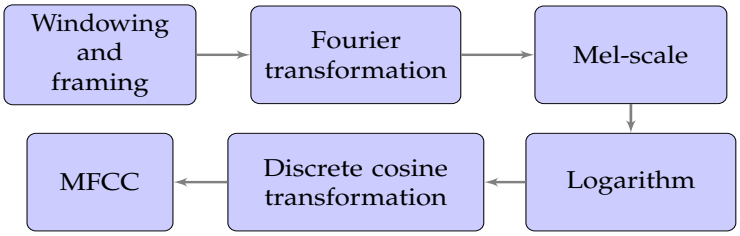


Figure 3.4.: Flow chart to obtain MFCC features

audio with good results as it provides a good representation of the material. Short-time feature extraction, as used by MFCC, can be substituted by NMF cepstral-like coefficients, where filter banks are learned in an unsupervised manner during a training process by inspecting the spectrogram. This means, that the MFCC and NMF coefficients are obtained very similar, whereas MFCC uses the Mel-scale for its auditory filter bank and NMF uses the filter banks obtained from the training process. Figure 3.5 shows the process of obtaining the NMF coefficients in a bird species classifier (Ludeña-Choez, Quispe-Soncco and Gallardo-Antolín, 2017).

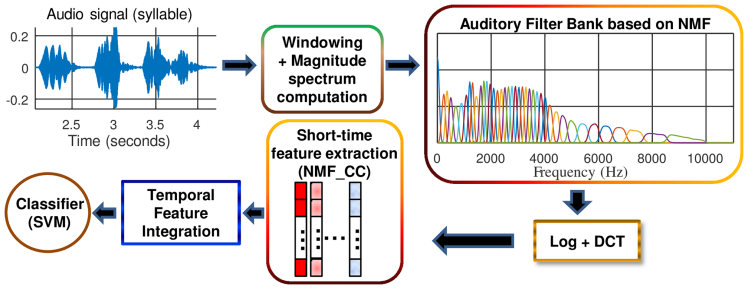


Figure 3.5.: Block diagram presenting the process of obtaining NMF filter banks (Figure from (Ludeña-Choez, Quispe-Soncco and Gallardo-Antolín, 2017))

4. Statistical Modeling

In this chapter the definition of observation vectors and how they are used in statistical models will be discussed. After the presentation of general terms commonly used for Hidden Markov Models, basic methods used for the training of HMMs will be explained. Finally, different ways to cluster the segmented samples and how to interpret the cluster results will be described.

4.1. HMM Based Synthesis

Synthetic speech can be created for different purposes and the result depends on the need of the application. Therefore, it is important to know what aspects of speech we want to optimise throughout the whole process. One aspect might be naturalness of the synthesised speech, another could be intelligibility. There is very little knowledge about what aspects birds focus on, while they hear vocalisations and only assumptions can be made. To model bird songs by means of speech synthesis tools, we need to ensure, that the underlying techniques can be applied there as well. A major component of a statistical parametric speech synthesis system is the vocoder which often is based on the source filter model (see figure 4.1)

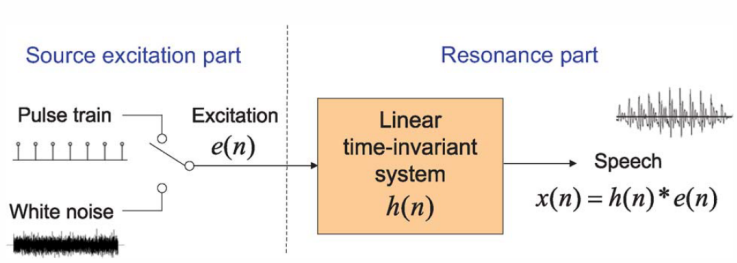


Figure 4.1.: Source filter model for human speech generation (Figure from (Tokuda, Nankaku et al., 2013))

The source filter model is an abstraction of human voice production, where excitation is modeled through switching between a noise signal and a pulse train. In analogy to human anatomy, the excitation part corresponds to the larynx. The signal is then filtered to incorporate the filtering that takes place in the vocal tract, as well as in lip radiation. Comparing it to sound production of birds strong similarities can be found. It was already discussed, that birds generate sounds not with the larynx, but with their syrinx. Still the production process can be compared to that of humans, with the distinctive feature that birds have two separate controllable excitation sources. The filtering done by the source tract is very similar to the birds', as the sound waves travel further through trachea, while lip radiation could be equated to the bird's beak. The Vocoder will derive the waveform from fundamental frequency, spectral envelope and voicing condition. Therefore, we need to provide this information in the training process by forming so called observation vectors. These do not only contain static information about the current frame, but may also include dynamic features, with time derivatives typical in first and second order. The content of the vectors can be separated into excitation and spectral parts. An example of an observation vector at one specific frame can be seen in figure 4.2. To calculate dynamic features, the static feature changes of neighboured observations are compared and for example Δc_t can be written in the following simple form:

$$\Delta c_t = c_t - c_{t-1} \tag{4.1}$$

Figure 4.3 shows the structure of a HMM speech synthesis system, that creates speech from a text input using the information obtained from the

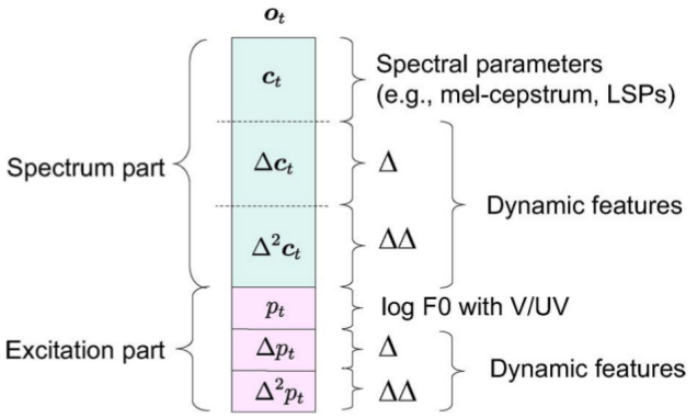


Figure 4.2.: Observation vector of one frame (Figure from (Tokuda, Nankaku et al., 2013, p.1236))

training. The architecture can be applied to bird synthesis as well, but text input has to be modified into the correct format specified in table 5.6.

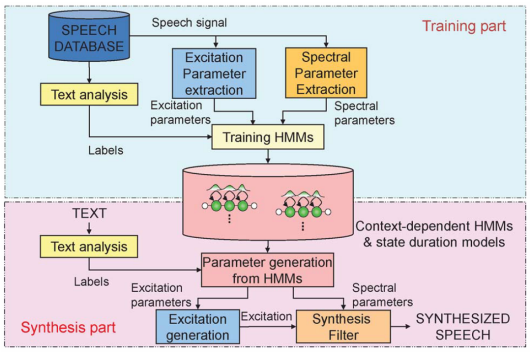


Figure 4.3.: Typical architecture of HMM based speech synthesis system (Figure from (Tokuda, Nankaku et al., 2013))

4.2. Hidden Markov Model

Markov-chains are sequences of states, whereas the probability of one state only relies on the previous one. Therefore, transition probabilities can be

established as seen in figure 4.4. With the knowledge of these transition probabilities, predictions about future events (states) can be made.
A *hidden Markov model* is a *Markov-chain* that includes states that are not

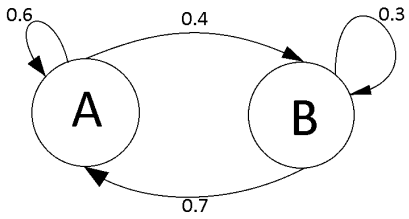


Figure 4.4.: Example of a Markov-chain with two states

observed directly but rather are hidden, while only their emission can be seen. The task is to find probability functions of the hidden states through their emissions, that can be observed and measured. To describe better what was just explained, consider the following example, which is often used in relation to HMM:
A guy named John has three main activities: Taking a walk, shopping, and cleaning ({Walk, Shop, Clean}). Depending on the weather ({Rainy, Sunny}) he has tendencies of choosing which of these activities he will do each morning (see figure 4.5). In the daily telephone conference with a friend he always tells what activity he did today. The friend knows John's preferences on rainy and sunny days and also knows the weather trends in John's area. Even though the friend cannot observe the weather directly, he can calculate the possibility of it through John's activities and can try to guess how the weather probably is on that day.

In the case of my studies the HTS toolkit was used and therefore the notation of the closely related HTK book (Young et al., 2015) will be used for further explanations. An example of a simple left-right 5-state (N=5) HMM like it is used in HTS can be seen in 4.6. The observations o are generated from different states and allow to calculate probabilities that tell from which state they might have evolved. In the example shown the entry state (j=1) and the exit state (j=5) do not emit any observations, which is the standard in HTK and HTS. That means, that only three output probability distributions $b_2(), b_3(), b_4()$ can be put up.

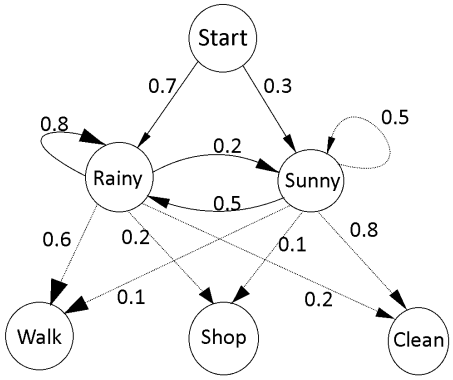


Figure 4.5.: Example of a simple HMM

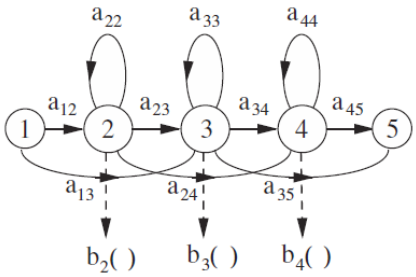


Figure 4.6.: HMM model with 5 states (Figure from (Young et al., 2015, p.128))

The transition matrix will be a $5x5$ matrix, whereas the 5^{th} row only contains zeros. As a rule, all other rows need to sum to 1.

$$\sum_{j=1}^{N} a_{ij} = 1 \qquad j = \{1, 2, ..., N\}$$

with i = {1, 2,... ,N-1}.
The transition probabilities a_{ij} are non-negative terms.
A small variation of the HMM is the Hidden semi-Markov Model (HSMM), where the duration of a hidden state is dependent on the time elapsed since the transition into that state. In a HMM the probability of a state transition is constant. The HSMM is an effort to improve the state duration modeling especially for durations that are not normal distributed (see subsection 4.2.1).
To explain the training of HMMs, let $\mathbf{o} = [\mathbf{o}_1^T, \mathbf{o}_2^T, , ..., \mathbf{o}_T^T]^T$ be the observation

vectors and their corresponding speech parameters and W be the context information, which is incorporated by the label files of the training data. The general equation for the training process can be written as follows:

$$\lambda_{max} = \arg\max_{\lambda} p(\mathbf{o}|\lambda, W) \tag{4.2}$$

with

$$p(\mathbf{o}|\lambda, W) = \sum_{\forall \mathbf{q}} \pi_{q_0} \prod_{t=1}^{T} a_{q_{t-1}} b_{q_t}(\mathbf{o}_t) \tag{4.3}$$

where $\mathbf{q} = \{q_t, q_2, ..., q_T\}$ is a state sequence (Tokuda, Nankaku et al., 2013, p.1236).
Relation to birdsong:
In speech synthesis the hidden states are phonetic symbols, while the acoustic sound and its representation are the observations or emissions. For song birds there might not be a fully discovered list of phonetic symbols, but attempts have been made to distinguish and label different sounds (see 4.7).

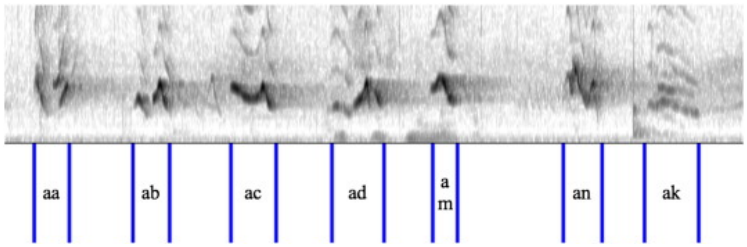


Figure 4.7.: Annotated spectrogram of a song by a black-headed grosbeak (Figure from (Arriaga et al., 2015))

Models built on that kind of syntax will firstly consist of a set of monophone HMMs, where for each phone a model is created. Each state then models different parts of what is happening acoustically during such a phone. That means that the duration of each state might vary a lot, if there are parts with abrupt changes and parts with little change. In case there are more instances of the same phone that are not exactly the same, the different model parameters will vary. For the initial steps still all of the same phones will be modeled through one model to get an overall description of it, before the models will be retrained with further refinements.

4.2.1. Gaussian Mixture Models

Mixture Gaussians are a combination of different Gaussian (normal) distributions and can be used to describe any kind of density function. To combine the different distributions each of them is multiplied with a weight factor c_m before the summation to normalise the result and to maintain, that the integral over the whole function is one. A Gaussian distribution can be written as $\mathcal{N}(\mu, \sigma^2)$, where μ is the mean or expectation, σ is the standard deviation and σ^2 the variance.

Figure 4.8.: Mixture Gaussian of two Gaussian distributions (Figure from (Turner, 2017))

Coming back to the example of a left-right HMM, the probability $b_j(o)$ of an observation sequence o can be written as

$$b_j(o) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(o : \mu_{jm}, \Sigma_{jm}), \qquad 1 \leq j \leq N \tag{4.4}$$

whereas j is the state number, M is the number of different Gaussian distributions, and Σ_{jm} is the co-variance matrix (Young et al., 2015, p.128, slightly simplified).
HTS uses mixture models together with multi-space probability distribution for F0 modeling, so that unvoiced regions can be described as zero-dimensional observations (Tokuda, Zen and Black, 2002). In figure 4.8 two Gaussian distributions and their combined two component mixture model

can be seen as an example for a Gaussian Mixture Model. A common task is to find the underlying individual Gaussian distribution, given a mixture model. A common method in statistic to solve this task is the use of the Expectation-Maximization (EM) algorithm. This algorithm was created to find the parameters of a statistical model based on given observations. In the case of mixture models, it is used to separate the data points into different classes, by calculating the probability of a data set to belong to one or another Gaussian distribution. The initial start is done by arbitrary Gaussian distributions and then the algorithm toggles between the two operation methods "Expectation" and "Maximization". During the "Expectation" step a log-likelihood expectation is calculated, that is being maximised in the second step. This procedure is described more detailed in the following:

- Expectation:
  With the knowledge of the parameters μ_m. σ_m^2 and c_m the expectation of a data point x_i belonging to class c_k:

$$r_m^{(i)} = \frac{p(x_i, C = c_m)}{p(x_i)} = \frac{c_m \mathcal{N}(x_i : \mu_m, \sigma_m^2)}{\sum c_m \mathcal{N}(x_i : \mu_m, \sigma_m^2} \tag{4.5}$$

- Maximisation: In this step the new parameters are changed so that the mean of a mixture model is moved to the direction of the highest responsibility.

$$\hat{\mu}_m = \frac{\sum_i r_m^{(i)} x_i}{\sum_i r_m^{(i)}} \tag{4.6}$$

$$\hat{c}_m = \sum_i \frac{r_m^i}{N} \tag{4.7}$$

The algorithm is normally stopped at a time when the improvement is only little.

4.2.2. Forward Algorithm

Given a HMM, this algorithm calculates the probability for a specific observation using dynamic programming methods. First the forward-variables

$\alpha_j(t)$ are introduced, that contain the probabilities of being in state $1 < j < N$ at time $1 < t \leq T$ and having seen observations $o_1 o_2 ... o_t$.

$$\alpha_j(t) = \left[\sum_{i=2}^{N-1} \alpha_i(t-1) a_{ij} \right] b_j(o_t) \tag{4.8}$$

For the initial condition we use

$$\alpha_j(1) = a_{1j} b_j(o_1) \tag{4.9}$$

$$\alpha_1(t) = 1 \tag{4.10}$$

For the special case of having reached the final condition we write

$$\alpha_N(T) = \sum_{i=1}^{N-1} \alpha_i(T) a_{iN} \tag{4.11}$$

4.2.3. Backward Algorithm

This procedure is very similar to the forward algorithm but calculates the probability of being in state $1 < i < N$ at time t and seeing the sequence $o_1 o_2 ... o_t$ next (Wunsch, 2001, p.10f).

$$\beta_i(t) = \sum_{j=2}^{N-1} a_{ij} b_j(o_{t+1}) \beta_j(t+1) \tag{4.12}$$

The initialisation is done by

$$\beta_i(T) = a_{iN} \tag{4.13}$$

with the final condition:

$$\beta_1(1) = \sum_{j=2}^{N-1} a_{1j} b_j(o_1) \beta_j(1) \tag{4.14}$$

4.2.4. Baum-Welch Re-Estimation

Once the overall parameters of a HMM are set, the next step is to create more accurate models and update the HMM. For that purpose, HTK uses the Baum-Welch Re-estimation, which was named after Leonard E. Baum and Lloyd R. Welch and is a version of the EM algorithm (Huang, Acero and Hon, 2001, p.387f). With the model λ and output sequence o given, we want to maximise the probability of the training data:

$$arg\max_{HMM} P(\mathbf{o}_i|\lambda) \tag{4.15}$$

With $\zeta_{i,j}(t)$ being the transition probability from state i to state j at the time t, the update can be calculated as:

$$\zeta_{i,j}(t) = \frac{\alpha_i(t)a_{i,j}b_j(o_{t+1})\beta_j(t+1)}{P(\mathbf{o}|\lambda)} = \frac{\alpha_i(t)a_{i,j}b_j(o_{t+1})\beta_j(t+1)}{\sum_{i=2}^{N-1}\sum_{j=2}^{N-1}\alpha_i(t)a_{i,j}b_j(o_{t+1})\beta_j(t+1)} \tag{4.16}$$

To calculate the state occupation probability $\gamma_{i,j}(t)$ we need to sum up the transition probabilities over the state numbers j:

$$\gamma_i(t) = \sum_{j=2}^{N-1}\zeta_{i,j}(t) \tag{4.17}$$

The models are improved, until a local maximum is reached for $P(\mathbf{o}_i|\lambda)$. Based on the initial parameters, the local maximum might not be the best model, which is why in practice the steps are repeated with different parameters (Deller, Jr., Hansen and Proakis, 1999, p.703f) (Furui, 2000, p.288ff).

4.2.5. Viterbi Algorithm

The Viterbi Algorithm provides the most likely hidden state sequence given an output vector o and the model parameters λ. It is closely related to the Forward Algorithm but instead of calculating previous states through summation, maximisation is used. In our work the Viterbi algorithm is used for forced alignment during the training process. The introduced

variable θ incorporates the maximum probability at time $1 \leq t \leq T$ and the observation $o_1 o_2 ... o_t$ to end up in state s_i after running through a sequence with the length of t.

$$\phi_t(i) = \max_{q_1, q_2, ... q_t} P(o_1 o_2 ... o_t; q_1 q_2 ... q_t | \lambda) \tag{4.18}$$

The variable $\psi_t(i)$ contains the preceding states that were involved in getting the maximum probability. For initialisation we don't need a maximisation of preceding states as there are none at the starting point in state o at time 1.

$$\phi_1(i) = a_{0i} b_i(o_1), \qquad 1 \leq i \leq N \tag{4.19}$$

$$\psi_1(i) = 0 \tag{4.20}$$

Next we can use recursion to calculate the maximum likelihood of being in state j at time $t+1$ and seeing observation o_{t+1}

$$\phi_{t+1}(j) = \max_{1 \leq i \leq N} [\phi_t(i) a_{ij}] b_j(o_{t+1}), \qquad 2 \leq t \leq T, 1 \leq j \leq N \tag{4.21}$$

$$\psi_{t+1}(j) = \underset{1 \leq i \leq N}{arg\,max} [\phi_t(i) a_{ij}], \qquad 2 \leq t \leq T, 1 \leq j \leq N \tag{4.22}$$

The termination equation can be written as

$$P^* = \max_{1 \leq i \leq N} [\phi_T(i)] \tag{4.23}$$

and the maximum final state probability is

$$q_T^* = \underset{1 \leq i \leq N}{arg\,max} [\phi_T(i)] \tag{4.24}$$

With the following equation, finally the path of the most likely state sequence can be calculated.

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad 1 \leq t \leq T - 1 \tag{4.25}$$

4.3. Clustering

In the most-spoken human languages well described phone sets exist, containing all the different phones of that language. If the language has not been fully studied yet and there is no knowledge about the amount of existing phones, the first step is to find all different sounds that occur. In our case an initial segmentation of budgerigar songs was already done and available, but a relation between the segments had to be built. To obtain a rough estimation of similarities between the segments and their sounds clustering can help to find a solution. Clustering is a process that groups together data points which have something in common, based on respective attributes. The following section describes an attempt to do this in theory and in section 5.2 the results of the clustering used in the experiment will be presented.
At first, the question of the number of different components that the analysis should provide has to be handled. While most analysis methods need further definition of the desired number of clusters, there are methods that try to estimate the optimal number of clusters without a user specifying it. Clustering can be split into the following parts (Halkidi, Batistakis and Vazirgiannis, 2001, p.2f):

- Feature selection: Not all of the features might be helpful for finding good clusters and normally some pre-processing of data needs to be done, before it can be used in a proper manner
- Choice of algorithm: There are many different algorithms with their own advantages and disadvantages. Distance measurement and cluster criteria describe the distinct functionality best. The first one handles the question about the definition of neighbouring, while the second one considers what is called a cost function.
- Validation: To ensure, that the result is meaningful, a validation following some criteria needs to be done.
- Interpretation: With everything done, the results will normally be used for further experiments and need to be integrated back into the work environment.

4.3.1. K-Means

One popular clustering method is K-means clustering, where K describes the number of cluster centroids. The task is to find K mean vectors $(\mu_1, \mu_2, ...\mu_3)$ that distinguish the data points in the sense of their proximity measure.

$$Cost function: \qquad J = \sum_{i=1}^{k} \sum_{x_j \epsilon S_i} d(x_j, \mu_i) = \sum_{i=1}^{k} \sum_{x_j \epsilon S_i} ||x_j - \mu_i||^2 \tag{4.26}$$

with $d(x_j, \mu_i) = ||x_j - \mu_i||^2$ being the Euclidean distance between a point x_j and μ_i and with S_i being a cluster set $(S_1, S_2, ...S_k)$. The initialisation of cluster centroids is done by using random data points and assign each data point to its nearest cluster. Then the cluster centroid is set to the mean of the cluster and the data points are re-assigned. This process is repeated.

4.4. Decision Trees

Decision trees are a supervised method for classification and regression (Breiman et al., 1984). It consists of branches, decision nodes, and leaf nodes with a node question about an attribute that can be answered with "yes" or "no". The branches then continue to the next question and so on until a final leaf node is reached. Decision trees can help to reduce the amount of training data needed. As the amount of possible phone combinations is huge, it is very likely that not all combinations will occur in the training set and that some combination do only occur rarely. In order to get stable models for rare and unseen combinations, different labels that share some part of context information are merged together to a robust cluster of acoustic qualities. In the HTS toolkit this process is also referred to as state tying where every state has its own decision tree.

4.5. Dynamic Time Warping

With the use of dynamic time warping (DTW) it is possible to find similarities between two time-dependent sequences. It enables to match two sequences, even if their time alignment varies in speed, as it can be seen in figure 4.9.

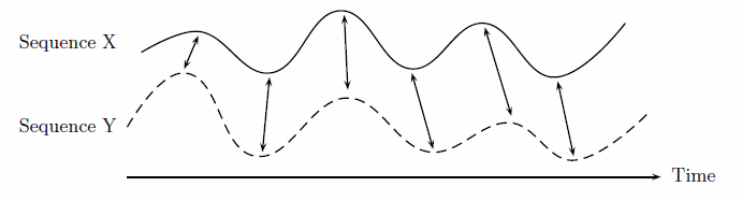


Figure 4.9.: Alignment of two time-dependent sequences (Figure from (Müller, 2007))

In human speech, phone duration can vary for the same phone in different contexts and has never exactly the same length. It is very likely that this also happens in budgerigar sounds. In a paper dealing with the synthesis of chaffinch birds based on HMMs, it is suggested to find similar syllable elements with the help of DTW, instead of labelling them manually by experts (Bonada, Lachlan and Blaauw, 2016, p.3). The procedure that could be used to execute a clustering using DTW is described in the following:

- 1. The first decision that has to be made is the feature parameter selection to compare the similarity of time aligned data. F0/LF0 parameters as well as more complex features like spectrogram, linear predictive coding (LPC), Mel-generalized-cepstrum (MGC),... can be used for that purpose.
- 2. In the pre-processing work step, an appropriate choice of window size and frame shift of each phone has to be made to divide the recordings into frames of the chosen feature. The DTW based on the fundamental frequency contour would be an adequate method for voiced sounds but fails to cluster unvoiced phones without a fundamental frequency. Either there could be used two different clustering methods for voiced and unvoiced sounds or a different parameter is being chosen. As MFCC are used in the training process of the HTS

toolkit, these features are a good starting point for clustering using DTW. It is advised to do the MGC analysis with a high order, but then only take the first coefficients for the clustering process to reduce the dimension, while retaining the most important coefficients. This is done in order to reduce the complexity of the calculation.

- 3. The next step would be to create a similarity matrix where each phone is compared with all other phones using DTW as distance measure. This matrix can then be used for clustering.

4.6. Bayesian Information Criterion (BIC)

The Bayesian information criteria is a value, that helps to validate the cluster result. The lower the BIC value is, the better the model fits. It can be calculated as follows:

$$BIC = -2 * \ln L + k * \ln n \tag{4.27}$$

whereas L is the maximised value of the likelihood function,
k is the number of parameters that should be calculated
and n is the number of observations.
Compared to the Akaike information criterion (see section 4.7), the BIC penalises the number of parameters more. If the BIC is plotted over the number of clusters, there optimally should be a peak for the BIC or at least a point at which the improvement of the BIC is only little and a plateau is reached. (see figure 4.10)

In the experiment the software R (R Core Team, 2014) will be used to compute the classification of the data set. There are a lot of different packages and ways to carry out the needed tasks. The chosen package is called mclust (Scrucca et al., 2016) and automatically detects the best number of clusters based on finite normal mixture modeling, offering useful representations and visualisations of fitted models. Contrary to the literature, the best model is the one with the highest BIC (Fraley and Raftery, 2007, p.5), because they use a slightly different definition of the BIC. Different model attributes for volume, shape and orientation are used during the computation and figure

4.10 illustrates the selection of the best model on an example of a diabetes data set (Fraley and Raftery, 2007, p.4f).

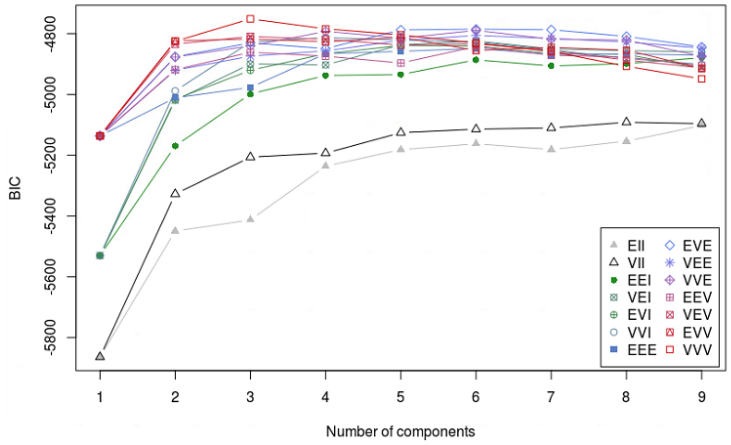


Figure 4.10.: Estimated BIC for a data set with different model parameters (Figure from (Fraley and Raftery, 2007))

Table 4.1.: Result of BIC estimation for the example given above

log-likelihood	n	df	BIC	ICL
-2303.496	145	29	-4751.316	-4770.169

- whereas the first column represents the log-likelihood of the optimal BIC,
- n indicates the number of observations in the data,
- df the number of estimated parameters, and
- ICL the Integrated complete-data likelihood.

The best model estimated by mclust for the given example is the VVV (ellipsoidal, varying volume, shape and orientation) with 3 components as it has the highest BIC (see table 4.1 and figure 4.10).

4.7. Akaike Information Criterion (AIC)

In simple terms, the AIC measures how much data is lost in a statistical model, that describes a process. It is related to the Bayesian information criterion which is described in section 4.6, but has a slightly different calculation formula:

$$AIC = 2 * k - 2 * \ln L \tag{4.28}$$

with L as the maximised value of the likelihood function,
k as the number of parameters
and n as the number of observations.

5. Synthesis and Experiments

The starting point was a segmented set of recordings of one specimen with additional information such as context and song type. The recordings contained 50 songs with a total length of 26 minutes and 49 seconds by a budgerigar (Melopsittacus undulatus) specimen with the nickname “Puck”. Budgies (short form of budgerigars) belong to the species of parrots and have the ability to produce and mimic all kinds of sounds, including human speech. The following figure 5.1 shows an example of an annotated recording from the data set, aligned to the wave form and spectrogram of its underlying audio. The segmentation was made in Praat by an automated script created by PhD candidate Daniel M. Mann from Queens College, City University of New York, who is currently working at the university of Vienna in the Department of Cognitive Biology.

5.0.1. Data Set

The 5 different tier levels include information about the following:

- if the song is directed or undirected and who it is directed to
- if typical head or courtship movement is going on
- phrase type (or song type)
- periodicity
- segmentation of contact calls

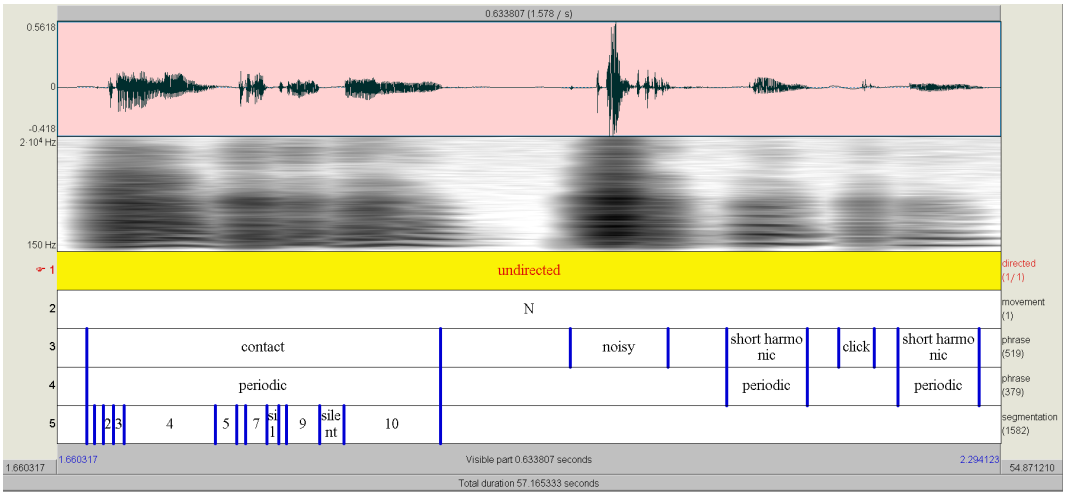


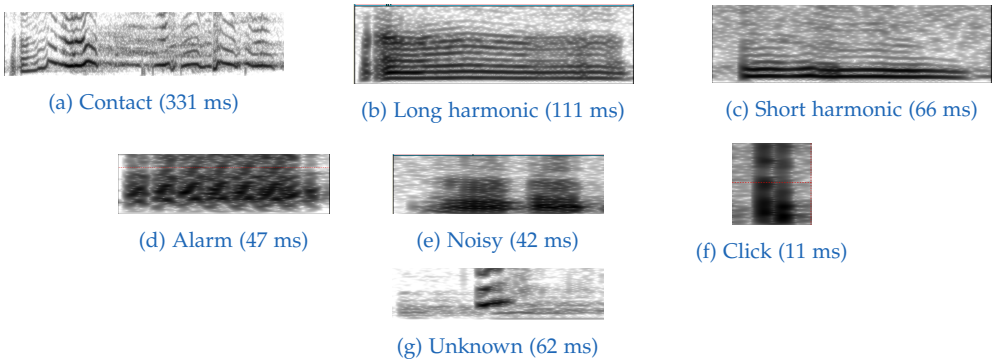
Figure 5.1.: Example of segmented budgie recording

To get high quality recordings without much background noise and interjections from the other birds the budgies were trained to get used to having a microphone held directly to their head. That way a high SNR could be achieved. Still there are parts where background chirps or instrumental noise is present in some amounts. The files are recorded as 48 kHz Wavefiles with 16 bits per sample.
The segmentation of the different budgie sounds is categorised into 7 different groups:

- contact
- long harmonic
- short harmonic
- alarm
- noisy
- click
- unknown

Contact calls are segmented into smaller units called syllables. Syllables are separated by pauses (labelled “silent” in figure 5.1), whereas each syllable consists of smaller elements further called element segments. The element segments represent what could be called phone segments in human language. Yet the number of existing phones of budgerigar sounds is not

known, which is why the non-silent parts within a contact call are only labelled as incremented numbers not indicating any repetitive occurrences. The space between one of the seven groups will be called “silence”. Recapitulating “silence” separates two categories and “pause” separates syllables within a contact call. In figure 5.2 examples of each groups are illustrated.



(a) Contact (331 ms)
(b) Long harmonic (111 ms)
(c) Short harmonic (66 ms)
(d) Alarm (47 ms)
(e) Noisy (42 ms)
(f) Click (11 ms)
(g) Unknown (62 ms)

Figure 5.2.: The 7 main tone quality categories of the budgerigar. All spectrograms have a logarithmic scaled frequency axis (0-15000Hz) on the ordinate, while time windows are specified in the corresponding subcaptions.

The segmentation is done automatically through a Praat script developed by Daniel M. Mann and will be described in the following.

- 1.The audio used for segmentation is first of all band-passed by applying a Hann filter to reduce the spectrum to frequencies between 100 Hz and 15 kHz. The same filtering will also be used on the training files for the HTS toolkit.
- 2. The minimal fundamental frequency of the budgies is set to 400 Hz and the maximum to 10 kHz. The filtering should therefore not affect the fundamental frequency.
- 3. Firstly the rough phrase types are detected and labelled according to parameters like frequency shifts, duration of the calls and amount of voiced or unvoiced frames.
- 4. After the initial phrase division, contact phrases are further segmented. Using thresholds regarding changes of amplitude, pitch, Wiener entropy, voiced/unvoiced regions and noise segment boundaries are

added and labelled with consecutive numbers. In areas with silence within a contact call a "silent" label is introduced. The shortest elements have a duration of only one millisecond. This turned out to be a problem in the training process with HTS, because there need to be enough parameter observations for each state. To model states with very short durations, window size and frame shift would need to be adjusted to values shorter than one millisecond. Further investigation of the short duration segments indicated, that they do not contain valuable information and led to an adaptation of the Praat scripts, with an increased minimal duration of 5 ms.

5.0.2. Evaluating F0/LF0 in Practice

To compare the detection rate of different pitch trackers one respective recording of a budgie was analysed and the results compared. In figure 5.3 the detected fundamental frequency can be seen, as it is detected by different PDA. In the lower part of figure 5.3 a method is used, in which the audio is slowed down to one third of its original speed, by treating the 48 kHz signal as a 16 kHz signal for the analysis. This sometimes helps to get better results, as pitch detectors have problems with very quick varying parameters and areas that contain only short tonal parts. The RAPT (sometimes referred to as "GET_F0") algorithm struggles to find a fundamental frequency and therefore treats a lot of sounds unvoiced as it can be seen in figure 5.3. Trying the trick to slow down the recording does not increase the result significantly. The result from Praat seems to give quite accurate results as it can be seen in figure 5.3. The fundamental frequency matches with the auditory perception and the distinction between voiced, unvoiced and silent areas are meaningfully detected. It was therefore decided to use Praat as the PDA for the analysis part. A significant aspect of fundamental frequency estimation is to decide at which point a sound is still voiced or already unvoiced, if it contains a lot of self-produced noise. Using Praat, the parameter called "voiced / unvoiced" enables to set a threshold for that decision.
To use Praat as the PDA in conjunction with HTS, the F0 estimation needs to be converted into an appropriate format, that can be recognised by the

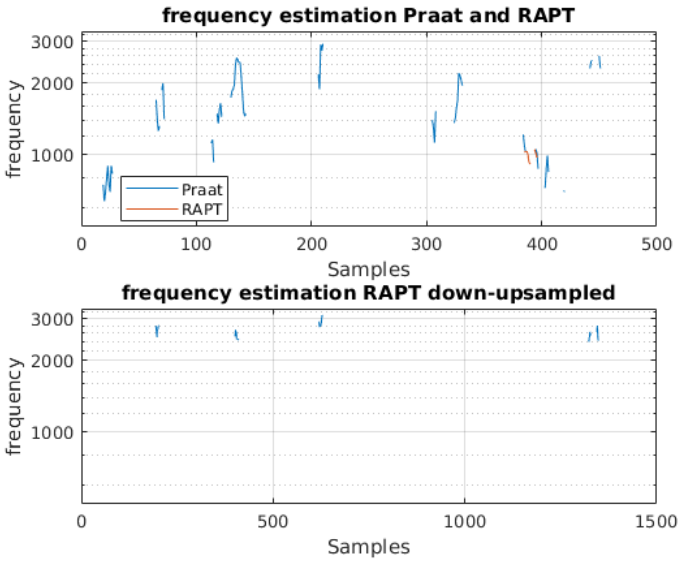


Figure 5.3.: F0 estimation with different methods and PDAs

software. HTS uses the logarithmic frequency (LF0) and stores the information in a binary format. To convert the frequency vector from Praat into a logarithmic scale, the unvoiced regions have to be considered separately, as their value is zero. HTS defines, that unvoiced regions need to have a value of the number -10000000000 ($=-10 * 10^9$). After using the logarithm on voiced regions and setting all unvoiced regions to -10000000000 the file has to be converted to binary format.
A problem with the format of Praat's frequency estimation is, that there are always a few milliseconds at the beginning and at the end of an audio file, where no F0 is calculated (Boersma and Weenink, 2014). Without further notice, that would mess up the alignment of the F0 estimation in relation to the sound signal and speech parameters like MGC. To use the data, without messing up the correct time/frequency interconnection a way had to be found to preserve the correct time alignment. This could be achieved by writing a Praat script that uses the frame number and the corresponding time of that frame for the frequency vector output. The start and end time was then compared to the actual length of the recording and zeros were added for the parts that were not analysed by Praat.

```
selectObject: ``Sound budgie\_songs\_0''
To Pitch (ac): 0.001, 400, 15, ``yes'', 0.03, 0.45, 0.04, 0.15, 0.04, 10000
selectObject: ``Pitch budgie\_songs\_0''
numberOfFrames = Get number of frames
for iframe to numberOfFrames
        time = Get time from frame: iframe
        pitch = Get value in frame: iframe, ``Hertz''
        if pitch = undefined
                appendFileLine: ``budgie\_songs\_0.txt'', fixed\$ (time, 6), ``,'', 0
         else
                appendFileLine: ``budgie\_songs\_0.txt'', fixed\$ (time, 6), ``,'', fixed\$ (pitch, 3)
        endif
endfor
```

5.0.3. Mel-Generalized Cepstral Representation

Trying different parameter extraction methods like linear prediction analysis[1] and Mel-cepstral analysis [2] the latter turned out to give best results, based on a subjective evaluation of the author This also correlates with literature about the comparison of the usage of MFCC and other parametrisation possibilities in conjunction with automated bird song recognition (Kogan and Margoliash, 1998), where also MFCC achieved the best results in relation with HMM and the HTK toolkit. It must be pointed out, that MGC uses the Mel scale - a scale where the human auditory perception is taken into account (Volkmann, Stevens and Newman, 1937). Still the focus of the frequency range that should have the best resolution can be adjusted by the frequency warping factor. The warping factor is dependent of the sampling frequency and puts more emphasis on the analysis of a specific frequency range.

5.0.4. Vibrato and Tremolo

It is a common problem, that HTS smooths vibrato and tremolo effects within syllables, because of the state based modeling. A solution for that is to introduce a 4-dimensional continuous stream that contains information about the vibrato depth and rate, as well as about tremolo depth and its

[1] LPC with 10^{th} order and setting $\gamma = -1$ (Which is done by setting $\gamma = 1$ in HTS) and the frequency warping factor to zero

[2] Mel-cepstral analysis with 34^{th} order

resonance frequency as it is done in a paper, where chaffinch songs are analysed (Bonada, Lachlan and Blaauw, 2016). It should be noted, that the songs of chaffinches can be modeled by single sinusoids with energy and frequency features combined with vibrato and tremolo characteristics, that are very present in their songs. Therefore, it is reasonable, that the spectrograms of chaffinches and budgerigars differentiate a lot. If listened to the investigated budgerigar sound examples with human ears, vibrato or tremolo effects do not seem present, which is also, why synthesis is possible without the incorporation of those features. As a conclusion the implementation of further features might increase the perceived naturalness of the synthesised sounds for birds.

5.1. Resynthesis

In the resynthesis process offered within the HTS toolkit, it is possible to synthesise a waveform, based on the derived LF0 values and MGC analysis. This can be seen as the best synthesised result that can be derived since no statistical modeling is involved. In figures 5.4 and 5.5 the slight differences between original and resynthesised versions are illustrated.

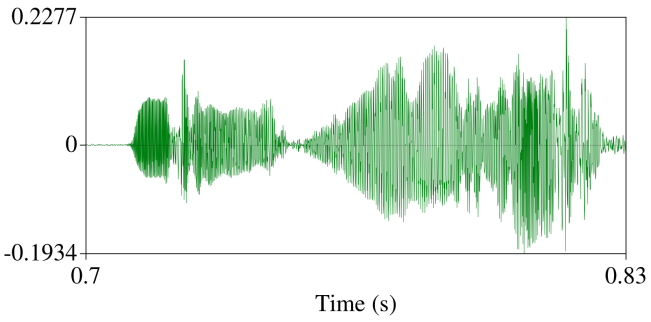


Figure 5.4.: Waveform of an original recording

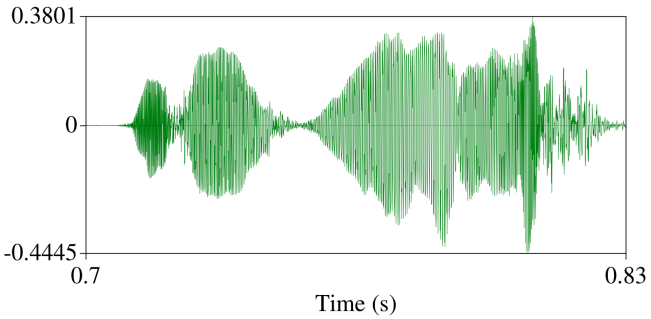


Figure 5.5.: Waveform of a resynthesised recording

5.2. Cluster Solution and Discussion

5.2.1. Voiced sounds

To get an initial clustering of the different phonemes the software R (R Core Team, 2014) was used in conjunction with mclust (Scrucca et al., 2016). In a first step a major subdivision between voiced and unvoiced sounds is made by observing the centre frame of each segment. That means there are two separate processes. The method that was used and will be described now is following a method proposed in [Mak and Barnard, 1996] with some adjustments being made. The data vectors used for the voiced segments consist of the first 12 coefficients of the 34th-order MFCC as well as the energy, all measured on the centre-frame of each segment. Apart from the MFCC features, the logarithmic F0 and Wiener entropy is added to the data vector, so that each vector ends up with a dimension of [1 x 15]. Direct use of these parameters would result in a domination of high values. To solve that problem, the data firstly needs to be scaled, so that all different parameters have the same weight and can be compared. This can be done by using the following formula.

$$x' = \frac{x - \mu}{\sigma} \tag{5.1}$$

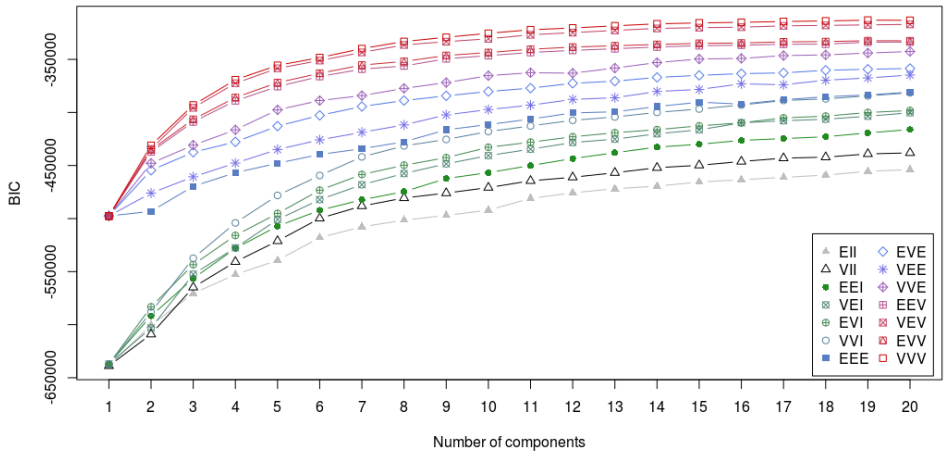


Figure 5.6.: BIC value with different model types for voiced sounds with 20 possible models

In an additional attempt to get better results, the F0 is weighted to have a heavier judicial effect. The method of using the weighting function within mclust software fails, as it only weights certain data vectors heavier than others, which is not the desired effect. Therefore, all columns, except the column containing the LF0 values, are multiplied with a factor of 0.5 to reduce their effect on the clustering. The calculation result can be seen in table 5.1 and the plot of BIC values for different number of components in figure 5.6.

Table 5.1.: BIC value of the voiced-segments cluster result

log.likelihood	n	df	BIC	ICL
-83840	17533	1495	-182289	-185471

The improvement of the BIC for models with more components is rather small. For an optimum solution a clear maximum within the BIC estimation curve would have been expected, which is not the case for this data set. To choose the number of components, the idea was to avoid having too many components containing rather similar segments, but also not to have very distinct ones within the same class. The chosen model is one with 11 components and an ellipsoidal shape, with varying volume, shape, and

orientation (VVV). The following table 5.2 shows the amount of segments contained in each component class:

Table 5.2.: Component size of the voiced segments and their labelling

1=v1	2=v2	3=v3	4=v4	5=v5	6=v6	7=v7	8=v8	9=v9	10=v10	11=v11
1924	1245	1563	1942	1675	2148	2076	1574	1670	725	991

In figure A.1,A.2 and A.3 the first in the training set occurring voiced elements of each component class are visualised by their corresponding waveform, spectrogram (range 0-15 kHz) and segment boundaries.

5.2.2. Unvoiced Sounds

The unvoiced observation vectors have no F0 information and have a dimension of [1 x 14] therefore. The result of the BIC estimation over different component sizes can be seen in figure 5.7 for the case of unvoiced sound segments whereas the chosen model (see table 5.3) is one with 9 components and an ellipsoidal shape, with varying volume, shape, and orientation (VVV)

Table 5.3.: BIC value of the unvoiced-segments cluster result

log.likelihood	n	df	BIC	ICL
-201743.8	14005	944	-412500.2	-417974

The following table 5.4 shows the amount of segments, that each unvoiced component class contains:

Table 5.4.: Component size of the unvoiced segments and their labelling

1=uA	2=uB	3=uC	4=uD	5=uE	6=uF	7=uG	8=uH	9=uI
1405	2095	2293	1566	1369	1222	1911	1709	435

In figure A.4 the first occurring unvoiced elements of each component class are visualised by their corresponding waveform, spectrogram (range 0 Hz

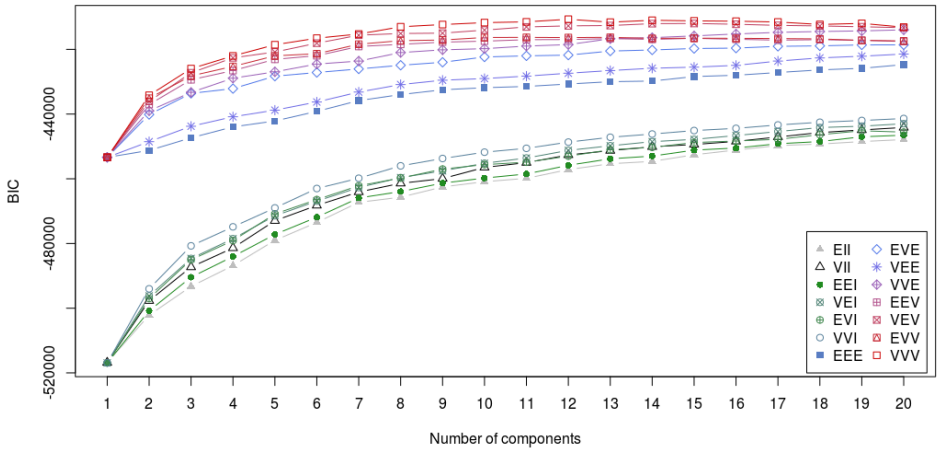


Figure 5.7.: BIC value with different model types for unvoiced sounds with 20 possible models

- 15 kHz) and segment boundaries. The cluster method does not take any duration information into consideration. Figure A.5 shows the distribution of the different voiced sounds of all training and synthesis data, to get an idea of their differences. The additional dependency of the frequency can be seen in a 3D-histogram in figure A.6. The high order of feature parameters makes it difficult to detect the optimum size of components for the cluster analysis. Instead of comparing only the centre frame, segments could be divided into more sub-segments and their features averaged. This method might not be meaningful for very short segments but might improve the outcome of the clustering of longer voiced sounds. Before that, a manual correction of the automated segmentation is inevitable.
Because the segmentation and clustering are not accurate for all segments, some files need to be deleted from the corpus to avoid termination from errors. After removing the error causing files, the training corpus consists of 21 minutes and 2 seconds (out of 27 minutes 13 seconds) within 62 Wave files (out of 75 files).

5.3. Label Files and Decision Trees

To generate a context clustering for birds, a set of questions has to be set up that matches with our available information. The recordings available for that thesis were submitted with additional information about behavioural context, like head movement of the bird or to whom the song was directed to. As it is not known which questions will be useful in the final classification, the aim was to provide as much context information as possible, which resulted in the list of questions in table 5.5.

Table 5.5.: format of label files

p1	the previous element identity
p2	the current element identity
p3	the next element identity
p4	position of the current element in the current syllable (forward)
p5	position of the current element in the current syllable (backward)
p6	whether the previous element is voiced or not (0: not voiced, 1: voiced)
p7	whether the current element is voiced or not (0: not voiced, 1: voiced)
p8	whether the next element is voiced or not (0: not voiced, 1: voiced)
a1	the number of elements in the previous syllable
b1	the number of elements in the current syllable
b2	position of the current syllable in the current phrase (forward)
b3	position of the current syllable in the current phrase (backward)
c1	the number of elements in the next syllable
d1	the number of syllables in the previous phrase
d2	the number of syllables in the current phrase
d3	the number of syllables in the next phrase
d4	Number of elements in the previous phrase
d5	Number of elements in the current phrase
d6	Number of elements in the following phrase
e1	Directed (0: undirected, 1: male directed, 2: malemix directed, 3: inanimate directed
e2	Headmovement in current song(0: no, 1: yes, 2: unknown)
d1	(unused!) Frequency area (0: $= 0$, 1: < 1000, 2: < 1500, 3: < 2000, 4: < 2500, 5: < 3000, 6: < 4000, 7: ≥ 4000)

The format of the created label files is shown in table 5.6:

Table 5.6.: Format of the label files

p1−p2+p3^p4 =p5@p6_p7&p8/*A* :a1/*B* :b1−b2−b3/*C* :c1 /*D* :d1_d2+d3!d4#d5|d6/*E* :e1$e2/*D* :d1

The last feature "d1" finally is not used in the question files so that no information about fundamental frequency needs to be specified for the creation of a full-context label file. This means, that the toolkit generates the F0 solely from the trained models.

After training, there is the possibility to visualise the decision tree-files for features like fundamental frequency, MGC and duration. The following figure 5.8 shows a part of the third feature stream of MGC parameters. The full decision tree can be seen in figure A.7 and should demonstrate the high complexity of the derived model.

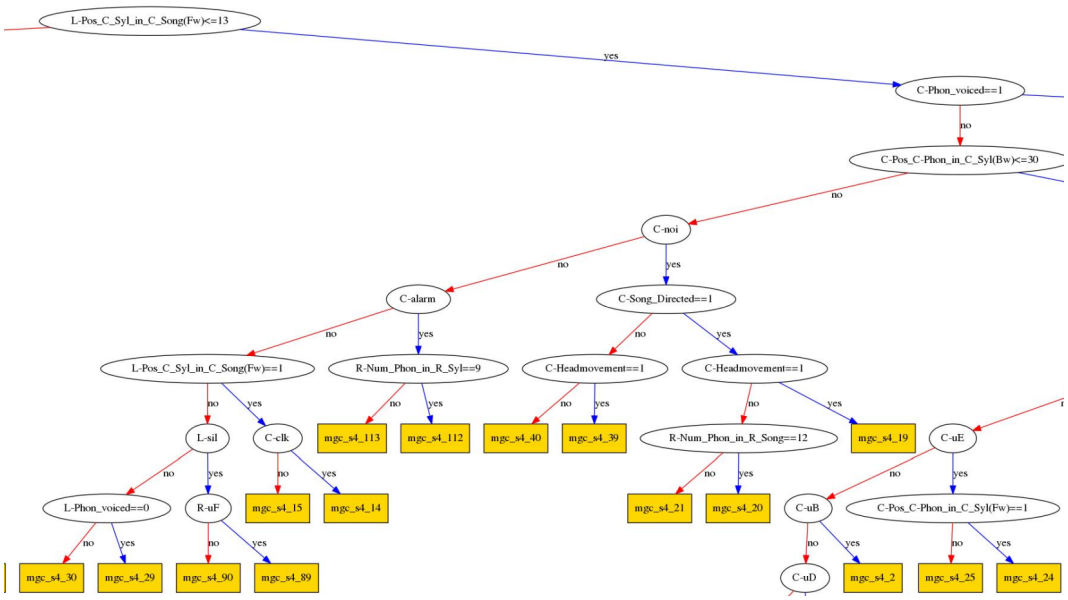


Figure 5.8.: Cut-out at the top of feature stream 3 to determine MGC parameters

The first question of the decision tree in figure 5.8 regards the forward position number of the left element. This is the most decisive question and an interpretation could be, that an element sounds different in long songs that have more than 13 syllables than it sounds in songs with 13 or less syllables. The next decision in the illustrated tree is whether the element is voiced or not. Apparently voiced and unvoiced segments have different sound features, which is why we find that question in the decision tree that soon. A very interesting aspect is the question about head-movement, that has a direct influence on the "noise" leaf node in the example. This question seems to have great significance, as it appears more often in other parts of the tree.

In figure 5.9 the decision tree concerning LF0 is illustrated. If an element is voiced or unvoiced makes the biggest difference concerning the LF0 stream. Elements that were labelled as unvoiced, can still have areas where a fundamental frequency exist. Another F0 decisive question is the "directed" information. It is possible, that budgerigars use different frequencies if they communicate with a bird or an inanimate object. Head-movement also seems to have an impact on the fundamental frequency in "male directed" songs.

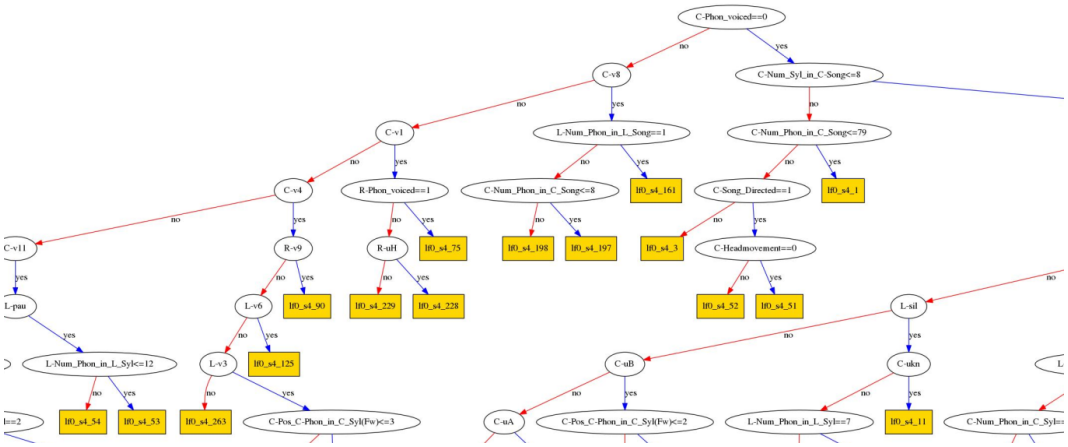


Figure 5.9.: Cut-out at the top of feature stream 3 to determine LF0 parameters

The decision tree for duration modeling can be seen in figure 5.10. The first question in this tree is about the backward number of elements in

the current syllable. The duration of an element in a syllable with many elements is therefore different to the duration of an element in a short syllable. The second most important question is a discrimination between voiced and unvoiced sounds which is comprehensibly, as unvoiced sounds tend to be much shorter than voiced sounds. Voiced and unvoiced elements seem to have a critical influence on the duration of neighbouring elements, as a question for the unvoiced element "uE" concerns the element type (voiced or unvoiced) to its right.

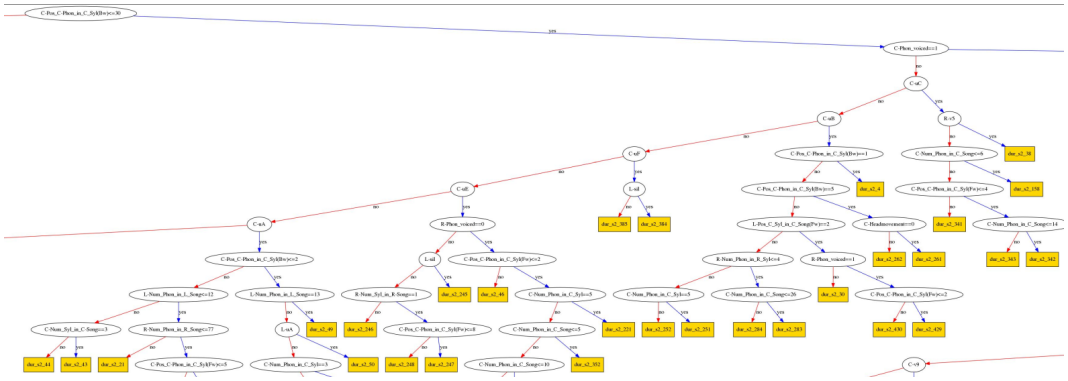


Figure 5.10.: Cut-out at the top of the decision tree for duration modeling

The amount of final leave nodes contained in duration, LF0 and MGC feature-streams are summed up in table 5.7. Duration is modeled through one feature stream, whereas LF0 and MGC features have one feature stream for each of the five states.

Table 5.7.: Summation of all tree leaf nodes per feature

Duration	LF0	MGC
435	1946	823

5.4. Synthesis

To synthesise a budgerigar song from a user input, the following steps need to be done:

- Create sequences of phrases, whereas each phrase should be separated with a silence label.
- Contact calls need to be further divided into element segments.
- Create a text-file that includes not only the current element, but also the previous and the following one, according to the syntax presented in table 5.6. A non-computable value (for example the previous phoneme identity of the first element) will be labelled "xx".
- Add the context information that is available together with the numerical computations of position.
- Let the duration be calculated either by itself (see figure 5.11) or introduce time information in front of each element as demonstrated in figure 5.12, whereas the second method is not full synthesis.

```
xx-sil+v3^0=0@xx_0&1/A:xx/B:0-0-0/C:9/D:0_0+0!xx#0|9/E:0$1/D:0
sil-v3+v8^1=9@0_1&1/A:xx/B:9-1-1/C:0/D:0_1+1!xx#9|7/E:0$1/D:3
v3-v8+v7^2=8@1 1&1/A:xx/B:9-1-1/C:0/D:0 1+1!xx#9|7/E:0$1/D:6
```

Figure 5.11.: Short example of a label file without alignment information

```
0 50000 xx-sil+v3^0=0@xx_0&1/A:xx/B:0-0-0/C:9/D:0_0+0!xx#0|9/E:0$1/D:0
50000 117629 sil-v3+v8^1=9@0_1&1/A:xx/B:9-1-1/C:0/D:0_1+1!xx#9|7/E:0$1/D:3
117630 431640 v3-v8+v7^2=8@1 1&1/A:xx/B:9-1-1/C:0/D:0 1+1!xx#9|7/E:0$1/D:6
```

Figure 5.12.: Short example of a label file with alignment information

5.5. Objective Evaluation

5.5.1. Distance Measurement

To compare the synthesised versions with the original recording, MFCC values are compared using dynamic time warping and the score output, which tells the difference between a test data and the reference data. A high

distance score indicates that the two data vectors are very different from each other, whereas a low score of 0 signifies no distance and therefore complete conformity. The process was performed by the "Speech Signal Processing Toolkit" (SPTK, 2015) and the result can be seen in table 5.8. Original and resynthesised versions match best in all cases, as no statistical modeling is involved. The comparison of the full synthesis and the original version shows, that they have the highest distance score. In relation to the synthesis that incorporates the original duration, we see a slight increase of similarity to the original recording by an improvement of 0.1 (song 99) to 0.5 (song 10). Comparison of original and resynthesis of song 10 has the highest similarity of all song and also works best in the full synthesis.

Table 5.8.: Distance between original (Orig.), resynthesised (Resyn.) and synthesised (Syn.) versions

Song number	Orig. - Resyn. Original duration	Orig. - Syn. Original duration	Orig. - Syn. Full synthesis
10	0.784	1.240	1.478
15	0.812	1.203	1.779
17	0.836	1.452	1.682
99	0.814	1.531	1.641

Figure 5.13 and 5.14 show the comparison of spectrograms between the original and two synthesised versions. The aligned version makes use of the time alignment, given in the input label, whereas the unaligned one uses the alignment of its trained duration model. The fundamental frequency is emphasised on those parts where it is detected. In both examples the synthesised versions show strong similarities with the original ones. The harmonics follow the contour of the ones obtained from the original recording to some point, but miss parts of the fine structure, which makes the synthesised versions sound a bit whistle-like and lack some natural noise. Duration modeling works well all together, but as expected the aligned versions are more similar to the original. It can be seen, that harmonic sounds, that are labelled as unvoiced elements are synthesised with less energy and broadband (see the "uG" element in figure 5.14).

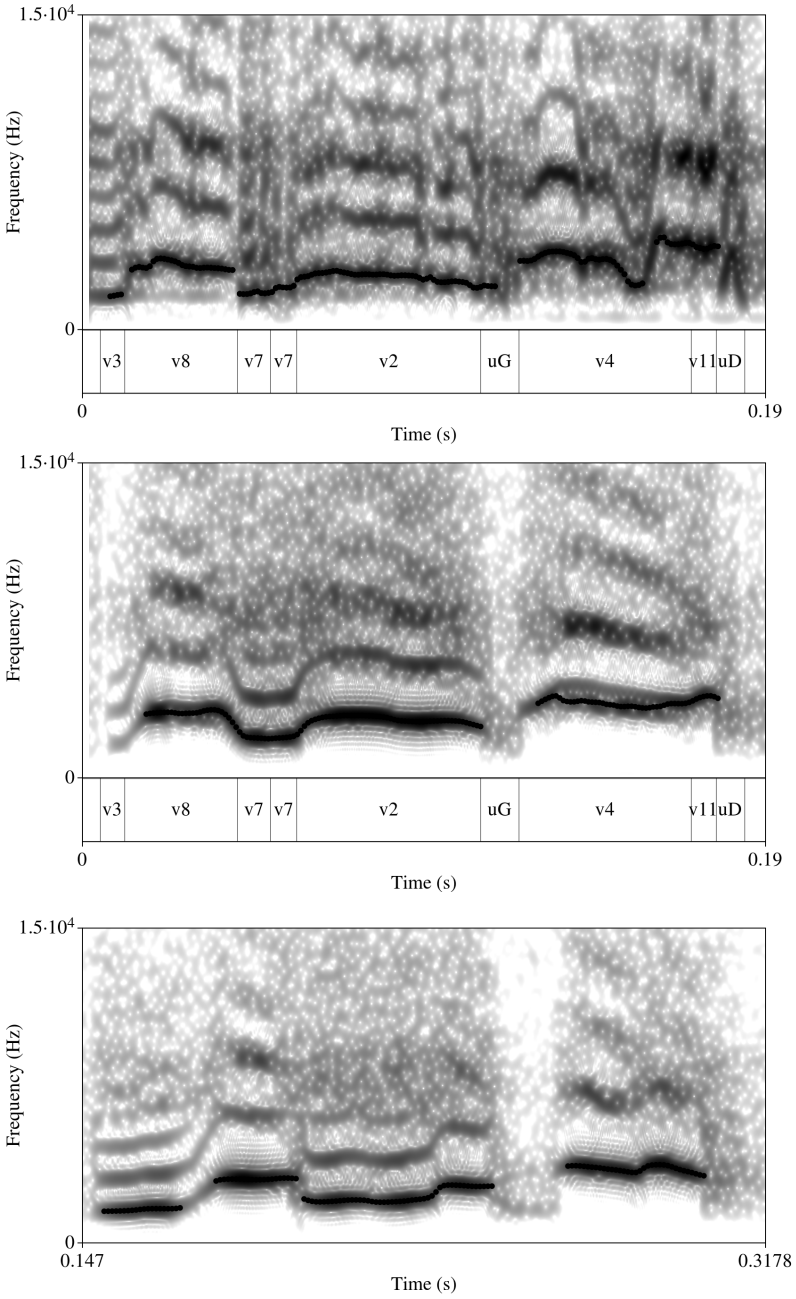


Figure 5.13.: Spectrogram of original (top), aligned synthesis (middle) and full synthesis (bottom), excerpt of song 15

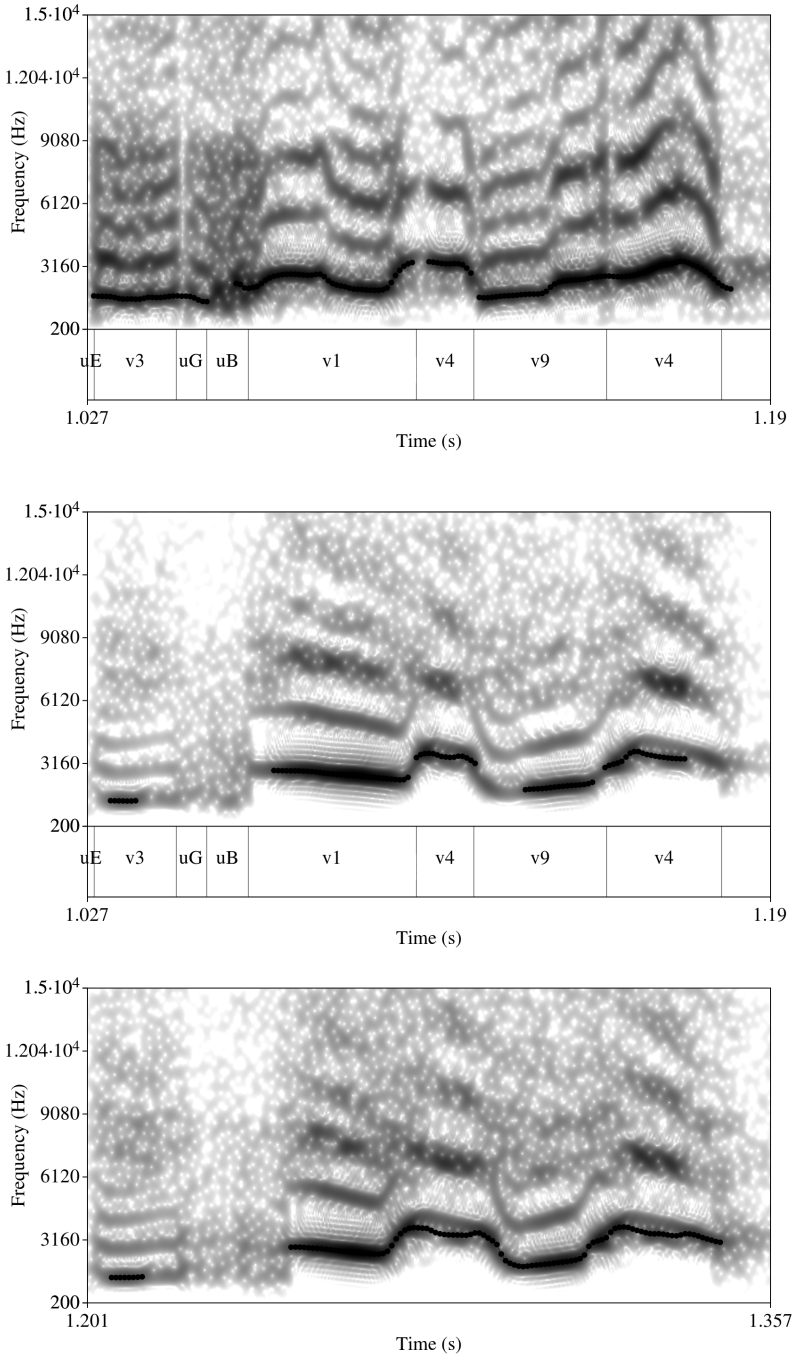


Figure 5.14.: Spectrogram of original (top), aligned synthesis (middle) and full synthesis (bottom), excerpt of song 99

5.5.2. F0

To illustrate the F0 estimation of synthesised samples in relation to F0 of real recordings, the following figures 5.15 and 5.16 show the comparison between the original phrase and synthesised version of it. The aligned synthesis is not a full synthesis, as the durations are defined explicitly. The fundamental frequency of the aligned synthesis (blue line) follows the fundamental frequency of the original (black line) clearly but misses some quick variations for instance at the end of song 15 (see figure 5.16). The full synthesis has different durations for each element and therefore might not show strong similarities on the first look. In song 15 for example the first elements are much longer than the original ones, which introduces a time shift to the F0 shape to the full synthesis (green line).

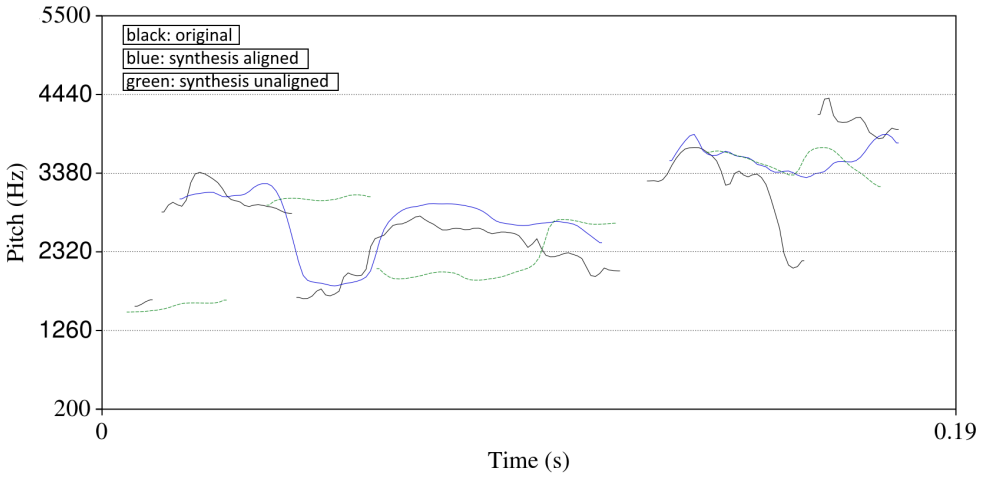


Figure 5.15.: F0 comparison of original recording, aligned synthesis and unaligned synthesis, excerpt of song 15

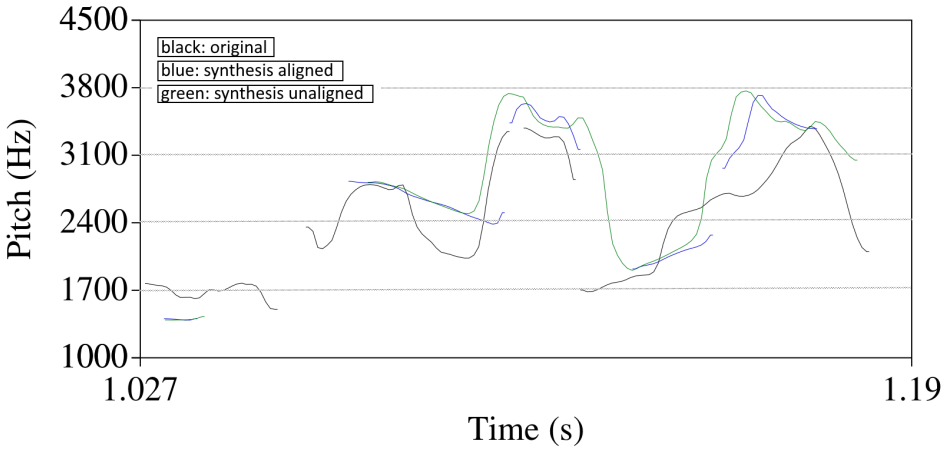


Figure 5.16.: F0 comparison of original recording, aligned synthesis and unaligned synthesis, excerpt of song 99

Figure 5.17 represents the difference between the extracted fundamental frequency of the original recording and the aligned synthesised versions throughout the whole song.

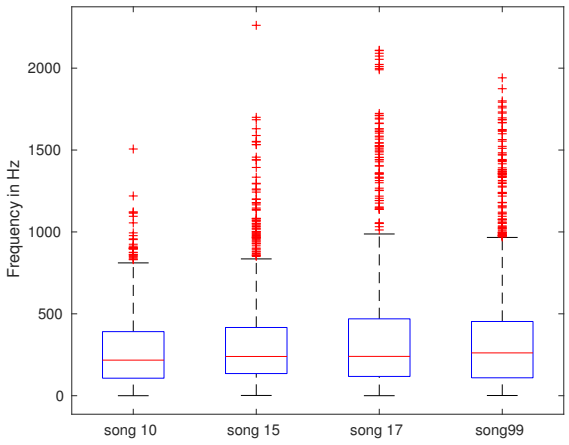


Figure 5.17.: F0 difference between the original and aligned songs

The fundamental frequency contour of the synthesised versions is definitely smoothed, which is not unexpected, as the models use some kind of average pitch contour. Even though the average difference between original and synthesised version is around 300 Hz this is a satisfying result, considering that many frame points differ due to a time lag of the transition between adjacent syllable elements. The statistical outliers of the box-plot, also called whiskers, occur because of quick frequency changes over a wide range, that are smoothed in the synthesis which gives them a bit of time lag.

5.5.3. Own Example

In this section an own song is composed, by putting together the available elements and phrases. The input of the desired elements looks like the following:
{“sil”;“uD”;“v5”;“v6”;“v4”;“uE”;“pau”;“uF”;“v5”;“v7”;“v9’;“uG”;“sil”;“pau”;
“uD”;“uH”;“v4”;“sil”;“uI”;“uI”;“v3”;“v1”;“v3”;“pau”;“v3”;“uD”;“uA”;“uA”;
“sil”;“alarm”;“sil”;“v3”;“uF”;“uB”;“uB”;“uH”;“sil”}
whereas “sil” means “silence”,
“pau” means “pause”,
and “xx” indicates non-computable information.
Additional context information about head movement is set by the user as well as the birds directed intentions. In the example movement is set to “no movement” and that the song should be “undirected”. A Matlab script puts all the information together and ends up with a label file seen in figure 5.18.

```
xx-sil+uD^0=0@xx_0&0/A:xx/B:0-0-0/C:5/D:0_0+0!xx#0|11/E:0$0/D:
sil-uD+v5^1=5@0_0&1/A:xx/B:5-1-2/C:5/D:0_2+1!xx#11|4/E:0$0/D:
uD-v5+v6^2=4@0_1&1/A:xx/B:5-1-2/C:5/D:0_2+1!xx#11|4/E:0$0/D:
v5-v6+v4^3=3@1_1&1/A:xx/B:5-1-2/C:5/D:0_2+1!xx#11|4/E:0$0/D:
v6-v4+uE^4=2@1_1&0/A:xx/B:5-1-2/C:5/D:0_2+1!xx#11|4/E:0$0/D:
v4-uE+pau^5=1@1_0&0/A:xx/B:5-1-2/C:5/D:0_2+1!xx#11|4/E:0$0/D:
uE-pau+uF^0=0@0_0&0/A:5/B:5-2-1/C:5/D:0_2+1!xx#11|4/E:0$0/D:
pau-uF+v5^1=5@0_0&1/A:5/B:5-2-1/C:0/D:0_2+1!xx#11|4/E:0$0/D:
uF-v5+v7^2=4@0_1&1/A:5/B:5-2-1/C:0/D:0_2+1!xx#11|4/E:0$0/D:
v5-v7+v9^3=3@1_1&1/A:5/B:5-2-1/C:0/D:0_2+1!xx#11|4/E:0$0/D:
v7-v9+uG^4=2@1_1&0/A:5/B:5-2-1/C:0/D:0_2+1!xx#11|4/E:0$0/D:
```

Figure 5.18.: Excerpt of the script generated label file of an own composition

The spectrogram can be seen in figure 5.19. The silence and pause parts can be seen very clearly, as well as some voiced elements. The noisy signal between $1.0 - 1.2$ seconds is also well distinguished from the other elements. It sounds very budgerigar-like to the reader and motivates to try out more sequences.

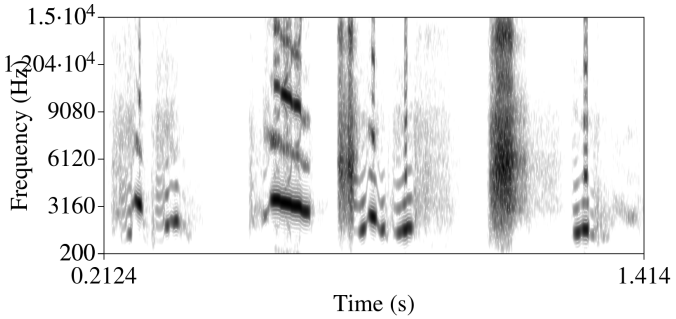


Figure 5.19.: Example of a self created budgerigar song

5.6. Problems

As the synthesis of budgerigars is a very new approach, there arise quite a few problems that now can be addressed more clearly. One of the first tasks that were done is the segmentation of the audio data. The initial segmentation that was used in this thesis is produced completely automated, which gives the possibility to find thousands of segment boundaries, but also contains errors, that effect the clustering, training and synthesis part. A manual correction of all segments would take too much time for the tasks of this thesis, but would be inevitable for a further improvement, directly solving a few problems in this section.

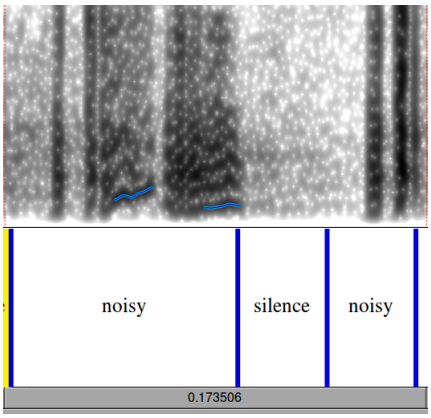


Figure 5.20.: Two segments labelled "noisy"

Phrases with more context information achieve much better synthesis results than short sounds. This might be due to the great variance of the different elements and phrases. Therefore, "contact" calls give the best results, whereas especially "clicks" and "noisy" produce clearly audible errors and artefacts. Reducing the number of states for short calls might reduce the problem (Kogan and Margoliash, 1998, p.12), but the solution probably lies within an earlier stage. The training set we used contains quite a few different sounds labelled as "click" and "noisy" that have great variance in terms of duration and spectral parameters without enough additional context information. To illustrate the disparity of their spectral parameters, two segments from the training set - both labelled "noisy" - can be seen in figure 5.20. This is not completely unexpected, as the prior focus was to model contact calls. What makes the synthesis even worse, is that the "clicks" and "noisy" sounds are inserted in a too rhythmic way. The sounds that produce the problems are all surrounded with silence segments and could therefore be replaced with direct cut-outs from original or resynthesised audio material, without having to consider transitions between sounds. One approach to resolve the creation of bad models could be the reduction of the training set by excluding most of the problematic sounds, if it can be assumed, that those calls do not contain important information. An even better solution would be to find classes to label groups of them differently. As those calls are not that frequent, this could be done manually just by looking at their spectrograms and find similar ones.
Some parts of synthesised audio data are over-amplified, most likely because of the processes done by global variance and post filters that have problems to deal with the big spread within models. Because turning off global variance and post-filtering resulted in even worse quality, the effect could only be reduced by a weight factor applied on global variance (factor was set to 0.8). Internet research revealed that over-amplifying is a common

issue in the relation with the HTS toolkit. Advices to reduce the volume of the training data and to adapt the program code were followed, which reduces the warnings, but do not fully dissolve the problem.
Because frame shift is very short, the data vectors of long sounds can soon get too big for some computation processes during the training. Therefore, long recordings were split into separate parts (< 50 seconds), which means that this resulted in changes of some contextual information because some contextual factors depend on the whole song. It is questionable if cutting the recordings has an effect on the synthesis. Knowledge about associated syllables would be needed, to clarify the legitimacy to adjust the file length to the computing capacity of the software.
In direct comparison of original and synthesised material a clear smoothing of the frequency contour for the fundamental as well as for harmonics can be seen. This was expected because of the fast variation of pitch and intensity. Global variance reduces the effect a bit, but some details still disappear. A method to retain those features will be explained in the section of future work in chapter 7.

6. Conclusion

A synthesis toolkit based on Hidden Markov Models was developed, that produces budgerigar vocalisations from user input. The toolkit gives the possibility to conduct further experiments with budgerigars to find out more about their preferences and may even help to find out more about simple syntax contexts and pattern sequence. The quality of the synthesis varies across the different budgerigar calls, whereas the main focus was placed on contact calls, which were thought to be the most problematic phrases. For these sounds satisfactory results can be obtained, though typical "click" and "noisy" sounds have too much variation in the training set and too less context information. The problems that occur have been described in the corresponding sections and seem to be solvable to some point. To my knowledge, there is no work with budgerigar synthesis made so far, due to the problematic segmentation of complex contact calls. With the provided segmentation, an effort was made to put its effectiveness to a hard test. The used methods are widely used in speech synthesis and all the software needed is available as open-source.

7. Future Work

Even though a lot of improvements have already taken place, there is still a lot to be optimised and done to contribute to a more realistic synthesis of budgerigar sounds. First of all, the technique to segment the budgerigar songs is not always accurate, especially if there is some background noise going on. There are a few ways to eliminate those problems. The first step could be to clean up the recordings and cut out sections with too much background noise, which would reduce the data-set drastically. Another possible way might be the production of new recordings, with a special focus on the isolated sounds of the recorded specimen to get a good signal-to-noise ratio and paying attention to the direction between budgerigar and the microphone. It should be noted that this process might sound much easier than it is in practice because budgerigars are living creatures with uncontrollable behaviour. Another task would be the manual removal of sections where false insertions occur after the overall segmentation, or to directly adapt the segmentation script further to make it more accurate and more robust against background noise. All those methods would help to generate more accurate models and produce less artefacts in the synthesised sounds. Strongly related to segmentation is the question of how many distinct phonemes there actually are in the communication of budgerigars, as there has only been made assumptions in that thesis. Leaving the pre-processing work steps behind we go further to the task of optimising the training process. The HTS toolkit offers many parameters that can be trimmed and experimented with to increase the naturalness of the result. The incorporation of vibrato and tremolo features is very successful in retaining spectral details and rapid volume changes (Bonada, Lachlan and Blaauw, 2016) and might increase the vividness of areas where tremolo and vibrato segments appear.
Behavioural experiments with budgerigars could evaluate, whether the representation of the resynthesised samples actually seem natural to the

birds. This could be done by a preference test in a setup already familiar to the birds of the Viennese budgie lab, where the recordings were made. The preference test allows the birds to choose between three different wooden slats that are placed in front of a speaker. Two of the speakers are used for playback, while one always remains silent. The elapsed time that a bird sits on each slat is then measured and evaluated. In addition to the preference test with budgerigars, an experiment with human listeners could evaluate the subjective quality of the synthesised sounds for different sizes of fast Fourier transformation, analysis window and frame shift.

Bibliography

Arriaga, J. et al. (2015). 'Bird-DB: A database for annotated bird song sequences'. In: *Ecological Informatics* 27, pp. 21–25.

Bezzel, E. and R. Prinzinger (1990). *Ornithologie*. Stuttgart, Ulmer Verlag, 1990, UTB große Reihe Nr.8051.

Boersma, P. and D. Weenink (2014). *Praat: doing phonetics by computer*. Accessed: 12.10.2018. URL: http://www.fon.hum.uva.nl/praat/manual/Script_for_listing_F0_statistics.html.

Bonada, J., R. Lachlan and M. Blaauw (2016). 'Bird Song Synthesis Based on Hidden Markov Models'. In: *Interspeech 2016*. ISCA.

Breiman, L. et al. (1984). *Classification and Regression Trees*. Taylor & Francis Ltd. 368 pp.

Brumm, H. and M. Naguib (2009). 'Chapter 1 Environmental Acoustics and the Evolution of Bird Song'. In: *Advances in the Study of Behavior*. Elsevier, pp. 1–33.

Budney, G. F. and R. W. Grotke (1997). 'Techniques for Audio Recording Vocalizations of Tropical Birds'. In: *Ornithological Monographs* 48, pp. 147–163.

Cheveigné, A. de and H. Kawahara (2002). 'YIN, a fundamental frequency estimator for speech and music'. In: *The Journal of the Acoustical Society of America* 111.4, pp. 1917–1930.

Deller, Jr., J. R., J. H. L. Hansen and J. G. Proakis (21st Sept. 1999). *Discrete-Time Processing of Speech Signals*. Wiley-Blackwell. 936 pp.

Dent, M. L. et al. (1997). 'Perception of synthetic /ba/–/wa/ speech continuum by budgerigars (Melopsittacus undulatus)'. In: *The Journal of the Acoustical Society of America* 102.3, pp. 1891–1897.

Fraley, C. and A. Raftery (2007). 'Model-based Methods of Classification: Using the mclust Software in Chemometrics'. In: *Journal of Statistical Software* 18.6.

Furui, S. (2000). *Digital Speech Processing: Synthesis, and Recognition, Second Edition*. Marcel Dekker Inc. 476 pp.

Gill, L. F. et al. (2016). 'A minimum-impact, flexible tool to study vocal communication of small animals with precise individual-level resolution'. In: *Methods in Ecology and Evolution* 7.11. Ed. by R. Freckleton, pp. 1349–1358.

Halkidi, M., Y. Batistakis and M. Vazirgiannis (2001). 'Clustering algorithms and validity measures'. In: *Proceedings Thirteenth International Conference on Scientific and Statistical Database Management. SSDBM 2001*. IEEE Comput. Soc.

Huang, X., A. Acero and H.-W. Hon (2001). *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*. PRENTICE HALL. 1008 pp.

Hummel, G. (2000). *Anatomie und Physiologie der Vögel. Kompendium für Studium und Praxis.* UTB, Stuttgart.

Jacob, J. (2018). Accessed: 20.11.2018. URL: https://commons.wikimedia.org/wiki/File:Avian_respiratory_and_vocal_anatomy.png.

Katahira, K. et al. (2011). 'Complex Sequencing Rules of Birdsong Can be Explained by Simple Hidden Markov Processes'. In: *PLoS ONE* 6. Ed. by G. G. de Polavieja, p. 9.

Klapuri, A. (2003). 'Multiple fundamental frequency estimation based on harmonicity and spectral smoothness'. In: *IEEE Transactions on Speech and Audio Processing* 11.6, pp. 804–816.

Kogan, J. A. and D. Margoliash (1998). 'Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden Markov models: A comparative study'. In: *The Journal of the Acoustical Society of America* 103.4, pp. 2185–2196.

Ludeña-Choez, J., R. Quispe-Soncco and A. Gallardo-Antolín (2017). 'Bird sound spectrogram decomposition through Non-Negative Matrix Factorization for the acoustic classification of bird species'. In: *PLOS ONE* 12.6. Ed. by B. Sokolowski.

Mak, B. and E. Barnard (1996). 'Phone clustering using the Bhattacharyya distance'. In: *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP*. IEEE.

Manley, G., G. Schwabedissen and O. Gleich (1993). 'Morphology of the basilar papilla of the budgerigar,Melopsittacus undulatus'. In: *Journal of Morphology* 218.2, pp. 153–165.

Mann, D. M. (2018). *Recording of a budgerigar at the budgie laboratory vienna*.
Marler, P. (2001). *Origins of music and speech: insights from animals*. A Bradford Book.
Mindlin, G. and R. Laje (2006). *The Physics of Birdsong*. Springer Berlin Heidelberg.
Müller, M. (26th Sept. 2007). *Information Retrieval for Music and Motion*. Springer Berlin Heidelberg. 332 pp.
O'Reilly, C. and N. Harte (2017). 'Pitch tracking of bird vocalizations and an automated process using YIN-bird'. In: *Cogent Biology* 3.1. Ed. by H. Burda.
Odom, K. and L. Benedict (2018). 'A call to document female bird songs: Applications for diverse fields'. In: *The Auk* 135.2, pp. 314–325.
Pieplow, N. (7th Mar. 2017). *Peterson Field Guide to Bird Sounds of Eastern North America*. Houghton Mifflin. 608 pp.
R Core Team (2014). *R: A Language and Environment for Statistical Computing*. Accessed: 02.12.2018. R Foundation for Statistical Computing. Vienna, Austria. URL: https://www.R-project.org.
Rothenberg, D. (18th Sept. 2007). *Warum Vögel singen*. Spektrum-Akademischer Vlg.
Rothenberg, D. et al. (2014). 'Investigation of musicality in birdsong'. In: *Hearing Research* 308, pp. 71–83.
Saratxaga, I. et al. (2012). 'Perceptual Importance of the Phase Related Information in Speech'. In: *13th Annual Conference of the International Speech Communication Association (Interspeech 2012), Portland, USA*, pp. 1448–1451.
Saunders, J. C., W. F. Rintelmann and G. R. Bock (1979). 'Frequency selectivity in bird and man: A comparison among critical ratios, critical bands and psychophysical tuning curves'. In: *Hearing Research* 1.4, pp. 303–323.
Scrucca, L. et al. (2016). 'mclust 5: clustering, classification and density estimation using Gaussian finite mixture models'. In: *The R Journal* 8.1. Accessed: 25.11.2018, pp. 205–233. URL: https://journal.r-project.org/archive/2016-1/scrucca-fop-murphy-etal.pdf.
Sharma, M. and R. Mammone (1996). '"Blind" speech segmentation: automatic segmentation of speech without linguistic knowledge'. In: *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP '96*. IEEE.

SPTK (2015). *Speech signal processing toolkit (sptk)*. Accessed: 16.11.2018. URL: http://sp-tk.sourceforge.net/.

Suthers, R. (2004). 'How birds sing and why it matters'. English. In: *Nature's Music: The Science of Birdsong*. Elsevier Inc., pp. 272–295.

Talkin, D. (1995). *A robust algorithm for pitch tracking (RAPT)*.

Taylor, P. (2009). *Text-to-Speech Synthesis*. Cambridge University Press.

Templeton, C. N. (2005). 'Allometry of Alarm Calls: Black-Capped Chickadees Encode Information About Predator Size'. In: *Science* 308.5730, pp. 1934–1937.

Thompson, N., K. LeDoux and K. Moody (1994). 'A system for describing bird song units'. In: *Bioacoustics* 5.4, pp. 267–279.

Thorpe, W. H. (1958). 'The Leaning of Song Patterns by Birds, with Especial Reference to the Song Chaffinch Fringilla coelebs'. In: *Ibis* 100, pp. 535–570.

Tokuda, K., T. Kobayashi et al. (1994). 'Mel-generalized cepstral analysis - A unified approach to speech spectral estimation'. In:

Tokuda, K., Y. Nankaku et al. (2013). 'Speech Synthesis Based on Hidden Markov Models'. In: *Proceedings of the IEEE* 101.5, pp. 1234–1252.

Tokuda, K., H. Z. Zen and A. Black (2002). 'An HMM-based speech synthesis system applied to English'. In: *Proceedings of 2002 IEEE Workshop on Speech Synthesis 2002 WSS-02*. IEEE.

Turner, A. (2017). Accessed: 29.11.2018. URL: https://angusturner.github.io/generative_models/2017/11/03/pytorch-gaussian-mixture-model.html.

Volkmann, J., S. S. Stevens and E. B. Newman (1937). 'A Scale for the Measurement of the Psychological Magnitude Pitch'. In: *The Journal of the Acoustical Society of America* 8.3, pp. 208–208.

Wickstrom, D. C. (1982). 'Factors to Consider in Recording Avian Sounds'. In: *Acoustic Communication in Birds*. Elsevier, pp. 1–52.

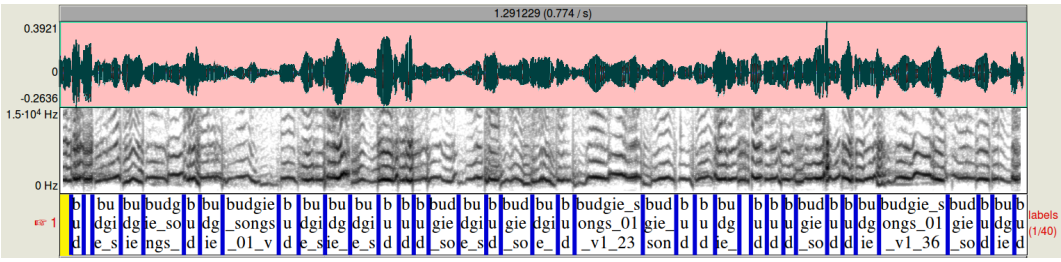
Wildtronics, LLC (2017). Accessed: 23.10.2018. URL: https://www.wildtronics.com/parabolicarticle.html#.W89NQ2gzaUk.

Wunsch, H. (2001). 'Der Baum-Welch Algorithmus für Hidden Markov Models, ein Spezialfall des EM-Algorithmus'. In: *NA*. Accessed: 28.11.2018. URL: http://www.sfs.uni-tuebingen.de/resources/em.pdf.

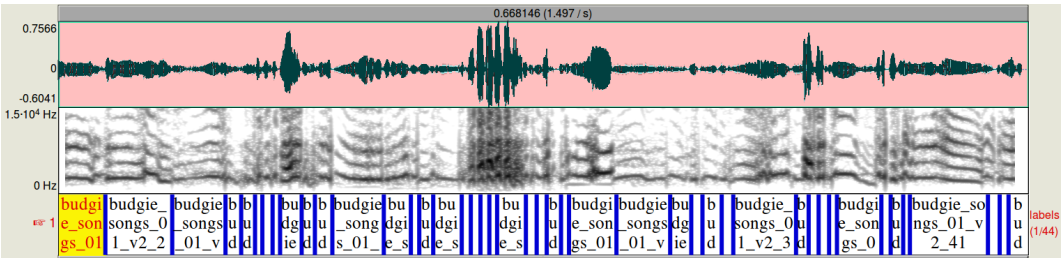
Young, S. et al. (2015). *The HTK Book (version 3.5a)*. Cambridge University Engineering Depart- ment.

Appendix A.

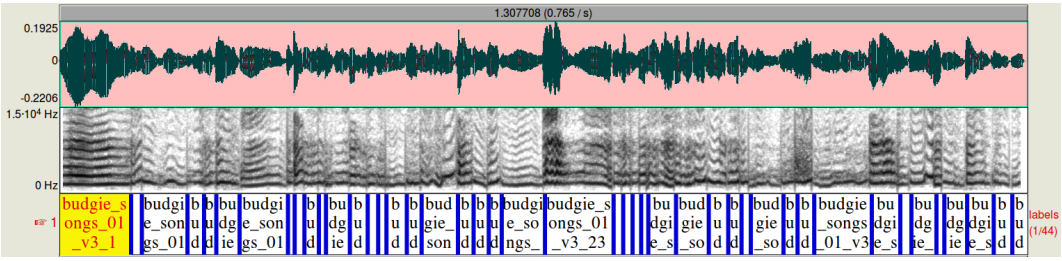
Appendix



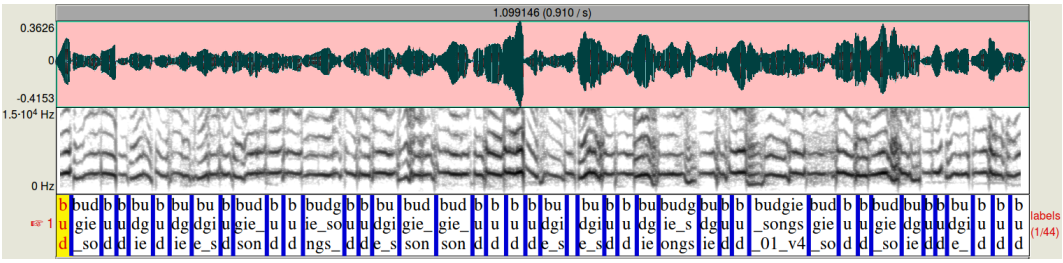
(a) v1



(b) v2

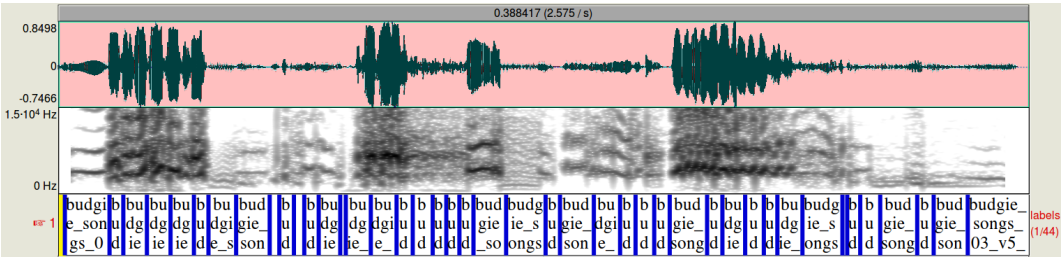


(c) v3

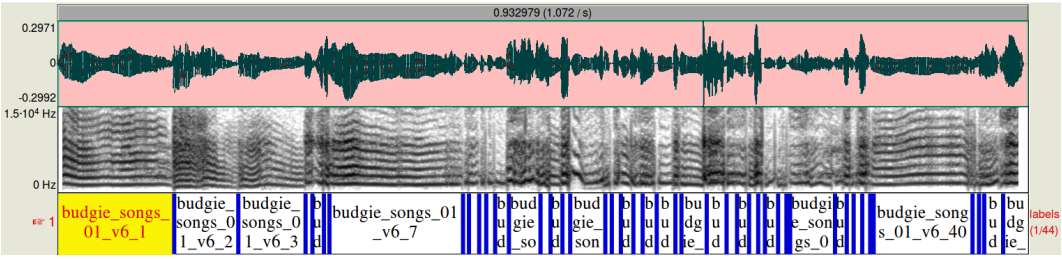


(d) v4

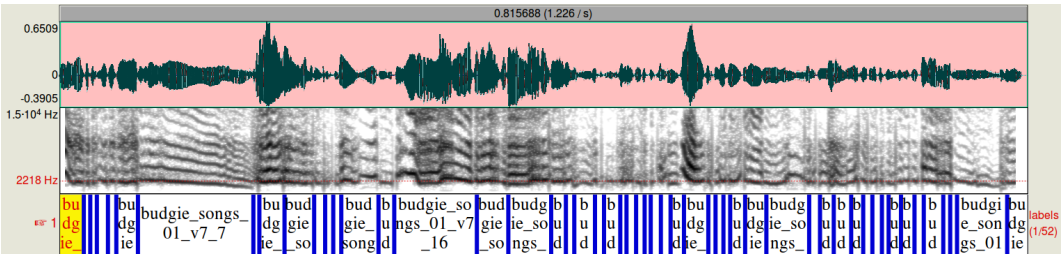
Figure A.1.: Exemplary visualisation of component class 1-4



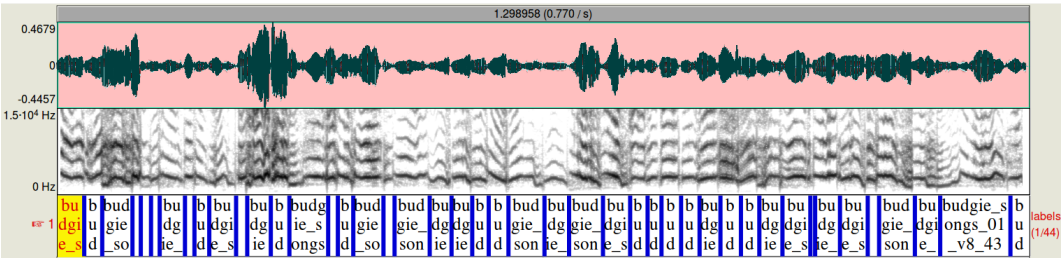
(a) v5



(b) v6



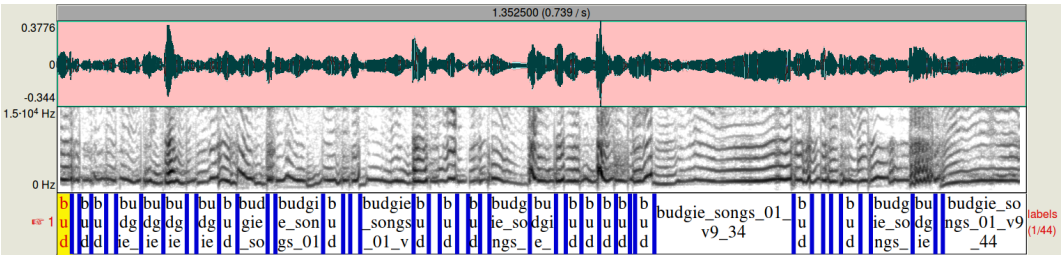
(c) v7



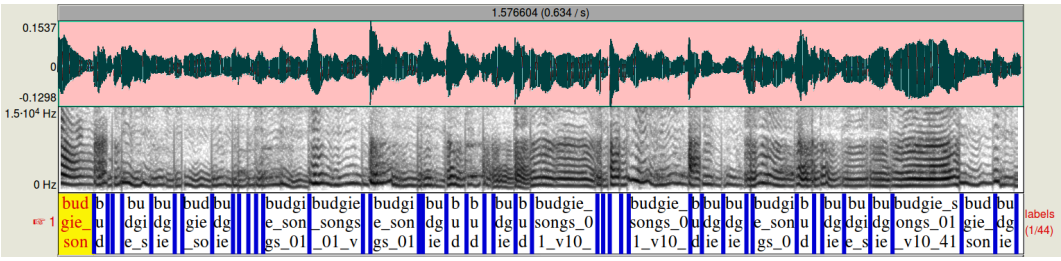
(d) v8

Figure A.2.: Exemplary visualisation of component class 5-8

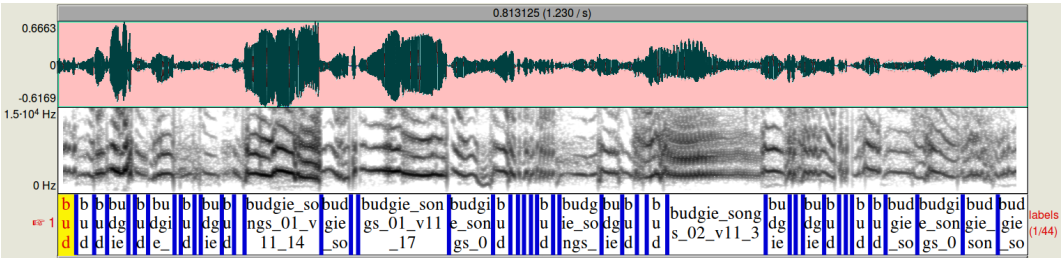
Appendix A. Appendix



(a) v9

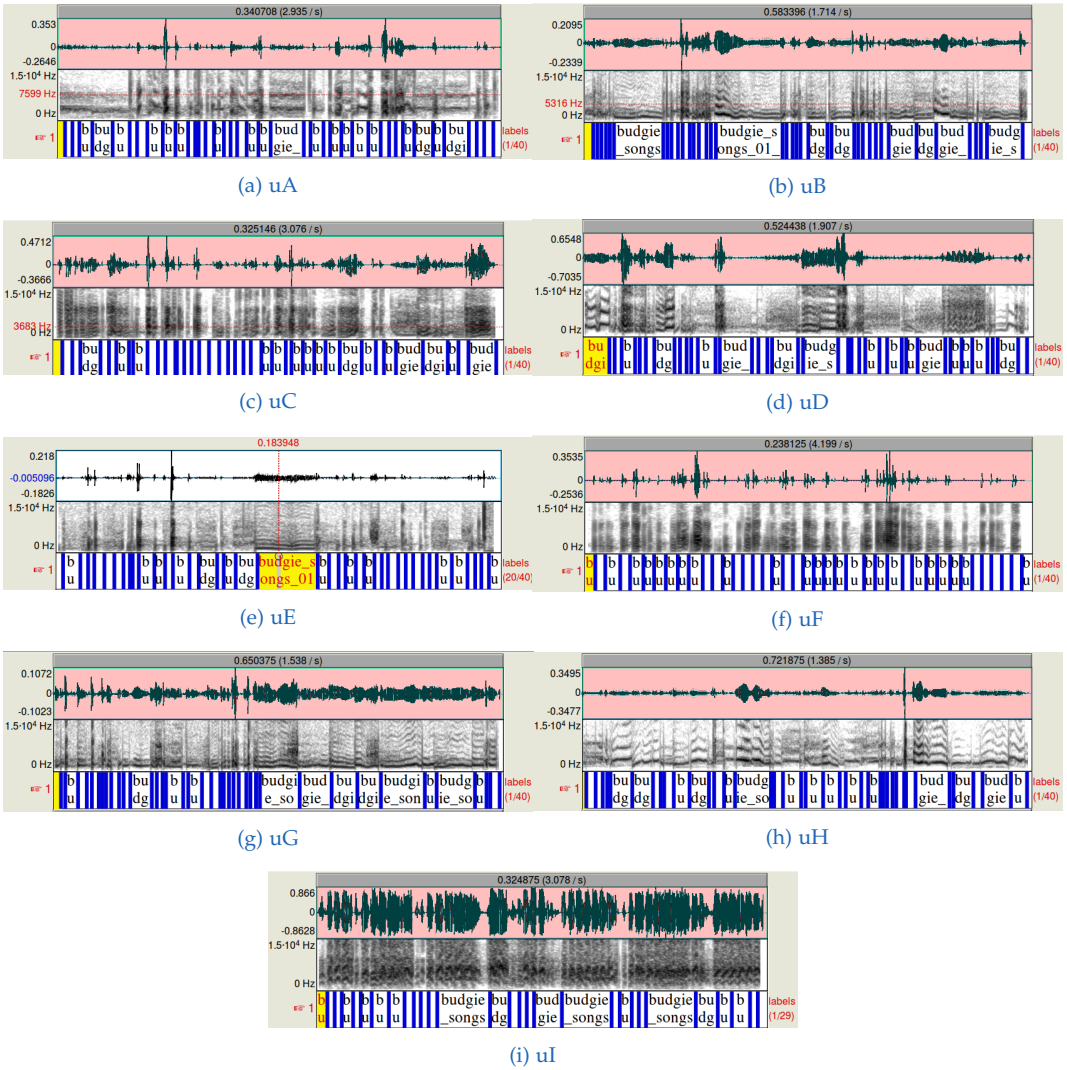


(b) v10



(c) v11

Figure A.3.: Exemplary visualisation of component class 9-11



(a) uA

(b) uB

(c) uC

(d) uD

(e) uE

(f) uF

(g) uG

(h) uH

(i) uI

Figure A.4.: Exemplary visualisation of unvoiced component class 1-9

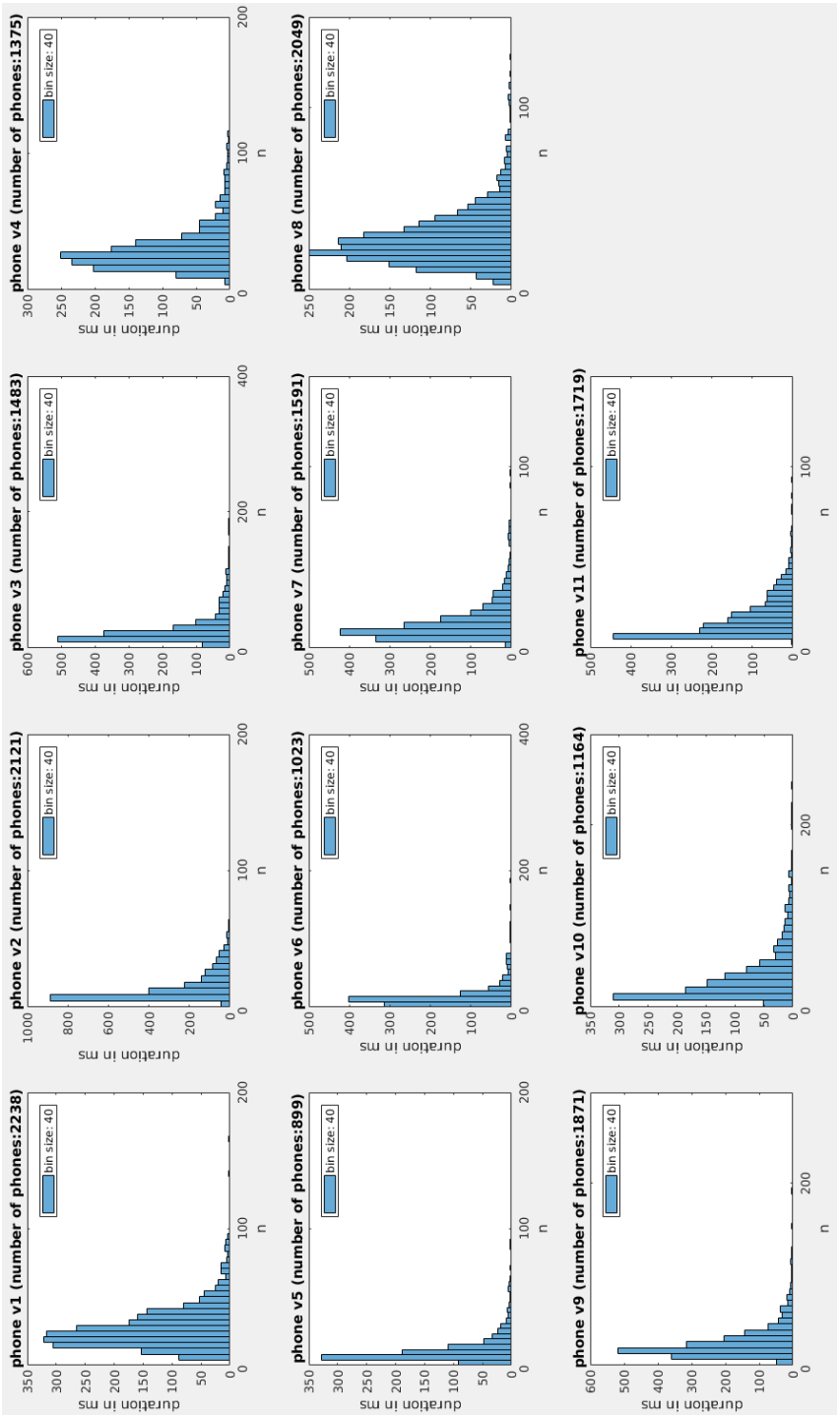


Figure A.5.: Duration histogram of voiced sounds

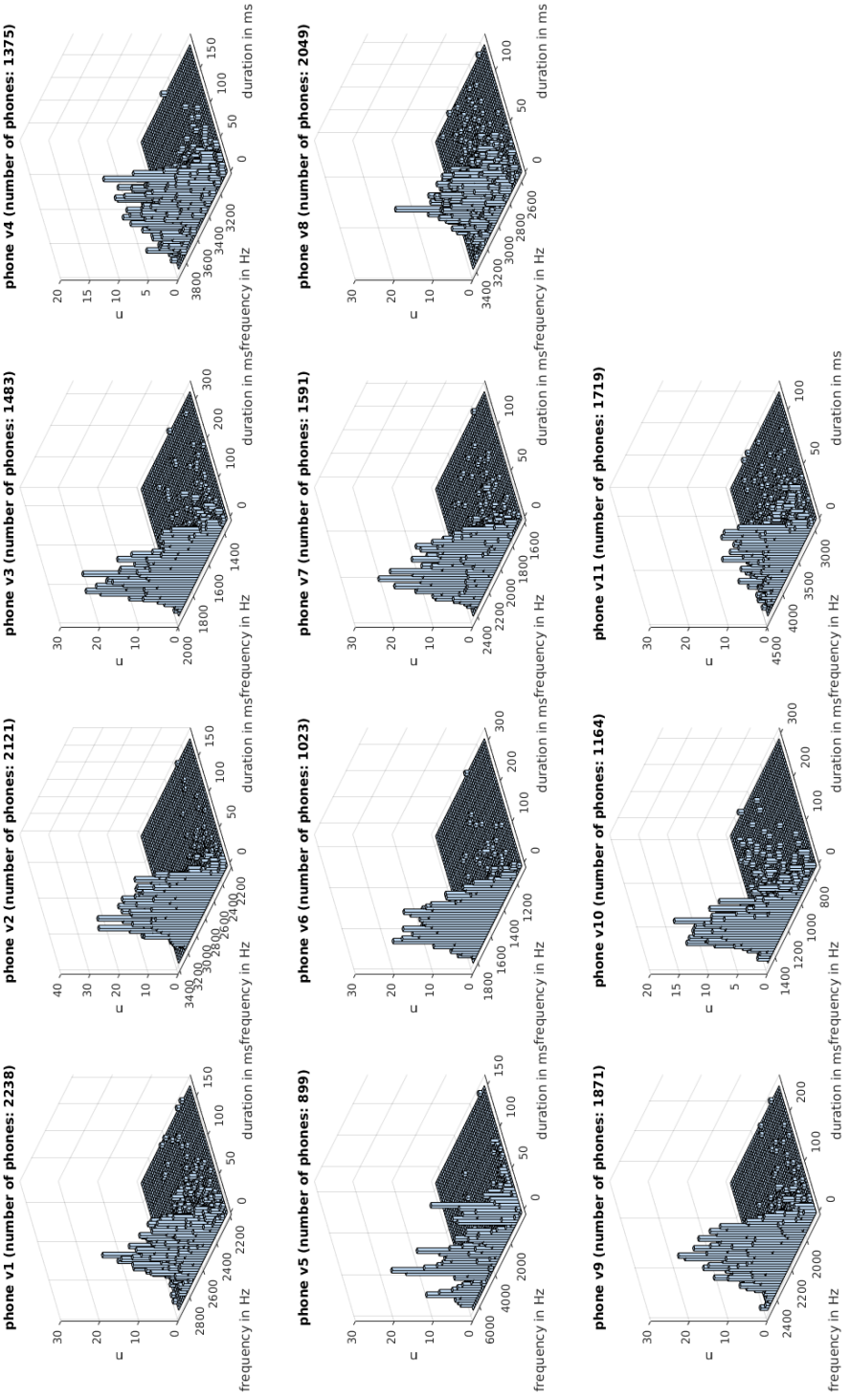


Figure A.6.: 3D-histogram of voiced sounds

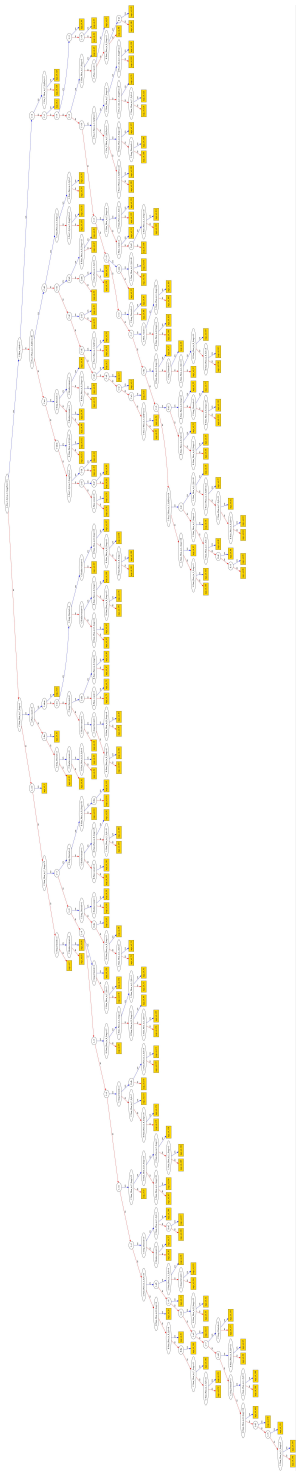


Figure A.7.: Decision tree of the third state of MGC features

Appendix A. Appendix

Acronyms:

AIC	Akaike information criterion
BIC	Bayesian information criterion
DFT	Discrete Fourier transformation
DTW	Dynamic time warping
EM	Expectation-maximization algorithm
F0	Fundamental frequency
HMM	Hidden Markov Model
HTK	Hidden Markov Model Toolkit
HTS	HMM-based Speech Synthesis System
LF0	Logarithmic fundamental frequency
LPC	Linear predictive coding
MFCC	Mel-frequency cepstral coefficient
MGC	Mel-generalized-cepstrum
NMF	Non-negative matrix factorisation
PDA	Pitch detection algorithm
RAPT	Robust Algorithm for Pitch Tracking
SNR	Signal-to-noise ratio
SPTK	Speech Signal Processing Toolkit
VVV	Ellipsoidal, varying volume, shape, and orientation
YIN	Fundamental frequency estimator for speech and music