Pitch Based Sound Classification

A master's thesis by

Andreas Brinch Nielsen

15 August 2005

Technical University of Denmark

Abstract

The fact that different sound environments need different sound processing is no secret, but how to select between the different programs is very different from hearing aid to hearing aid. Complete automatic and reliable classification is desirable, because many hearing aid users are not able to select programs themselves. In this project the emphasis is on classification based on the pitch of the signal, and three classes, music, noise and speech, is used. Unfortunately pitch is not straightforward to extract, and the first part of the project is about finding a suitable pitch detector.
A new pitch detector is suggested based on two existing algorithms, pattern match with envelope detection and the harmonic product spectrum. The new algorithm is compared to a Bayesian algorithm and HMUSIC, and is found to perform better for classification purposes.
Features are extracted from the signal produced by the pitch detector. Apart from the pitch itself, the error from the pitch detector is used to get a measure of how well the extracted pitch describes the signal, i.e. whether the signal is pitched or not. A total of 28 features, some overlapping, are suggested. A model is set up for classification to evaluate the features found. The Bayes classifier is used and during training an interesting property is discovered. The training error increases for high numbers of features. Maximum likelihood estimations should always result in decreasing training error for increasing dimensions of the model. The explanation is that the Bayes classifier is not trained for classification, but for the within class likelihood. When the data is not distributed like the model, it does not result in maximum likelihood in classification. A new model that ensures maximum likelihood in classification is suggested and compared to a generative and a discriminative model. A better performance than the generative and comparable to the discriminative is obtained.
Finally a model, using the new model and 5 features, is suggested. The validation classification error of this model is only 1.9 %. The influence of the pitch detector's precision on the classification is investigated. The error is clearly increasing for worse precision, but very little seems to be gained for higher precision than already used.

Keywords: Pitch detection, HPS, HMUSIC, feature extraction, classification, sound, music, noise, speech, Bayes, generative, discriminative.

Resumé

At forskellige lydmiljøer kræver forskellig behandling af lyden er ingen hemmelighed, men hvordan valg mellem forskellige programmer træffes er meget forskelligt fra høreapparat til høreapparat. Fuldstændig automatisk og pålidelig klassifikation er ønskværdigt, fordi mange brugere af høreapparater ikke er i stand til at skifte mellem programmerne selv. I dette projekt vil fokus ligge på klassifikation ved hjælp af lydens tone, og lyden vil blive inddelt i tre klasser, musik, støj og tale. Desværre er tonen ikke lige til at måle, og første del af projektet handler om at finde en passende tonedetektor.
En ny måde at detektere tonen på foreslås. Den er baseret på to eksisterende algoritmer, *pattern match with envelope detection* og *harmonic product spectrum*. Denne nye algoritme bliver sammenlignet med en Bayes algoritme og HMUSIC, og den nye viser sig at være den bedste.
Forskellige features bliver fundet baseret på signalet fra tonedetektoren. Ud over selve tonen bliver fejlen fra tonedetektoren brugt som et mål på, hvor godt tonen beskriver lydsignalet. 28 features, nogle mere forskellige end andre, bliver foreslået. En klassifikationsmodel opsættes og anvendes til at evaluere de forskellige features. Bayes klassifikationsmodellen bruges, og under træningen bliver en interessant egenskab opdaget. Træningsfejlen stiger, når der bruges mange features. *Maximum likelihood* estimeringer burde altid resultere i faldende træningsfejl. Forklaringen er, at modellen ikke trænes til at klassificere klasserne, men bliver trænet til at passe med klassernes sandsynlighedsfordelinger. Når data ikke er fordelt, sådan som modellen foreskriver, resulterer det ikke i *maximum likelihood* i klassifikationen. En ny måde at træne modellen på, der træner klassifikationen, bliver foreslået. Denne nye model sammenlignes med en generativ og en diskriminativ model. Den nye model præsterer bedre resultater end den generative model og sammenlignelige resultater med den diskriminative model.
Til sidst foreslås en endelig klassifikations model, der består af 5 features og bruger den nye metode til træning. Validerings-klassifikations-fejlen for denne model er kun 1,9 %. Betydningen af præcisionen for tonedetektoren undersøges. Det viser sig, at klassifikationsfejlen tydeligt bliver værre for dårligere præcision, men der er tilsyneladende ikke meget at hente for bedre præcision end den, der er blevet anvendt gennem projektet.

Nøgleord: Tonedetektion, HPS, HMUSIC, feature udtræk, klassifikation, lyd, musik, støj, tale, Bayes, generativ, diskriminativ.

Preface

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Science in Engineering at the Technical University of Denmark (DTU), Lyngby, Denmark.

Author is Andreas Brinch Nielsen (s001558).

Thesis supervisor is Prof. Lars Kai Hansen, Dept. of Informatics and Mathematical Modelling (IMM), DTU.
Thesis co-supervisor is Ph.d. Ulrik Kjems, Oticon A/S.

Thesis work was conducted partly at Oticon A/S, Strandvejen 58, and partly at Dept. of Informatics and Mathematical Modelling (IMM) from Feb. 2005 - Aug. 2005.

Contents

1 Introduction

Sound processing in hearing aids

Many different sounds are listened to every day. When at work you hear the noise of machines and computers, while you have to concentrate on your own work or someone talking to you. In your leisure time you can be outdoors with all kinds of sounds or you can be listening to music. In order for a hearing aid to make the best of each environment, different sound processing is necessary. For example when someone is speaking to you, intelligibility is the most important factor. This means the sound can be modulated in order to enhance the intelligibility of the speech. In other situations, like when listening to music, it is important to get the full range of sound. If music was treated like speech, it would distort the sound, and, on the other hand, if speech was treated like music you would understand less.
When amplifying the sound, it is a lot more difficult to ignore the sounds presented to you. This means that people using hearing aids are more sensitive to noise, and if the noise is amplified to get the full intelligibility of it, it would simply drive you mad. Different amplification schemes are necessary for different sound environments.

Classification in hearing aids

The listening situations in everyday life can be divided into classes. It is done a little differently from place to place, but the three main classes - music, noise and speech - are always included. Sometimes a combination of them is used, speech in noise is often used, and sometimes silence is a class of its own. The classes specify the different sound processing requirements.
In most hearing aids of today, speech is handled on its own because of the importance of this class. The concept is called Voice Activity Detectors or VAD. Other situations, like music, are handled with different programs that can be selected either directly on the hearing aid or using a remote control. Noise is simply handled with a volume control. Some early steps towards more advanced sound classification have been made, but many hearing aids still have the volume control and the program switch.
The problem with the manual switching between programs is, besides the convenience, that many hearing aid users are not accustomed to using technology in general. This means that they might not be capable of understanding the different programs or how to switch between them. If a user accidentally selects a wrong program, it can damage the experience of the hearing aid. Since it is already associated with difficulties to make people use hearing aids, for example because of embarrassment, this would be a pity.
The need for reliable automatic sound classification is obvious and much research goes into describing the different classes. What characterizes music, noise and speech? Some obvious characteristics can be thought of, but what about the difference between rap music and speech, or between the monotone humming of a cooling fan and music. In hearing aids it is mostly the energy levels of different frequency bands that are used.

Pitch detection

The pitch of sound is receiving growing attention, both in classification, but also in other research areas such as monitoring of machines. The pitch can tell us the melody

of music, if it is a male or female voice, and the speed of a running engine. It seems that much information can be gathered from this, rather simple, feature.
Another interesting property of the pitch is that it is very robust to modulation, caused by the room that you are in or by the connection when speaking on the phone. Speech on a phone compared to clean speech, is very different in the spectrum where only a narrow frequency band is left on the phone. The pitch however is unaffected by the phone line.
Even though the pitch is a simple concept to understand, it is unfortunately not so easy to extract. As mentioned before the pitch is not affected by, for example, a phone line, but the extraction of the pitch is affected a lot. This means that a pitch detector needs to be robust to changes of energy level in the different frequency bands. Many different kinds of pitch detectors have been suggested using very different approaches. It is not clear, however, which of them are superior. Especially the problem of pitch detection used for classification purposes is not very well researched.

Classification models

Many different kinds of models exist for classification purposes. The hidden Markov models are often used when dealing with time series data, because they include the serialized information. They can however be quite difficult to optimize because of their very complex error space. The Bayes classifier is a very common model because of its simplicity and ease of understanding, and it shows very good performance in many situations. Regression methods and neural networks are quite advanced methods that can adapt to any function. They can be hard to interpret though.

This project

The main goal of the project is to investigate the use of pitch in sound classification. The three main classes - music, noise and speech - will be used. In general a classification problem can be divided into two stages. The first stage is the feature extraction and the other is the actual classification. The feature extraction stage is often neglected and features are simply selected off the shelf. The second stage has to compensate for the bad features with an advanced classification model that can model a very advanced distribution. Complex models need large training sets to avoid poor training and generally make the classification system a lot more complex. If care is taken during the selection of features and very descriptive features are found, it can simplify the classification stage and make the system more efficient.
In this project the pitch will be used for the classification. First a good pitch detector will be found by comparing three pitch detectors of which one is a new combination of two existing algorithms. To make the comparison valid, parameters must be set to find the best pitch detector for classification, which is not the usual condition used for comparing pitch detectors.
Even though the pitch is extracted, there are still too many measurements to use directly in a classifier and features must be extracted based on the pitch signal. The pitch signal is examined thoroughly and a list of features is generated. More features, than can be used at once in the classification part are constructed, but the decision about which features to use is not made until the features are a part of the classification system.
For the classification, the rather intuitive and quite simple Bayesian model is used. An interesting property of the model is discovered and a new model will be suggested. This leads to a comparison of the new model to other existing methods.

Roadmap

In the introduction I have briefly described the motivation for starting this project.
Chapter 2 is about the pitch. The first section will be used to describe the pitch in general. Then the next three sections will present each of the pitch detectors. The first is the new combination of algorithms. Second pitch detector is a Bayesian algorithm and last is the HMUSIC algorithm. In the end of chapter 2 a comparison of parameters and a reference data set are found. Based on the comparison, a pitch detector is selected as the best one for classification.
In chapter 3 the features are found. The pitch and reliability signals produced by the selected pitch detector will be investigated in the first section. A way of separating true pitch estimations into so-called reliable windows is suggested in the second section. In section three all the features are presented. If features are logarithmically distributed or not, is investigated in section four.
In chapter 4 the database on which the classifier will be trained is presented together with the considerations done in the selections.
Chapter 5 is about classification models. In the first section the Bayes classifier will be presented and used on the sound database, then, in the second section, a new model will be presented. In section three some problems with the training of the new model is identified and solved. The new model is related to the existing issue of generative and discriminative models in section four and a comparison is performed in section five. In section six the best features are selected and a final model is suggested.
In chapter 6 the final model is evaluated. In the first section the features are presented and in the second section the misclassifications are identified. And finally, in section three, the performance degradation of choosing a simpler pitch detector is evaluated.
Chapter 7 contains the conclusion.
In the end the bibliography and the appendices are included.

2 Pitch detection

In this chapter, the pitch will be investigated. First section will be used to define the pitch and to show some general characteristics. The next three sections investigate three different methods for pitch detection. The first method is a new combination of two existing methods working in the frequency domain. The next is a Bayesian algorithm working in the time domain, and finally HMUSIC, an algorithm that divides the dimensions of the covariance of the signal into noise and signal. After each section a small evaluation of the algorithms is done. In section five, parameters for comparison of the three algorithms will be defined and a reference data set is introduced. Finally the three pitch detectors are compared. Based on the results, a single pitch detector will be selected and used for the remainder of the project.

2.1 Pitch theory

To understand the concept of pitch some general basics have to be understood. When physical structures are oscillating and producing a sound of a single tone, not only a single frequency will be present. Many frequencies will be present, but they will all be harmonically related to each other. Harmonically related means that each frequency will be at an integer multiple of the lowest frequency.

A simple experiment can be done with a string and a pulse generator. The string is attached with one end fixed and the other end connected to the pulse generator. When the frequencies of the pulse generator are changed some frequencies affect the string more strongly than others. These frequencies are said to be critical, and the lowest of these is called the fundamental frequency, ω_0. The string will move in a pattern as depicted in figure 2.1.1. When the frequency is increased to exactly double the fundamental frequency the string moves again, but now in a different pattern, figure 2.1.2. This frequency is called the first harmonic frequency, ω_1. And further it goes for triple the fundamental frequency, which is called the second harmonic, ω_2, and so forth.

$$\begin{aligned} \omega_1 &= 2\omega_0 \qquad \omega_2 = 3\omega_0 \\ \omega_i &= (i+1)\,\omega_0 \end{aligned} \tag{2.1.1}$$

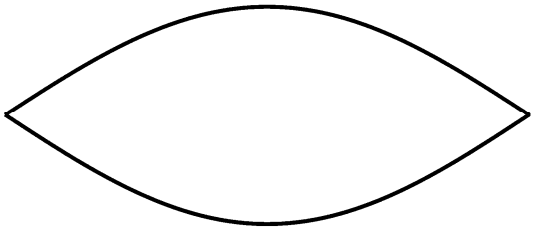


Figure 2.1.1: String oscillating at the fundamental frequency, ω_0.

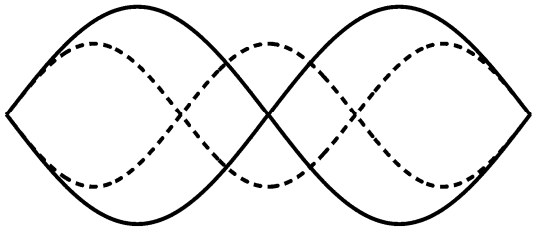


Figure 2.1.2: String oscillating at the first, ω_1, and second harmonic, ω_2, full and dashed respectfully.

The value of the fundamental frequency of the string depends on many things, such as the type of string, the length and the force it is being pulled by. When a string is excited, like on a violin or a piano, not only the fundamental frequency appears, but a number of harmonics will be present as well. The sound is heard as being one frequency, the fundamental frequency, and this percepted tone is referred to as the pitch. The value of the pitch is the value of the fundamental frequency. A model of a sound consisting of a fundamental and a number of harmonic frequencies is,

$$s(t) = \sum_{i=0}^{K} A_i \sin\left((i+1)\,\omega_0 t + \phi_i\right) \tag{2.1.2}$$

with A_i and ϕ_i being the amplitude and phase of the i'th frequency and ω_0 being the fundamental frequency. A plot of a signal containing the fundamental frequency and four harmonics, all with an amplitude of one and zero phase, looks like this,

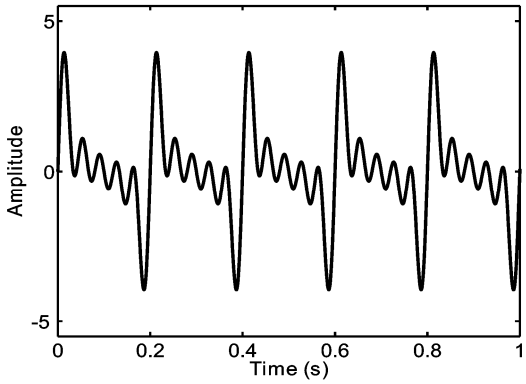


Figure 2.1.3: Synthetic time plot of a signal consisting of 5 sinusoids with equal amplitude.

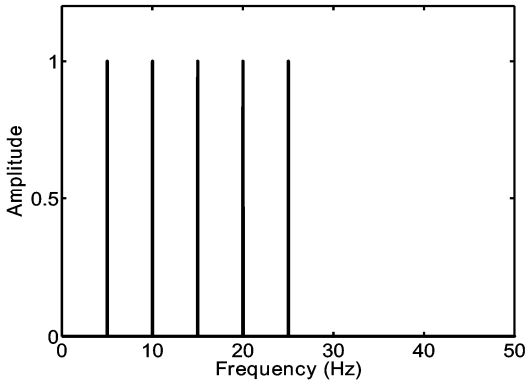


Figure 2.1.4: The spectrum of the signal to the left. Each frequency stands out clearly and the fundamental frequency is 5 Hz.

In real life the amplitude is, of course, not the same for all frequencies. A model with different amplitudes looks like this,

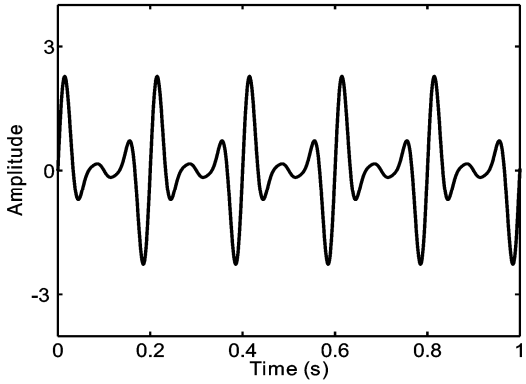


Figure 2.1.5: Synthetic time plot of a signal consisting of 5 sinusoids with different amplitudes.

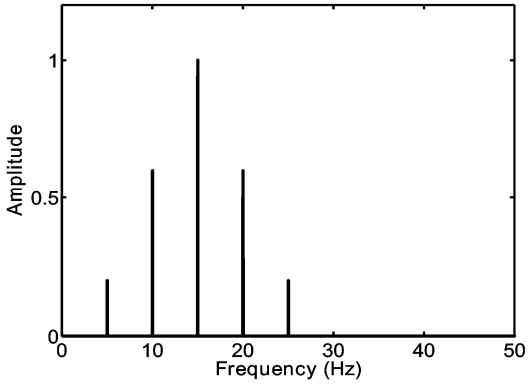


Figure 2.1.6: The spectrum of the signal to the left. The frequencies clearly have different amplitudes.

A sound of a single key on a piano has been recorded to show what a real signal looks like. The structure in figure 2.1.8 is apparent, and more than 10 harmonics can be seen in the plot. Also notice the very different amplitudes of the harmonics. In some cases some harmonics can disappear completely. This can also happen for the fundamental frequency. This does not mean that the pitch changes. The human ear perceives the pitch even if the fundamental frequency is not present.

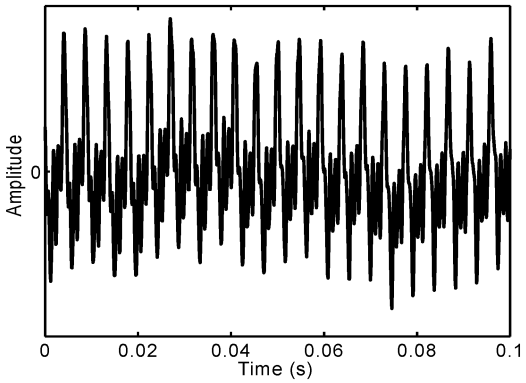


Figure 2.1.7: The note A at 220 Hz played on a piano.

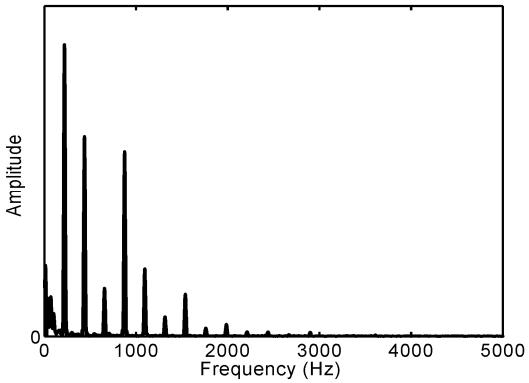


Figure 2.1.8: The spectrum of the figure on the left. The peaks at the harmonic frequencies are very clear.

Even though the pitch and the fundamental frequency seem to reflect the same thing this is not exactly the case. The pitch is the fundamental frequency together with the harmonics and is related to human perception, a conceptual thing, whereas the fundamental frequency is a physical characteristic [Jørgensen, 2003, chap. 3]. Further more the pitch can be identified even though the fundamental frequency is missing and the pitch can be changed even if the fundamental frequency is not. When talking in the phone only a limited bandwidth, which does not include the low frequencies of the voice, is available. Still the pitch of the voice does not sound higher than when talking directly. By inserting tones in between the harmonics you can change the pitch, as experienced by humans, even though the lowest frequency is not changed. This is beyond the scope of this paper though, and only the pitch similar to the fundamental frequency is of interest here.

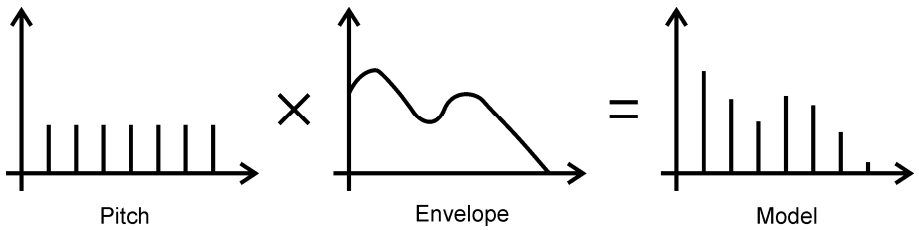


Figure 2.1.9: The relation between pitch and envelope.

If the peaks of the spectrum are connected the resulting line is called the envelope of the signal. The model is often separated in two parts. A part with the fundamental frequency and harmonics all with uniform amplitude, this is the pitch part. The other part contains the envelope which modulates the first part. When these two parts are combined the result is the complete signal. When detecting the pitch, the envelope is not relevant, but because you only have the complete signal you have to account for the envelope in the detector. The pitch is somewhat independent of the envelope and visa versa. For example when pronouncing the letter ‘u’ it has a certain envelope. The pitch of the sound can be changed by saying ‘u’ with a low or a high pitch. This only changes the pitch part, whereas the envelope is constant. The other way around can be to say ‘a’ and ‘u’ with the same pitch. ‘u’ and ‘a’ has different envelopes, but the pitch will remain the same.

When identifying the pitch manually, the most obvious way is to look at the spectrum. The peak with the lowest frequency is found and this peak lies at the fundamental frequency. Sometimes the fundamental frequency is not present. Then it can be found

as the distance between harmonics or as the highest common divisor of the peak frequencies.

2.1.1 Behaviour of the pitch in speech

People use a wide range of different sounds when communicating [Poulsen, 1993]. The sounds can coarsely be divided into two groups, the voiced and the unvoiced sounds. Voiced sounds is when a tone is heard like in the letter ‘a’, and is the kind of sound used when singing. Unvoiced are sounds close to white noise like the letter ‘s’ and ‘h’. Whether a sound is unvoiced or voiced is determined when the air passes the vocal cords. The voiced sounds are generated when the vocal cords open and close in a periodic pattern, the fundamental frequency. Unvoiced sounds are generated if the vocal cords are firm and narrow. Then a turbulent airflow is generated causing the unvoiced sound. After the vocal cords both the voiced and unvoiced sounds are shaped by the mouth and lips, but regardless the voiced/unvoiced structure remains. Unvoiced segments will be close to white noise with a flat spectrum, whereas voiced segments show a very clear harmonic structure. The spectrum of the voiced sound can be modelled in the same way as the physical sounds with an envelope and a pitch. A plot of a voiced sound is shown below.

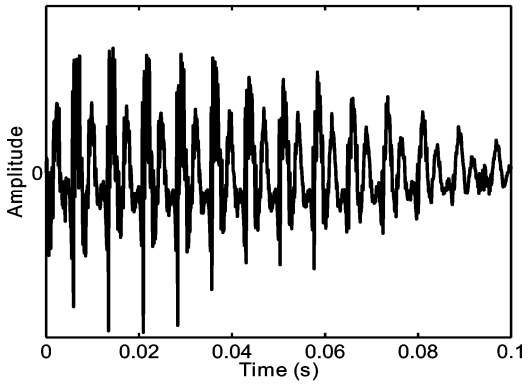


Figure 2.1.10: 100 ms of speech sampled at 10 kHz. The sound ‘ea’ from the word ‘easy’.

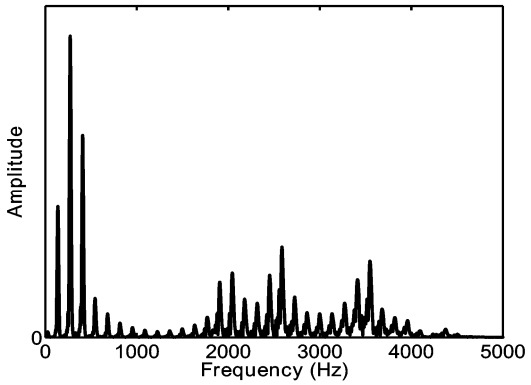


Figure 2.1.11: Spectrum of the signal to the left. The structure is very clear though some noise is present in between the harmonics. The envelope is also clear.

Only in the voiced sounds a pitch can be found. When we speak, both unvoiced and voiced sounds are used and this means speech will show parts with pitch and parts without pitch.

2.1.2 Classification based on pitch

The reason why the pitch is so interesting is that the pitch of the three classes, speech, music, and noise, behaves differently. First of all a single pitch is not present in noise. Noise consists of many frequencies not harmonically related to one another. A noise example can be seen in figure 2.1.12 and figure 2.1.13.

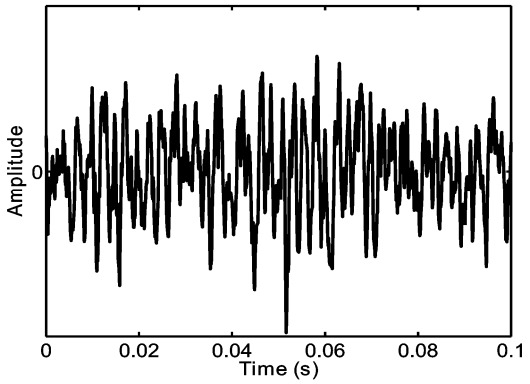


Figure 2.1.12:100 ms of noise sampled at 10 kHz. It is noise from a café, including speech babble and other noises.

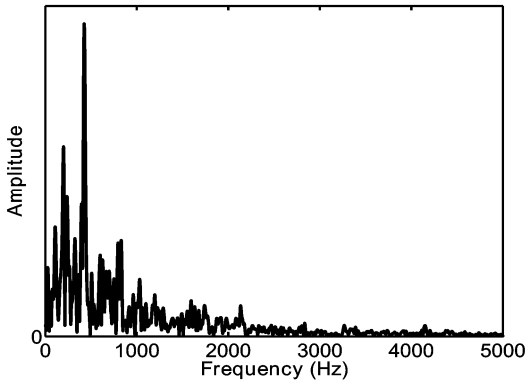


Figure 2.1.13: Spectrum of the signal to the left. There is no apparent pitch structure.

Music is almost always pitched. Even though many tones may occur a dominating pitch will usually be present. The human voice changes between pitch and unpitched sounds. This gives a general clue that the knowledge about if pitch is present or not can be used for classification. The dynamic behaviour of the pitch is also interesting. The pitch in music changes in steps and between the steps the pitch is very constant. The opposite goes for speech. In speech the pitch does not make steps, but changes constantly. The features of the pitch will be investigated in the next chapter, but first the pitch must be detected.

2.1.3 Pitch detection requirements

In order to make the search for a pitch detector possible some objectives must be specified. First of all a search space must be specified, here this means a range of possible frequencies. Since speech is the most important of the three classes, because speech is crucial for the communication between people, this is what decides the range. A range from 50 to 400 Hz assures that female, children and male voices are considered [Poulsen, 1993], [25]. The pitch is detected on a window and the size of it must be chosen, and is chosen to be 100 ms. This might seem large, but for the low pitch of 50 Hz only 5 periods are present during this window. The size is influenced by work done with FFT on speech. The lobe width of the peaks is dependent on the window size and gets bigger the smaller the window. In general, when doing classification, the smaller the window the better because it gives a quicker decision horizon. The classification will focus on the dynamics of the pitch though and the pitch does not change rapidly over time which means that the change in pitch during a window of 100 ms should be very small in most cases. To get a fluid transition of the pitch, overlapping of 75 ms is used. This means that a pitch value every 25 ms depending on the last 100 ms is found.
The resolution of the pitch detection algorithm is set to 1 Hz. Changes smaller than 1 Hz is hard to hear and will not give any extra information.

2.2 New pitch detection algorithm combining pattern match and HPS

The algorithm suggested here is a combination of two well known algorithms. Both of them work in the frequency domain. The harmonic product spectrum [de la Cuadra, 2001] is a very efficient algorithm and pattern match with envelope detection [Bach, 2004, app. A] is a very reliable algorithm. By combining them an efficient and reliable algorithm can be constructed. Both algorithms will be described before the combination of the two is presented.

2.2.1 Harmonic product spectrum

This algorithm exploits a very simple characteristic in the frequency domain. As explained earlier the fundamental frequency is related to the harmonics in a very simple manner. The harmonics are integer multiples of the fundamental frequency. If the spectrum is downsampled by 2, the first harmonic will align with the fundamental frequency. If the spectrum is downsampled by 3 the second harmonic will align with the fundamental frequency. This can be continued for as many harmonics as wanted. The principle is illustrated in figure 2.2.1. If the original spectrum and the downsampled ones are multiplied, the harmonic product spectrum (HPS) is realized. The HPS can be done with as many downsampled signals as necessary. The constant that controls this is usually called *R*, meaning that the last downsampling is by *R*, thus covering *R*-1 harmonics.

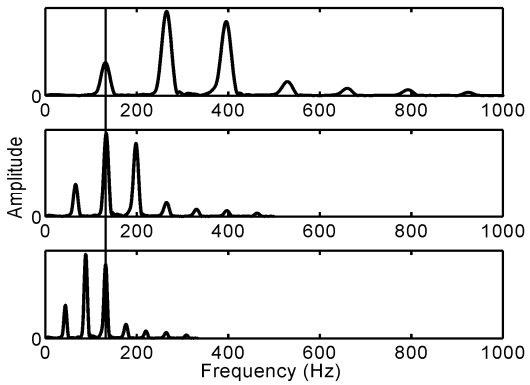


Figure 2.2.1: Original and downsampled by 2 and 3.

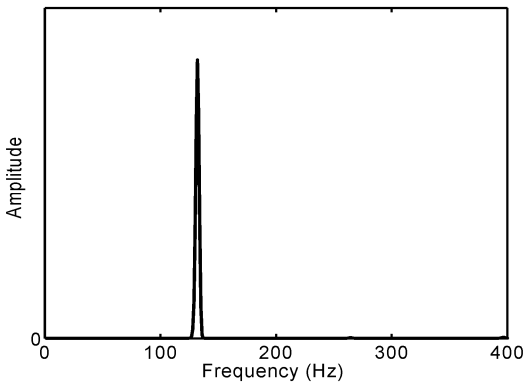


Figure 2.2.2: Multiplying the downsampled signals gives the Harmonic product spectrum. Here up to 4 downsamplings is used, *R*=5.

The HPS will have a peak at the fundamental frequency as can be seen in figure 2.2.2, and the pitch can be read immediately as maximum value. This is a very fast way of finding the pitch.

A problem with this algorithm is specifying a value for the constant *R*. The maximum pitch frequency together with the sampling frequency limits the value to,

$$R \leq \left\lfloor \frac{F_{sampling}}{2F_{maxpitch}} \right\rfloor \tag{2.2.1}$$

$\lfloor \bullet \rfloor$ denotes the floor, i.e. the number rounded to the smallest integer.

The spectrum tends to deviate more in the high frequencies than the lower ones from the ideal harmonic model. When choosing a high value of *R*, the HPS is depending more on the high frequencies. This is an argument for choosing *R* to be relatively low.

If R is chosen low and the envelope of the signal is large at high harmonics the algorithm tends to give double the frequency. This problem is very common for pitch detectors and is called doublings. The opposite, with half the pitch returned, is called halvings.
The pitch and envelope of a signal is independent as explained in the previous section. It is the combination of the pitch and the envelope that determines how many harmonics are present. The envelope will be more or less constant in a given environment and will cut off the frequencies above a certain threshold. This means that when the pitch gets higher more and more harmonics will be cut off by the envelope. Therefore the number of harmonics is dependent on the pitch and varies with it, which makes it difficult to choose a fixed number for R.
An illustration of the doubling problem is plotted below.

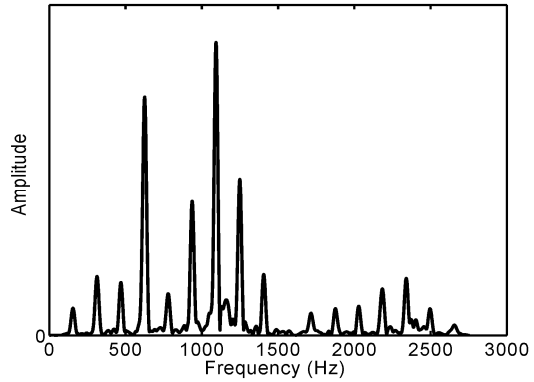


Figure 2.2.3: Spectrum of signal. Pitch is approximately 155 Hz. Harmonics beyond the 4th still have large amplitudes.

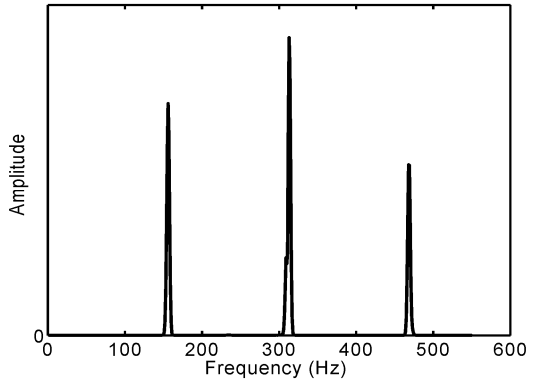


Figure 2.2.4: Harmonic product spectrum, R=5 of the signal to the left. Double the true pitch is returned.

The frequency returned by the algorithm is twice the frequency of the pitch. Even though the major peak is present at twice the pitch, a clear peak is present at the true pitch. This property is used in the combined algorithm.

2.2.2 Pattern match with envelope detection

The algorithm uses a model of the harmonic spectrum. An ideal representation of a harmonic spectrum contains a peak at the fundamental frequency and each of the harmonics. A model of the spectrum, $\boldsymbol{S}$, can be specified as,

$$\hat{\boldsymbol{S}} = \sum_{i=0}^{N} A_i \delta\left(f - (i+1)\,\omega_0\right) \tag{2.2.2}$$

with N being the number of harmonics, $\delta(\bullet)$ the Dirac delta function and ω_0 the fundamental frequency. Because of the finite window length this is not what we see even in the ideal case. Because of spectral leakage, lobes, instead of delta functions, will be present in the plot. Only the main lobe will be modelled, but this is not a big simplification because of the attenuation in the side lobes. A bump function will be used to model the main lobe. Different bump functions can be used, but in this project the main lobe from a synthetic signal has been sampled to get the real form. The width of the main lobe depends on the window length and the type of window used. Because of the attenuation of the side lobes, the Hanning window is used. The bump function is illustrated in figure 2.2.5.

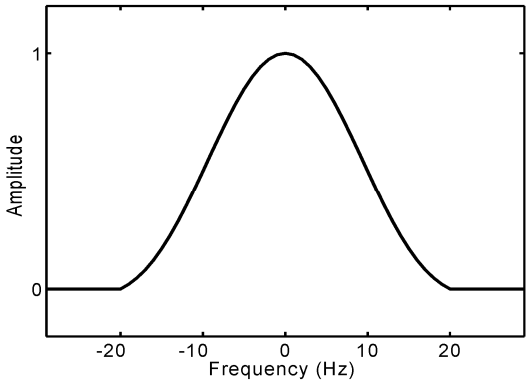


Figure 2.2.5: The bump function, *b*(*f*), is a sampling of the normalized main lobe.

In the model the Dirac delta function is exchanged with the bump,

$$\hat{\boldsymbol{S}} = \sum_{i=0}^{N} A_i \boldsymbol{b}\left(f - (i+1)\,\omega_0\right) \tag{2.2.3}$$

The variables of this model are the amplitudes, A_i, and the fundamental frequency, ω_0. The fundamental frequency is found by gridsearching the relevant frequency range for the smallest error using the sum-square-error function,

$$E_\omega = \frac{1}{2}\left|\boldsymbol{S} - \hat{\boldsymbol{S}}_\omega\right|^2 \tag{2.2.4}$$

where $\boldsymbol{S}$ is the FFT of the signal and $\hat{S}_\omega$ is the model with $\omega_0 = \omega$. The amplitudes are optimized for each frequency. In the error function above, each bump is independent of the rest of the signal and can be optimized on its own. The sum square error between two bumps at the same frequencies is directly proportional to the difference between the amplitudes of the two bumps,

$$\frac{1}{2}\left|A_1\boldsymbol{b} - A_2\boldsymbol{b}\right|^2 = \frac{1}{2}\left(A_1 - A_2\right)^2\left|\boldsymbol{b}\right|^2 \tag{2.2.5}$$

Instead of optimizing the amplitudes using the error of the complete signal, it can be calculated on the amplitudes of the bumps and the values of $\boldsymbol{S}$ at the fundamental and harmonic frequencies. The cost function is now defined as,

$$E_A = \frac{1}{2}\sum_{i=0}^{N}\left(\boldsymbol{S}\left((i+1)\,\omega_0\right) - A_i\right)^2 \tag{2.2.6}$$

If the cost function is simplified in this way the amplitudes are given directly by the values of $\boldsymbol{S}$ at the fundamental and harmonic frequencies. This is a coarse simplification and is valid only when the bumps of $\boldsymbol{S}$ and $\hat{\boldsymbol{S}}$ are aligned. They will only be aligned for a single frequency, the pitch, but when they are not aligned, the error of equation (2.2.4) will get worse. It means that errors at other frequencies than the pitch gets worse, but this is actually the point of it all, since the pitch is the right frequency. Equation (2.2.6) is only used for optimizing the amplitudes and equation (2.2.4) is still used in the gridsearch for the frequency.

Envelope detection

An inherent problem with the algorithm arises when the amplitudes can be chosen unrestrictedly. In this case the algorithm will always fit half the pitch better than the true pitch. This is because the noise in between harmonics can be modelled as well. When overfitting the data, the envelope will show a sawtooth behaviour because the amplitude of the noise in between harmonics is smaller than the amplitude of the

peaks. This kind of behaviour can mathematically be described by the envelope having a large second derivative, and this must be avoided.
To calculate the second derivative of the envelope demands that a smooth envelope is found. This is not a simple task and was done with splines in [Bach, 2004, app. A]. Instead a simplified approach is suggested.
To get zero second derivative, which is the best case, a function with a constant gradient is needed. If looking at a single peak this means that the optimal amplitude of this peak is on the straight line connecting its neighbour peaks to the left and right. The distance from this optimal amplitude is added to the cost function and (2.2.6) becomes,

$$\begin{aligned}
E_{A_0} &= \frac{1}{2}\left(\frac{3A_1 - A_2}{2} - A_0\right)^2 + \frac{1}{2}\left(S(\omega_0) - A_0\right)^2 \\
E_{A_i} &= \frac{1}{2}\left(\frac{A_{i-1} + A_{i+1}}{2} - A_i\right)^2 + \frac{1}{2}\left(S((i+1)\omega_0) - A_i\right)^2 \\
E_{A_N} &= \frac{1}{2}\left(\frac{3A_{N-1} - A_{N-2}}{2} - A_N\right)^2 + \frac{1}{2}\left(S((N+1)\omega_0) - A_N\right)^2 \\
E_A &= \sum_{i=0}^{N} E_{A_i}
\end{aligned} \tag{2.2.7}$$

The optimal amplitudes in the ends are found by subtracting half the distance between the two closest peaks from the closest peak. This approach seems to work better in experiments than extrapolating the optimal amplitude exactly by subtracting the full distance. Equation (2.2.7) is linear and to find the minimum the derivatives are found and set to 0. This gives,

$$\begin{aligned}
S(\omega_0) &= \frac{9}{4}A_0 - 2A_1 + \frac{3}{4}A_2 \\
S(2\omega_0) &= -2A_0 + \frac{9}{2}A_1 - \frac{7}{4}A_2 + \frac{1}{4}A_3 \\
S((i+1)\omega_0) &= \frac{1}{4}A_{i-2} - A_{i-1} + \frac{5}{2}A_i - A_{i+1} + \frac{1}{4}A_{i+2} \\
S(N\omega_0) &= \frac{1}{4}A_{N-3} - \frac{7}{4}A_{N-2} + \frac{9}{2}A_{N-1} - 2A_N \\
S((N+1)\omega_0) &= \frac{3}{4}A_{N-2} - 2A_{N-1} + \frac{9}{4}A_N
\end{aligned} \tag{2.2.8}$$

See appendix for details.
These equation needs to be solved to find A_i. This can be done in matrix form,

$$\boldsymbol{S} = \boldsymbol{K} \cdot \boldsymbol{A} \Leftrightarrow \boldsymbol{A} = \boldsymbol{K}^{-1} \cdot \boldsymbol{S} \tag{2.2.9}$$

This is how the amplitudes are found using a simplified form of envelope detection. The sum square error, (2.2.4), is still used in the gridsearch for the pitch.
This algorithm works quite well and does not suffer that much from halvings as without envelope detection.
A problem with the algorithm is that it is quite demanding computationally because of the gridsearch.

2.2.3 Pattern match and HPS combined

The harmonic product spectrum is fast, but suffers from doublings. The pattern match with envelope detection has good performance. It is quite slow though and suffers a bit from halvings. This suggests that a combination of the two could be a good idea. If one algorithm returns the double and the other returns the half it should, intuitively, be possible to find the true pitch.
As stated earlier the harmonic product spectrum has a peak at the true pitch. This is not always the biggest peak though. The combined algorithm uses the harmonic product spectrum initially. It identifies the three biggest peaks. This normally includes the double and the true pitch. It then finds a small interval around these frequencies and searches them using the pattern match with envelope detection algorithm. The running time of the harmonic product spectrum is much less than the other so the extra running time here has no significance. The running time of the pattern match algorithm is reduced greatly because of the reduced grid to be searched. Another gain is that in many circumstances half the pitch is avoided giving a slightly better performance.

2.2.4 Advantages & disadvantages of the combined pitch detection algorithm

The advantage of converting the data to the frequency domain is that the structure of the data becomes very clear. When looking at a spectrum of speech it is immediately evident that it is actually speech. It is this structure that is exploited in the algorithm. This makes the algorithm very easy to understand and thus makes it easier to tweak and enhance.
Another advantage of the algorithm is that it is relatively fast. It is about 4-5 times slower than the HPS on its own, but it is 10-15 times faster than plain gridsearch and this is achieved while improving accuracy.
Doubling and halvings are accounted for, and the effects are to some extent neutralized, so this algorithm is less troubled by them than other algorithms.
A major disadvantage of going to the frequency domain is the modulation inferred by the window. In this project a Hanning window has been used. The Hanning window has a wider main lobe than the rectangular window, but has better attenuation in the side lobes. The width of the main lobe is only dependent on the window length in seconds and of course the type of window. It does not depend on the sampling frequency. The width of the main lobe of the Hanning window is 4/L Hz. A window size of 100 ms has been used, which gives a main lobe of 40 Hz. This does not imply that the accuracy of the spectrum is 40 Hz, but it means that when frequencies are closer than about half the main lobe they start to affect each other. When getting even closer the peaks can no longer be separated from each other and looks as if only a single frequency is present. The 40 Hz seems to be a lot, but as the frequencies of interest are separated by at least 50 Hz (the minimum pitch) this is not directly a problem. Plots are presented below to visualize the problem.

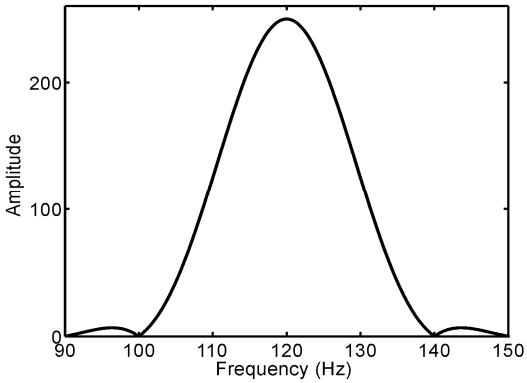


Figure 2.2.6: The value of a single frequency can easily be found in spite of the lobe. Here the frequency is 120 Hz.

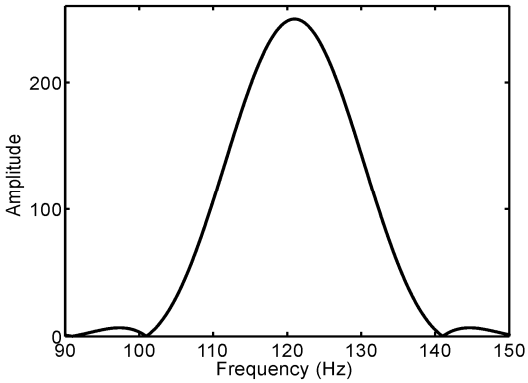


Figure 2.2.7: Here the frequency is 121 Hz.

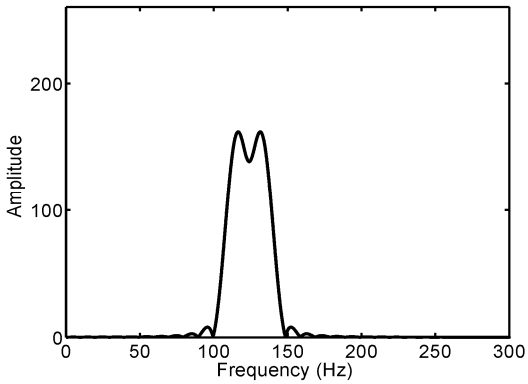


Figure 2.2.8: The lobe affects the ability to separate two frequencies. Here frequencies at 120 and 128 Hz can be separated, but the peaks lie at 116 and 132 Hz in the plot.

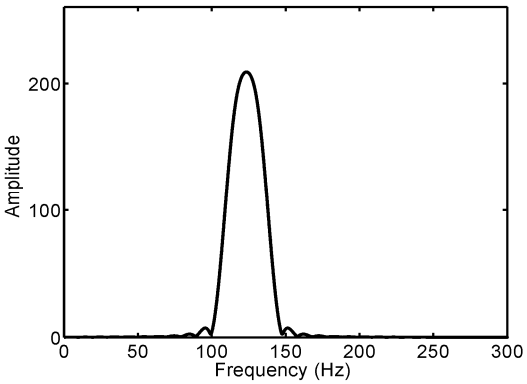


Figure 2.2.9: Here frequencies of 120 and 127 Hz can not be separated. Note the amplitude which is bigger than in the figure to the left.

The plots show that the accuracy is not the problem if only a single peak is present. The frequencies of 120 Hz and 121 Hz can easily be identified. The problem of two peaks being close together gives another resolution. If they are closer than 8 Hz they melt together and before they melt together the position of the peaks is changed. Another problem in regard to this is that even though all frequencies have equal amplitude, the peak in figure 2.2.9 shows higher amplitude than in figure 2.2.8. This means that if multiple frequencies are close together, even though each of them has smaller amplitude than the dominating frequency they might join up and appear bigger in the spectrum. This might be a problem for specific noise patterns.
If only the speech structure is considered there will be no problems by the windowing. The minimum frequency of interest is 50 Hz and causes a minimum separation of harmonics of 50 Hz as well. With a main lobe of 40 Hz this is fine.
The resolution of the algorithm is directly a function of the FFT that is being used. To obtain the resolution of 1 Hz, an FFT of the same length as the sampling frequency is used. If better resolution is desired the FFT must be made longer. This means that if very accurate pitch detection is wanted, a very large FFT must be used and the algorithm becomes cumbersome.

2.3 Bayesian pitch detector

This method works in the time domain. A model of the signal which consists of sinusoids and noise is used. By fitting the model to the signal, a likelihood of the fit can be found [Hansen, 2002], [Petersen, 2003]. This likelihood can be found for each frequency of interest and the one with the highest likelihood is the model that fits best and the pitch from the model is assigned to the signal.
The big challenge in this approach is of course to find the likelihood. The model is basically the well known harmonic model with added noise,

$$\hat{y}(t) = \sum_{k=1}^{K} A_{2k-1} \sin(k\omega_0 t) + A_{2k} \cos(k\omega_0 t) + e(t) \tag{2.3.1}$$

or in matrix notation

$$\begin{aligned} &\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{A} + \boldsymbol{e} \\ &\boldsymbol{A}_{2K\times 1} = \begin{bmatrix} A_1 & A_2 & \cdots & A_{2K} \end{bmatrix}^T, \quad \boldsymbol{t}_{T\times 1} = \begin{bmatrix} t_0 & t_1 & \cdots & t_{T-1} \end{bmatrix}^T \\ &\boldsymbol{X}_{T\times 2K} = \begin{bmatrix} \sin(\omega_0 \boldsymbol{t}) & \cos(\omega_0 \boldsymbol{t}) & \sin(2\omega_0 \boldsymbol{t}) & \cdots & \cos(K\omega_0 \boldsymbol{t}) \end{bmatrix} \end{aligned} \tag{2.3.2}$$

$\boldsymbol{e}$ is zero mean noise with variance, σ^2, $\boldsymbol{t}$ is the times of each measurement, and T is the length of the signal. The equation is a bit different from the usual model because it contains both a sine and a cosine for each frequency, but none of them has a constant for the phase. If a sine and a cosine with the same frequency are added, the result is a sine with the same frequency, but with a phase that can be controlled by the amplitudes of the two.
The algorithm finds the number of harmonic frequencies besides the pitch itself. The likelihood that is interesting is $P(\omega_0, K \mid \boldsymbol{y})$ where K is the number of frequencies.
This can be converted by Bayes' theorem to

$$\begin{aligned} P(\omega_0, K \mid \boldsymbol{y}) &= \frac{P(\boldsymbol{y} \mid \omega_0, K) P(\omega_0, K)}{P(\boldsymbol{y})} \\ &= \frac{P(\boldsymbol{y} \mid \omega_0, K) P(\omega_0, K)}{\sum P(\boldsymbol{y} \mid \omega_0, K) P(\omega_0, K)} \end{aligned} \tag{2.3.3}$$

In the model the only term with a probabilistic behaviour is the noise, $\boldsymbol{e}$. This means that the likelihood of the signal can be found as the likelihood of the difference coming from the error. It can be found like this,

$$P(\boldsymbol{y} \mid \omega_0, K) = P(\boldsymbol{y} \mid \sigma^2, \boldsymbol{b}, \boldsymbol{X}, \omega_0, K) = \left(\frac{1}{2\pi\sigma^2} \right)^{T/2} e^{-\frac{1}{2\sigma^2} |\boldsymbol{y} - \boldsymbol{X}\boldsymbol{b}|^2} \tag{2.3.4}$$

where T is the length of the observed signal, $\boldsymbol{y}$. The object of interest is neither the variance of the error nor the amplitudes of the sines. Instead the marginal distribution of the two is found by integrating them together with the prior knowledge of their distributions,

$$P(\boldsymbol{y} \mid \omega_0, K) = P(\boldsymbol{y} \mid X, K) = \int_{\boldsymbol{B}} \int_{\Sigma^2} P(\boldsymbol{b}, \sigma^2) P(\boldsymbol{y} \mid \sigma^2, \boldsymbol{b}, \boldsymbol{X}, \omega_0, K) \, d\sigma^2 \, d\boldsymbol{b} \tag{2.3.5}$$

When using the normal-inverse-gamma distribution as prior, the integral can be solved and gives,

$$P(\boldsymbol{y}\mid\omega_0,K)=\left(\frac{|\boldsymbol{V}_P|a^d}{|\boldsymbol{V}|(a_P)^{d_P}\pi^T}\right)^{1/2}\frac{\Gamma\left(\frac{d_P}{2}\right)}{\Gamma\left(\frac{d}{2}\right)} \tag{2.3.6}$$

The equation gives the likelihood of $\boldsymbol{y}$ given the base frequency ω_0 and the number of harmonics, K, with the following definitions [Hansen, 2002],

$$\begin{aligned}
&d=3\\
&d_P=3+T\\
&\nu=\frac{T}{Tr(\boldsymbol{XX}')}\\
&\boldsymbol{V}=\nu\boldsymbol{I}=\frac{T}{Tr(\boldsymbol{XX}')}\boldsymbol{I}\\
&\boldsymbol{V}_P=\left(\frac{1}{\nu}\boldsymbol{I}+\boldsymbol{X}'\boldsymbol{X}\right)^{-1}=\left(\frac{Tr(\boldsymbol{XX}')}{T}\boldsymbol{I}+\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\\
&a=\sigma_y^2=\frac{\boldsymbol{y}'\boldsymbol{y}}{T}\\
&a_P=(T+1)\sigma_y^2-\boldsymbol{y}'\boldsymbol{X}\boldsymbol{V}_P\boldsymbol{X}'\boldsymbol{y}
\end{aligned} \tag{2.3.7}$$

The logarithm is taken to simplify the equations and constant parts are neglected because the final use will be divided by the sum over all.

$$\begin{aligned}
\log P(\boldsymbol{y}\mid\omega_0,K)=&\frac{3}{2}\log\sigma_y^2-\frac{1}{2}\log\left|\frac{Tr(\boldsymbol{XX}')}{T}\boldsymbol{I}+\boldsymbol{X}'\boldsymbol{X}\right|+\frac{K}{2}\log\frac{Tr(\boldsymbol{XX}')}{T}\\
&-\frac{3+T}{2}\log\left((T+1)\sigma_y^2-\boldsymbol{y}'\boldsymbol{X}\left(\frac{Tr(\boldsymbol{XX}')}{T}\boldsymbol{I}+\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\boldsymbol{X}'\boldsymbol{y}\right)
\end{aligned} \tag{2.3.8}$$

Details are provided in appendix.

The likelihood found is only the conditional likelihood in Bayes' theorem. The focus is on the posterior likelihood. To find this the prior distribution is used,

$$P(\omega_0,K) \tag{2.3.9}$$

If no information of the prior likelihoods exist, they will simply be set equal, thus ignoring it, and this will be done in this project.

The likelihoods found can be compared with the likelihood of pure noise. This likelihood is quite easily found by removing all sinusoids from the signal model by setting $\boldsymbol{X}=\boldsymbol{0}$. The equation then become,

$$\log P(\boldsymbol{y}\mid\omega_0,K)=\frac{3}{2}\log\sigma_y^2-\frac{3+T}{2}\log(T+1)\sigma_y^2 \tag{2.3.10}$$

Details are provided in appendix.

2.3.1 Advantages & disadvantages of the Bayesian pitch detector

This algorithm is very versatile and can be used in a lot of scenarios. Besides the pitch it can find the number of harmonics present in the signal and it can be used to search for frequencies that are not harmonically related.

The algorithm is not dependent on the window in the same way as when running in the frequency domain, but some dependency still exists. The lobe is still dependent on

the length of the signal, but the width is narrower than with the FFT. This can be a problem for the algorithm. If the lobe gets too narrow and the search grid is not fine enough the peak can be missed. This is not a problem with the width of the lobe here as can be seen in the plots and from the fact that 1 Hz resolution is used.

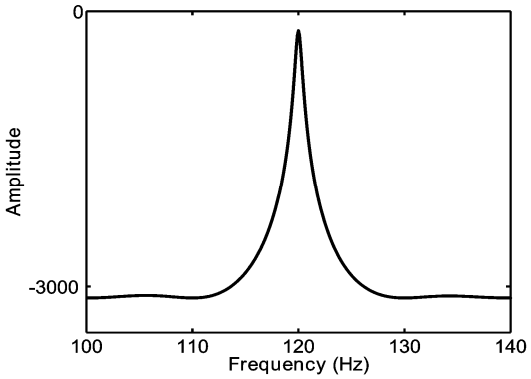


Figure 2.3.1: A very distinct frequency can be observed at 120 Hz.

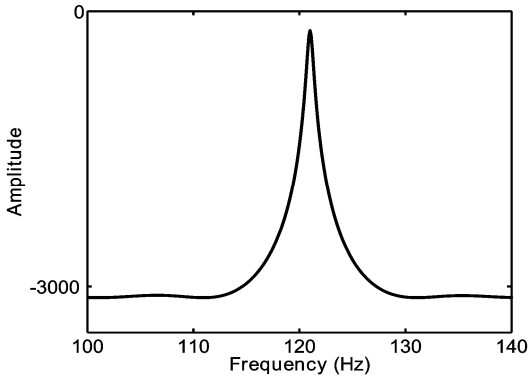


Figure 2.3.2: Here it is at 121 Hz.

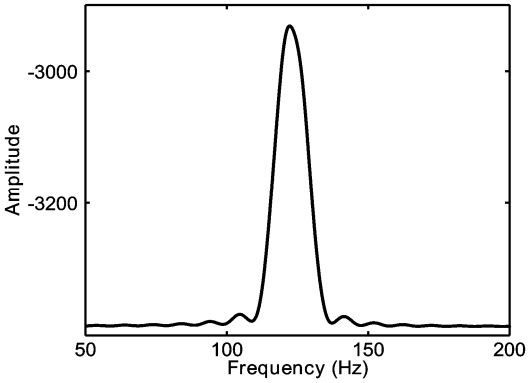


Figure 2.3.3: The separation is not as good as the plots above might suggest. Here frequencies at 120 and 126 Hz.

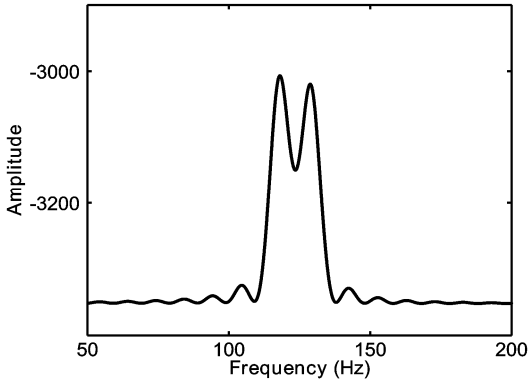


Figure 2.3.4: Here frequencies at 120 and 127 Hz can be separated, but they lie at 118 and 129 Hz.

Because the main lobe is very narrow one would assume that the separation is much better than with the FFT. The separation is better, but not as much as expected. The plots are found when modelling only a single frequency. If two frequencies are modelled, the accuracy can be increased dramatically, but this is not the relevant case here. These plots are included to show the influence of frequencies contributed by noise. From the plots it can also be seen that the certainty of a frequency is attenuated a lot when another frequency is present. This could suggest that the algorithm is sensitive to noise, but this will show up in the experiments.
A disadvantage of the algorithm is the speed, because the probability contains many calculations. This can also be seen in the experiments.

2.4 HMUSIC pitch detector

The HMUSIC [Christensen, 2004] algorithm is a development of the MUSIC [Schmidt, 1986] algorithm. It works in the time domain and uses a complex model of the signal. It makes use of the signal's covariance matrix and divides the feature space into a signal and a noise subspace. This is done with an eigenvector decomposition of the covariance matrix and it is assumed that the eigenvectors with the biggest eigenvalues are the vectors containing the harmonics, and the rest are characterized as noise. When the subspaces have been found it is quite easy to project the model onto them. When the right model is projected on to the noise space the values should be very small as they should be conjugates. A measure using the inverse of the projection is used and the model frequency with the highest score is the pitch.

2.4.1 MUSIC

The MUSIC algorithm uses an array of sensors and was originally designed to find the direction of arrival of multiple signals. The sensor array consists of M sensors and the algorithm is limited to find M or less signals. If the signals are only coming from a single direction and the actual direction is not important, a single sensor can be used. The M samples from the M sensors are synthesized by using M samples in serial from one sensor. This is the same as if the distance between the sensors infers a delay of exactly one sample period and the propagation between the sensors is negligible. If we are looking for L frequencies,

$$L \le M \tag{2.4.1}$$

A complex model of the signal, y, is the basis of the algorithm,

$$y(n) = \sum_{l=1}^{L} A_l e^{j(\omega_l n + \phi_l)} + e(n) \tag{2.4.2}$$

A single measurement of the synthesized M sensors are given by,

$$\tilde{\boldsymbol{y}}(n) = \begin{bmatrix} y(n) & y(n-1) & \cdots & y(n-(M-1)) \end{bmatrix}^T \tag{2.4.3}$$

A single frequency can be split up in a signal part and a part with the delay between samples,

$$y_l(n-m) = A_l e^{j(\omega_l(n-m)+\phi_l)} + e = A_l e^{j(\omega_l n+\phi_l)} e^{-j\omega_l m} + e \tag{2.4.4}$$

The model of the signal can then be written as,

$$\tilde{\boldsymbol{y}}(n) = \boldsymbol{X}\boldsymbol{f} + \boldsymbol{e} \tag{2.4.5}$$

where $\boldsymbol{X}$ describes the delays between the samples and $\boldsymbol{f}$ is the signal. They are given by,

$$\begin{aligned} \boldsymbol{X} &= \begin{bmatrix} 1 & 1 & \cdots & 1 \\ e^{-j\omega_1} & e^{-j\omega_2} & \cdots & e^{-j\omega_L} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-j\omega_1(M-1)} & e^{-j\omega_2(M-1)} & \cdots & e^{-j\omega_L(M-1)} \end{bmatrix} \\ \boldsymbol{f} &= \begin{bmatrix} A_1 e^{j(\omega_1 n+\phi_1)} & \cdots & A_L e^{j(\omega_L n+\phi_L)} \end{bmatrix}^T \end{aligned} \tag{2.4.6}$$

If the noise, $\boldsymbol{e}$, and the signal is assumed uncorrelated the covariance of $\tilde{\boldsymbol{y}}$ can be found as,

$$\begin{aligned}\boldsymbol{R}^{M\times M} &= E\left\{\tilde{\boldsymbol{y}}(n)\,\tilde{\boldsymbol{y}}^T(n)\right\}\\ &= E\left\{\left(\boldsymbol{X}\boldsymbol{f}+\boldsymbol{e}\right)\left(\boldsymbol{X}\boldsymbol{f}+\boldsymbol{e}\right)^T\right\}\\ &= \boldsymbol{X}E\left\{\boldsymbol{f}\boldsymbol{f}^T\right\}\boldsymbol{X}^T+\sigma^2\boldsymbol{I}\\ &= \boldsymbol{X}\boldsymbol{A}\boldsymbol{X}^T+\sigma^2\mathbf{I}\end{aligned} \tag{2.4.7}$$

$\boldsymbol{A}$ is a diagonal matrix containing the squared amplitudes and σ^2 is the variance of the noise [Pedersen, 2003, chap. 3].

It is assumed that there exist more sensors than signals and thus the $\boldsymbol{X}\boldsymbol{A}\boldsymbol{X}^T$ matrix will be singular and have L positive eigenvalues and $M-L$ eigenvalues will be 0. When adding a scaled unity matrix to another matrix, the eigenvectors are not changed and the eigenvalues are all changed by addition of the scale[1]. This means that the covariance $\boldsymbol{R}$ will have $M-L$ eigenvalues equal to the noise variance and L eigenvalues that are bigger than the noise variance,

$$\lambda_j = \sigma^2 \;\wedge\; \lambda_i > \lambda_j \;\;,\;\; i=\{1,2,\ldots,L\} \;\;,\;\; j=\{L+1,L+2,\ldots,M\} \tag{2.4.8}$$

This means that the subspace spanned by the signal can be found as the eigenvectors with the L biggest eigenvalues and the noise subspace can be found as the $M-L$ eigenvectors with the smallest eigenvalues.

Because all eigenvectors are orthogonal the noise subspace is orthogonal to the signal subspace. If the signal is projected on to the noise subspace the projection will be 0. This means that the model that minimizes the projection onto the noise subspace is the true model.

2.4.2 HMUSIC

In the above algorithm there were no restrictions in the selection of the frequencies. The pitch is only relevant in real signals. In HMUSIC the frequencies are the fundamental one and the harmonics and both positive and negative frequencies must be included,

$$\begin{aligned}\omega_i &= i\omega_0 \quad,\quad i=\left\{1,2,\ldots,\frac{L}{2}\right\}\\ \omega_i &= -i\omega_0 \quad,\quad i=\left\{\frac{L}{2}+1,\frac{L}{2}+2,\ldots,L\right\}\end{aligned} \tag{2.4.9}$$

If $\boldsymbol{G}$ is defined as a matrix containing the $M-L$ eigenvectors with the smallest eigenvalues, the pitch search can be defined as follows,

$$\arg\min_{\omega_0}\left\|\boldsymbol{X}^T(\omega_0)\,\boldsymbol{G}\right\|_F \tag{2.4.10}$$

where $\|\bullet\|$ is the Frobenius norm.

Note that the equations are independent of the amplitudes and phases of the signal frequencies.

The equations above were based on the covariance matrix of the measured signal. This is not available and must be approximated. This is done in the usual manner,

$$\hat{\boldsymbol{R}} = \frac{1}{N-M}\sum_{n=M}^{N}\hat{\boldsymbol{x}}(n)\,\hat{\boldsymbol{x}}^T(n) \tag{2.4.11}$$

---

[1] $\left(\boldsymbol{A}+\sigma^2\mathbf{I}\right)\boldsymbol{v} = \boldsymbol{A}\boldsymbol{v}+\sigma^2\boldsymbol{v} = \lambda\boldsymbol{v}+\sigma^2\boldsymbol{v} = \left(\lambda+\sigma^2\right)\boldsymbol{v}$

where N is the number of samples. The approximation has the consequence that the projection will no longer be exactly 0. The harmonic pseudo spectrum is defined as follows,

$$P(\omega_0) = \frac{LM(M-L)}{\left\| \boldsymbol{X}^T(\omega_0)\boldsymbol{G} \right\|_F^2} \tag{2.4.12}$$

and the pitch is now found by maximizing this value,

$$\arg\max_{\omega_0} P(\omega_0) \tag{2.4.13}$$

In summary a noise subspace is identified by using the eigenvalue decomposition of the covariance of the signal. Then the model is projected onto the noise subspace and the inverse is used to form a pseudo spectrum. The pseudo spectrum is calculated for all relevant frequencies and the maximum is selected as the pitch.

2.4.3 Advantages & disadvantages of the HMUSIC algorithm

This algorithm assumes that the noise is white. This is seldom the case. Based on this assumption the biggest eigenvalues are said to come from the speech structure. This is true for the white noise case, but if a frequency from a noise source is bigger than one of the harmonics, this frequency will be put in the speech domain and the harmonic will be put in the noise domain. This is of course a problem because when the noise and speech domains have been selected it says nothing about the importance of each of the feature vectors. This means the vector coming from the noise is as important as any of the other vectors. This disadvantage will not show in the synthetic data since white noise is used, but it can occur when running on real data. The Bayesian approach is also based on the assumption of white noise, but it does not divide into noise and speech domains on this assumption.
For the comparison of the other two algorithms the same plot of two close frequencies has been made for this algorithm.

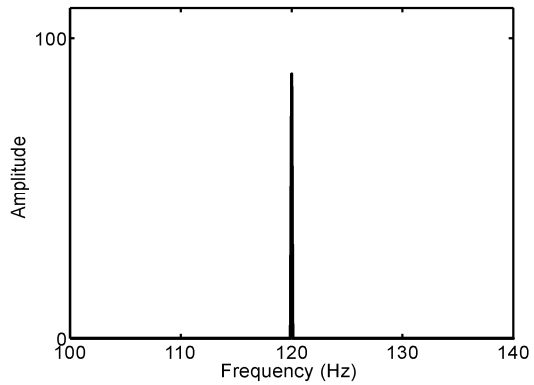


Figure 2.4.1: The pitch spectrum is very clear with a single frequency. Here at 120 Hz.

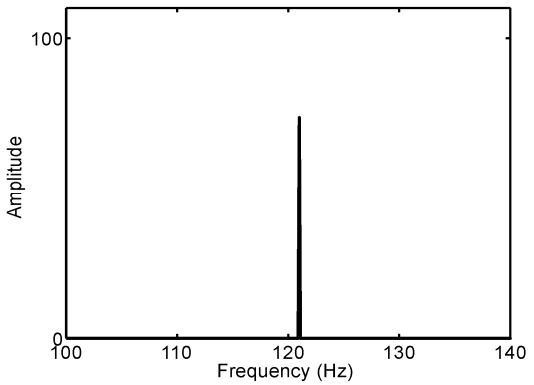


Figure 2.4.2: Here a frequency at 121 Hz.

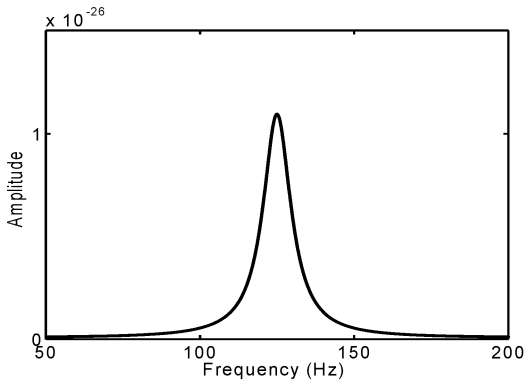


Figure 2.4.3: The separation of two frequencies is not good. This signal consists of frequencies at 100 and 150 Hz. Notice the amplitude.

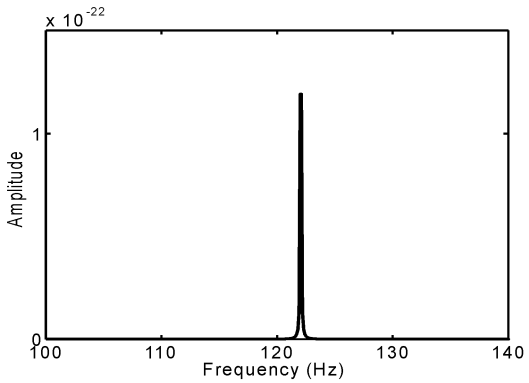


Figure 2.4.4: frequency of 122.05 Hz searched for in 0.1 Hz steps. If the frequency is not hit directly the value is attenuated greatly.

The real pitch gets a much higher value than other frequencies and the lobe width is close to 0 Hz. Figure 2.4.4 shows that the frequencies are not separated although they are 50 Hz apart. This indicates that the algorithm will have problems when the noise is not white. Another problem with this algorithm is indicated in figure 2.4.4. Here the frequency is not hit directly, but with a deviation of only 0.05 Hz. The pseudo spectrum is attenuated by 10^{-22} which is quite extreme. These experiments are run on synthetic data without noise and this is partly the reason for the extreme difference, but it might still be a problem for real data. The true performance must be shown in the experiments.

The algorithm is quite slow mostly due to the calculation of the covariance.

2.5 Reference data set and parameters

In the preceding sections three different pitch detector candidates have been reviewed. Only one will be used and two must be discarded. To evaluate the algorithms it is important to realize the important aspects of the pitch detection with classification in mind. It means that the pitch detectors should be compared on their ability to work in a classification system, and this is not necessarily just about accuracy. In the ideal world, the classification step would already have been done so that it could be clear what the requirements are. Even more ideally the process would change back and forth, finding a pitch detector for a classifier, and finding a classifier for the pitch detector. This is not possible in this case and not in most cases. Some general aspects, of what is important and what is not, will be investigated in the first subsection, and the comparison parameters will be found. In the second subsection the reference data set will be presented.

2.5.1 Comparison parameters

Many papers [Christensen, 2004], [Schmidt, 1986], [Pedersen, 2003, chap. 2] considers the precision of the detectors and measures it against the so called Cramer-Rao bound of the optimal solution. When pursuing this bound they often neglect the problem of doublings and halvings because they only search in a very narrow band around the true pitch. The narrow band is used both for computational reasons as many algorithms perform a gridsearch for the pitch and sometimes intentionally to avoid the problems of doublings and halvings. For human voices, music and noises it is not the exact pitch that separates the classes. Music played out of tune is still considered music and in speech and noise you would not notice a difference of 0.01 Hz in the pitch. The precision in this project is set to 1 Hz. This means that it is of no interest to reach the Cramer-Rao bound of precision.
A very common form of error functions are the mean- and sum-square-errors. They have the disadvantage of putting a lot of weight on big errors. A very good performing set can be destroyed by a single outlier. As neither the smallest errors nor the biggest errors are the most interesting a 0-1 miss rate error function is used. A miss is when a measured pitch is outside an interval around the true pitch and gets penalized with 1. A hit is when the measured pitch is within the interval and this is not penalized, 0. The accuracy of hits is calculated, and mean-square-error and mean-error is included for reference.

Miss rate

The miss rate, E_{miss}, is a measure of big errors or misses. The measure is incremented by one each time a window has a pitch error bigger than a predefined constant, e_M.

$$E_{miss} = \frac{1}{W}\sum_{i=1}^{W} e_{m,i} \qquad e_{m,i} = \begin{cases} 0 \ , & e_{p,i} \le e_M \\ 1 \ , & e_{p,i} > e_M \end{cases} \qquad (2.5.1)$$
$$e_{p,i} = \left| p_i - \hat{p}_i \right|$$

with W being the number of windows, p_i being the real pitch and $\hat{p}_i$ being the measured pitch of the i'th window and e_M is the constant deciding the interval around the true pitch. e_M was set to 10 during the comparisons. The miss rate will be between one for a completely erroneous result and zero for the perfect result.

Accuracy error

The accuracy error, E_{acc}, is a measure of how accurate the pitch is determined on a hit. A hit means it is closer than e_M.

$$E_{acc} = \frac{1}{\sum_{i=1}^{W} 1 - e_{m,i}} \sum_{i=1}^{W} \left(1 - e_{m,i}\right) e_{p,i} \tag{2.5.2}$$

The accuracy error simply measures the mean distance from the true pitch when a hit occurs. It should give an impression of the accuracy of the algorithms without being punished for outliers.

Mean (square) error

For the sake of comparison the mean-square-error and the mean-error have been included as well. They are given by,

$$\begin{aligned} E_{MSE} &= \frac{1}{W} \sum_{i=1}^{W} e_{p,i}^2 \\ E_{ME} &= \frac{1}{W} \sum_{i=1}^{W} e_{p,i} \end{aligned} \tag{2.5.3}$$

Algorithm complexity

Also the complexity of the algorithm is a very important characteristic. This project was initiated with hearing aids in mind, and for hearing aids computational burden is a very limiting factor. The complexity of the algorithms is difficult to evaluate, but instead the time consumed by the algorithms is measured. This is a very easy parameter to measure and it gives a good indication of the complexity. Of course the time consumed has a lot to do with how well the code has been optimized and if it is run in Matlab or in C. All code has been optimized to a first level, for example by avoiding for-loops where possible, but the measured times should be used carefully.

2.5.2 Reference data set

A reference pitch must be available in order to measure the parameters found in the previous section. A reference pitch is not available for most sound clips. Different databases do exist though, where a reliable reference pitch is available. Two of these will be used and they are both based on laryngograph readings which are explained next. Further more synthetic data will be generated. This has the advantage that there is full control of the data and it gives a very exact reference to compare against.

Laryngograph databases

The electro laryngograph works by putting a set of electrodes on the neck and measuring the electrical response. The laryngograph reading results in graphs as shown in figure 2.5.1.

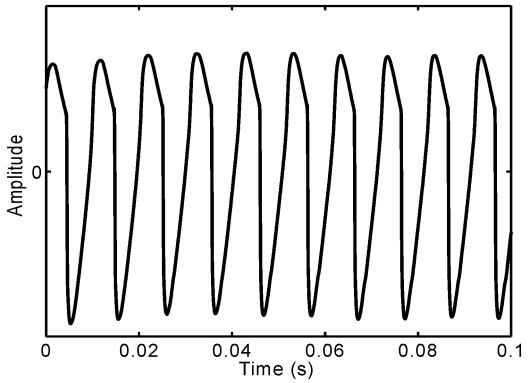


Figure 2.5.1: 100 ms of the laryngograph signal.

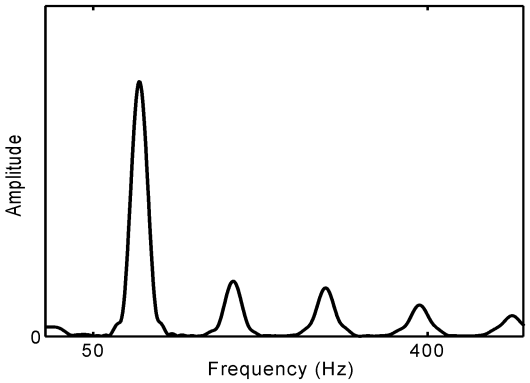


Figure 2.5.2: Spectrum of the signal to the left. The fundamental frequency is very clear.

In figure 2.5.1 a frequency around 100 Hz can be observed. A problem with the laryngograph readings is that even with this available the pitch still has to be detected. The FFT has been chosen for this, based on the accuracy and the ease. The largest peak is simply extracted from the relevant frequency range. The spectrum of the signal in figure 2.5.1 is shown in figure 2.5.2.
The procedure is very similar to what could be used for a pitch tracking algorithm, but there are some differences when using the laryngograph data. There is less noise because the measurements are taken directly from the vocal cords. The other difference is that the fundamental frequency is very clear in the laryngograph where as it can be nonexistent in other plots. This difference is what makes the data worth while. The two databases used are the Keele Pitch Database [23] and CSTR US KED Timit database [12].

Manual check

After the automatic detection of the pitch using the laryngograph data, the found pitch is plotted in a spectrogram plot of the speech to check if the found pitch is correct. The problem is that speech is not voiced all the time and thus the pitch is not valid all the time. A spectrogram clearly shows the pitch structure in the data. Figure 3 shows the way to find voiced and unvoiced parts and to check the found pitch. It limits the amount of data, because it has to be checked manually. The first 50 clips from the Timit database and the complete Keele pitch are used totalling a little over 7 minutes of speech.

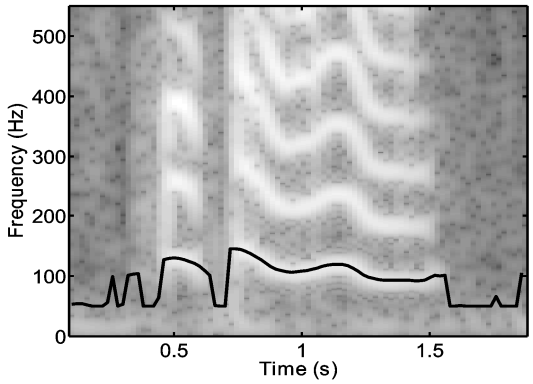


Figure 2.5.3: Spectrum of speech signal with found pitch plotted on top. Two regions of pitched signal are easily observed.

Synthetic data

To get more control over the performance measure, a synthetic model of voiced speech has been created. This model creates a sound file with variable pitch, envelope and signal-to-noise ratio that enables a very distinct test set. It also gives the exact pitch used to generate the data. By using this synthesizer one is not limited by the available sound files and any parts of the algorithms can be stressed. This is a big advantage. The generated signal is not a natural one and the model is of course simpler than reality so all aspects cannot be covered, but together with the real set a very good comparison of the different algorithms can be made. The reference pitch for this data is more accurate than the data from the laryngograph because it is certain that the pitch is actually present in the data. With the laryngograph data one can have a pitch when measuring at the throat, but because of the delay and the modulation in the mouth, it may be gone or hidden in the speech.
Different test sets has been created with the following characteristics.

Pitch

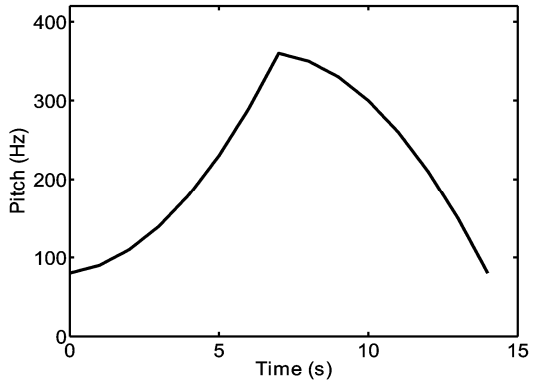


Figure 2.5.4: Pitch 1. Different slope values both positive and negative.

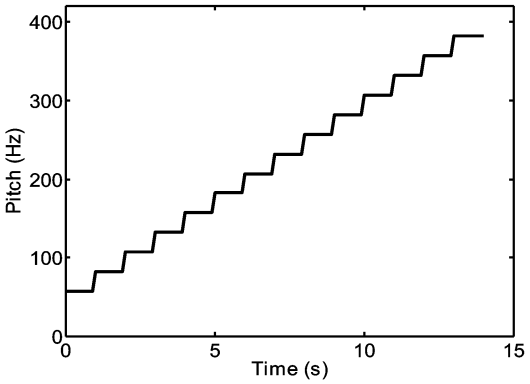


Figure 2.5.5: Pitch 2. Constant pitch in steps over the complete frequency range.

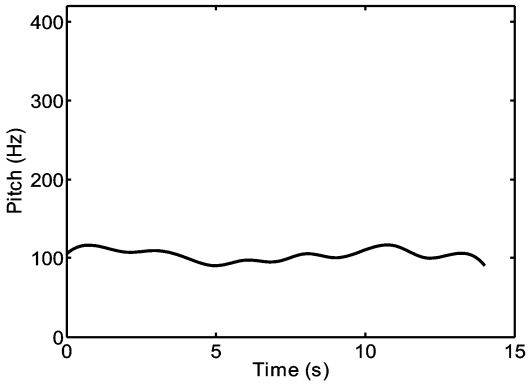


Figure 2.5.6: Pitch 3. Model of male voice.

Clips of 14 s are generated. Pitch 1 tests different steepness values. Pitch 2 has a constant pitch and covers the complete range and pitch 3 resembles a natural pitch of a male voice.

Envelope

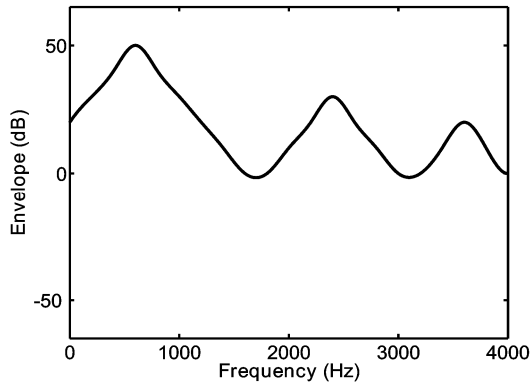


Figure 2.5.7: Envelope 1. Model of true speech.

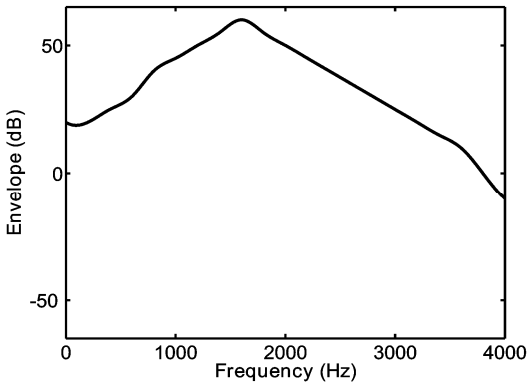


Figure 2.5.8: Envelope 2. A single peak at high frequency.

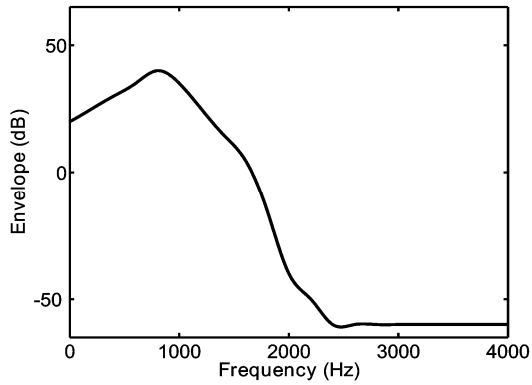


Figure 2.5.9: Envelope 3. A lowpass filtered signal.

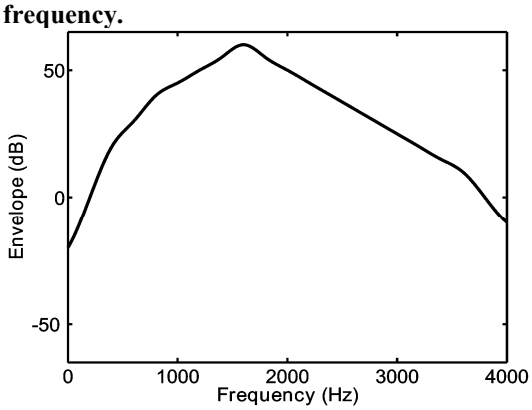


Figure 2.5.10: Envelope 4. A highpass filtered signal. Fundamental is not present.

Envelope 1 resembles an envelope in clear speech. Envelope 2 has a single large peak. Envelope 3 is a lowpass band limited signal and envelope 4 has the fundamental frequency cut out.

Noise

Besides the pitch and the envelope the amount of noise can be controlled as well. 4 variations with infinite, 20, 10 and 0 dB signal to noise ratio are used.

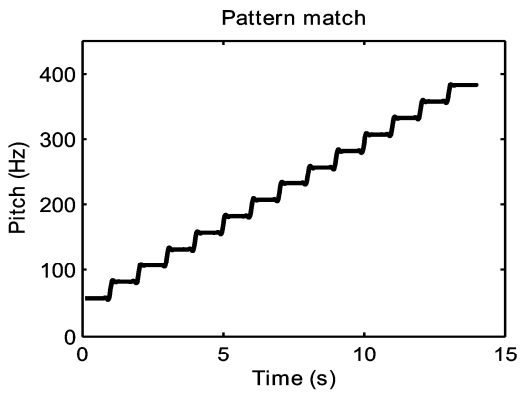
2.6 Comparison and choice of pitch detector

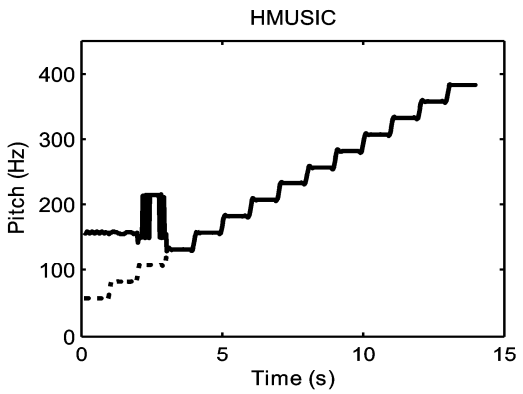
The algorithms have been run on the pitch databases and on the 48 synthetic combinations generated by combining the 3 pitch, 4 envelope and 4 noise patterns. Two of the algorithms, HMUSIC and Bayes, are dependent on the number of harmonics modelled, and can be run with different numbers. The time is measured for doing all of the clips and then divided by the length of the clips to get a measure of how long time it takes to find the pitch of 1s of sound. First it was run with 5 harmonics.

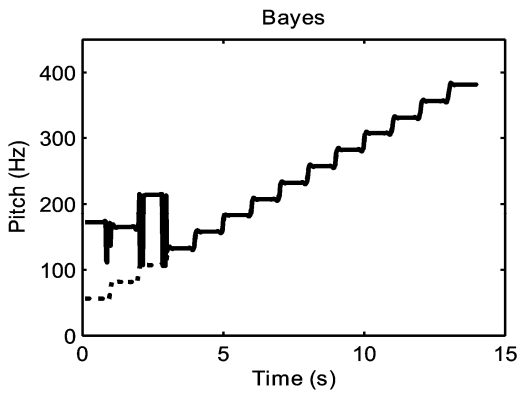
5 harmonics

	Pattern match	HMUSIC	Bayes
Time per s. / s	0.40	26.7	9.23
Miss rate	0.20	0.58	0.46
Accuracy error / Hz	0.64	0.97	1.04
MSE / Hz^2	4722	16396	15091
ME / Hz	26.5	83.8	76.4

The timings, if considered by themselves, clearly favourize the combined algorithm. It is 20 times faster than the Bayes and more than 60 times faster than HMUSIC.
The errors are quite big. The best miss rate is 20 % misses which is not good if it were used in a classification system. The high miss rate is not the result of poor algorithms though, but rather because the samples they are used on are chosen so to stress the algorithms. Again the combined pattern match algorithm is clearly the winner.
An example of the algorithms is shown below to get an idea of the problems. This is with 5 harmonics modelled, pitch 2, envelope 1 and no noise.







	Missrate	Accuracy
Pattern match	0.00	0.17
HMUSIC	0.21	0.35
Bayes	0.19	0.45

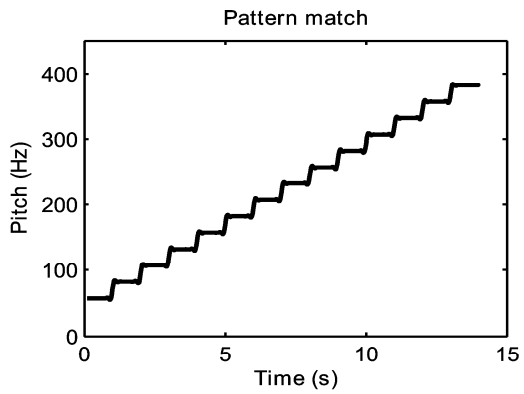
Figure 2.6.1: Synthetic signal with pitch 2, envelope 1 and no noise. 5 harmonics are modelled in Bayes and HMUSIC. They both show problems in the lower frequencies, where doublings and even triple the frequency occur. HMUSIC misses in the first two steps without doubling.

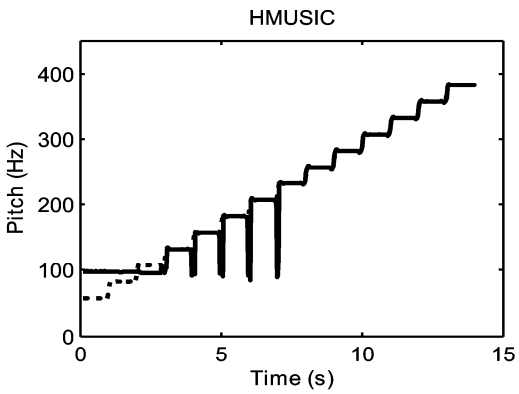
The plots reveal the low frequencies to be where HMUSIC and Bayes differ with the combined pattern match. When the pitch is low a lot more than 5 harmonics are present and the biggest harmonics is not included in the 5 first. A run with 10 harmonics is done to see if it helps.

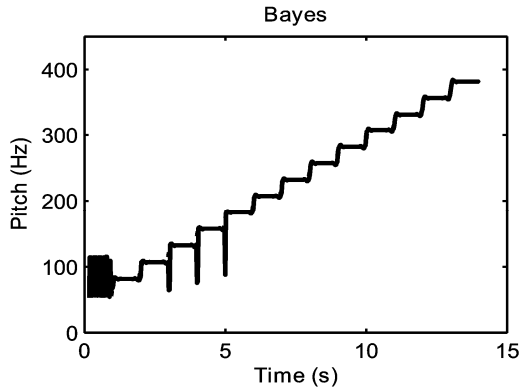
10 harmonics

	Pattern match	HMUSIC	Bayes
Time per s. / s	0.42	49.3	23.5
Miss rate	0.20	0.52	0.30
Accuracy error / Hz	0.64	0.75	0.69
MSE / Hz^2	4722	5706	4456
ME / Hz	26.5	47.3	34.7

Only HMUSIC and Bayes have different results because they are the only algorithms that are dependent on the number of harmonics in the model. For both algorithms the time close to doubles and they were already slow. The performance is improved though and especially the Bayes algorithm improves quite a lot being the best algorithm measured in MSE. The pattern match is still the best in the remaining errors and now it is more than 110 and 50 times faster than HMUSIC and Bayes respectfully. A plot of the same sound clip as before is shown with 10 harmonics.







	Missrate	Accuracy
Pattern match	0.00	0.17
HMUSIC	0.23	0.44
Bayes	0.03	0.42

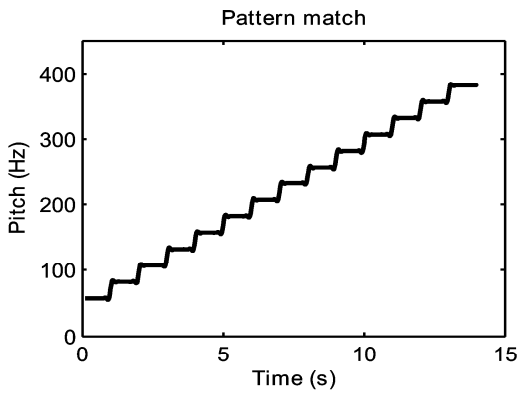
Figure 2.6.2: Synthetic signal with pitch 2, envelope 1 and no noise. 5 harmonics are modelled in Bayes and HMUSIC. Bayes has improved somewhat. HMUSIC misses in the first three steps without doubling. Pattern match still out performs both.

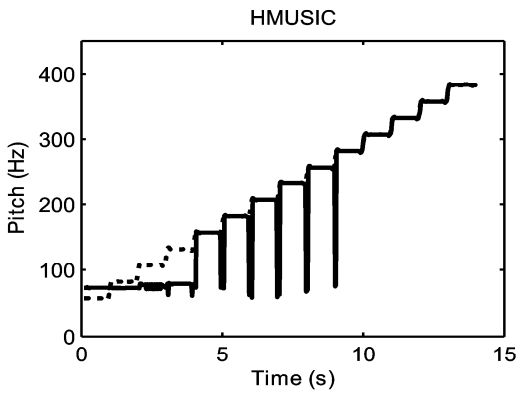
In the example the Bayes algorithm has improved a lot, but the HMUSIC actually performs worse. It seems to have trouble with transitions in the pitch. A clear choice of pitch detector seems to appear, but a run with 15 harmonics is run to be sure.

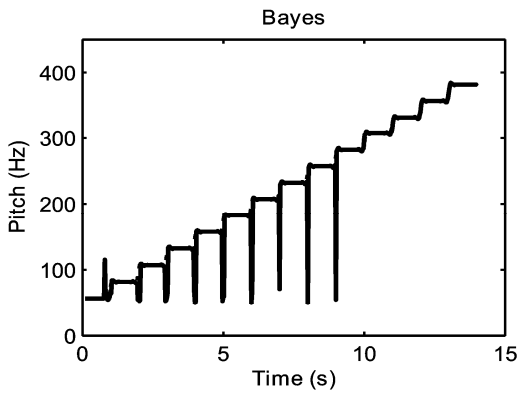
15 harmonics

	Pattern match	HMUSIC	Bayes
Time per s. / s	0.46	64.2	46.1
Miss rate	0.20	0.62	0.25
Accuracy error / Hz	0.64	0.89	0.58
MSE / Hz^2	4722	5351	3624
ME / Hz	26.5	46.6	28.3

The Bayes algorithm comes closer and closer to the performance of the combined pattern match algorithm and again it has the best MSE. It is possible that with a better optimized implementation that it could be the chosen algorithm. The times are though very big, a 100 times longer for the Bayes algorithm and more for the HMUSIC algorithm.







	Missrate	Accuracy
Pattern match	0.00	0.17
HMUSIC	0.26	1.10
Bayes	0.04	0.30

Figure 2.6.3: Synthetic signal with pitch 2, envelope 1 and no noise. 5 harmonics are modelled in Bayes and HMUSIC. Bayes shows no doubling now. HMUSIC seems to have a problem with low frequencies that is not related to the usual doubling. Pattern is still the best.

Choice of algorithm

The maximum possible number of harmonics is actually 100 for a pitch at 50 Hz and a sampling frequency of 10 kHz. In principle the model should take this into account. Even though there was a performance gain when trying more harmonics, the time consumption is simply too high and the performance gain too small. Actually the HMUSIC and the Bayes algorithm should ideally not only model a single number of harmonics, but should perform a complete search of possible harmonics. This is completely out of the picture as it would simply consume too much time.

More plots are included in appendix and from them it can be seen that it does not help to increase the modelled harmonics for all signals. It depends on the number of harmonics that is in the signal. This gives a dependency between the detector and the signal which is not desirable. This problem would probably be solved by modelling more than one number of harmonics and taking the best or the average, but as mentioned above and indicated in the runtimes it would take too much time. It does not mean however that the algorithms are inferior to the pattern match algorithm in general. The HMUSIC and Bayes potentially have a much better resolution than the pattern match algorithm. This kind of resolution is however not necessary in regards to the classification task, and the time aspect is quite important. If the algorithms are too slow it simply takes too long to do the experiments. Also considering the hearing aid aspect the favour is on a less time consuming algorithm. This means that the pattern match algorithm is the one that will be used further on in the project.

3 Pitch based features

In the following the pitch is assumed to be known. The combined pattern match and HPS algorithm was the pitch detector best suited for classification. Therefore the search for features will be based on the pitch detected by this algorithm. The features found should, to some extent, be independent on the pitch detector used. If a more effective pitch detector is developed, that one can be used instead.
In this chapter the three classes - music, noise, and speech - will be described in terms of their pitch characteristics. These descriptions will be used to generate pitch features. It will be done in an exploratory manner by visually inspecting the plots, and by running small experiments on data. To avoid overlooking features because of bad results in the limited experiments, features will be included even though the experiments do not look too good. This means that although the feature might not look very promising at first, it will be presented any way. When a respectable number of features are found, experiments will be carried out on a larger database to evaluate the performance of the features. When running on a larger database the statistics of the features can be evaluated more precisely and vulnerabilities as well as strong points can be found. The database will be presented in the next chapter and the final selection of features is done in chapter 5.
Because the investigation is based on the pitch returned from the pitch detector, a dependency cannot be avoided, and some features might seem tailored only for this algorithm. The pitch detector gives the evaluated pitch as well as the error between the model and the signal. Both parameters can be used for classification. The error may seem as a detector specific parameter, but a similar parameter will be possible for most detectors, so it will not be difficult to convert the features extracted here to another pitch detector.
In the chapter, different levels of windows will be used. The window that the pitch was detected on, will be called the pitch window, and the window that the features are extracted on, will be called the feature window. A feature window will be divided into smaller windows called reliable windows. The reliable windows are explained later.

3.1 Description of the pitch

The search will start by plotting typical examples for each class. The pitch detector returns two values, the detected pitch and the error between the model and the signal. First the pitch will be investigated.

3.1.1 Description of signal from pitch detection

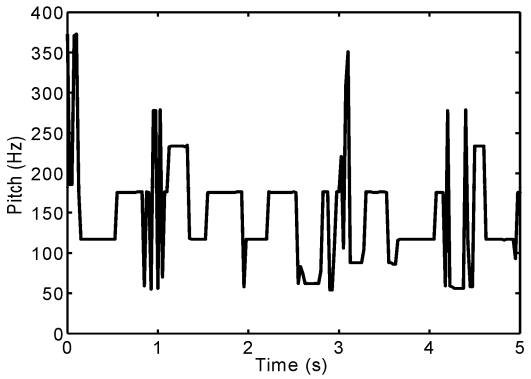


Figure 3.1.1: Pitch signal of music. The steps and the constant behaviour in between the steps are characteristic of music.

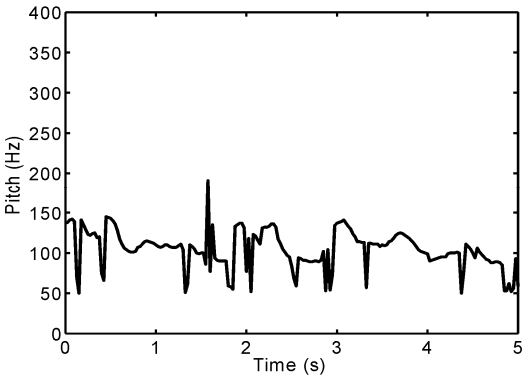


Figure 3.1.2: Pitch signal of speech. The slopy behaviour in all parts is characteristic of speech.

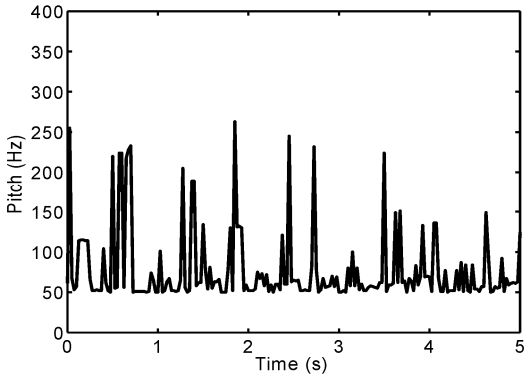


Figure 3.1.3: Pitch signal of noise. The low bottom and in general the random behaviour is characteristic of noise.

The first impression is that all classes show a lot of noise. This is due to two things. First of all a pitch is not present at all times in any of the classes. Speech has the unvoiced parts with no pitch. Music can have silent parts or parts containing only percussion instruments like a drum solo. Many kinds of noises are characterized by not having a pitch. The algorithm presents a pitch no matter if one is present or not. Secondly, even though the best pitch detector was selected, it can make mistakes. A division into pitched parts and unpitched parts would be useful. This is done with reliable windows.

In the music plot the pitch seems to confine itself to certain steps. This is of course not surprising as music are played on notes and sounds terrible if some are off key. The question is if all music is tuned to the same frequencies or if it is only tuned within a piece. General features could relate to the constant nature of the pitch. Features could capture if the frequency is close to a note or if the frequencies are related to each other in a musical way.

The speech is showing slopes in the pitch. It is not constant as the music. When people speak they use the tonality to express emotions and to put weight on some

words. This means that speech does not follow the same constant behaviour relating to tones as music does.
It is a little harder to describe the noise. The plot shows many spikes and no apparent pattern in between the spikes. It seems that the noise tends to give small values of the pitch in between the spikes. This is a pitch detector dependent characteristic. When no pitch is present in a signal it has a very flat spectrum. If a completely flat spectrum is modelled with the pattern match algorithm, the small pitches are favoured because small pitches cover more of the frequency range than larger pitches does. This means that if no pitch is present a small pitch will be found in general. This can be used to divide false from true pitches.

3.1.2 Description of reliability of pitch signal

The reliability of the found pitch is based on the error of the pitch detection, E_ω. It is one minus the error normalized with the energy in the signal. For a real signal it would be the same as one minus the energy of the error divided by the energy of the signal. 'One minus' is used to give one for a perfect match and zero for a total miss. It is simply an error measure that is not dependent on the signal strength.

$$R = 1 - \frac{E_\omega}{\frac{1}{2}|S|^2} = 1 - \frac{|S - \hat{S}|^2}{|S|^2} \tag{3.1.1}$$

Examples of the reliability is plotted below.

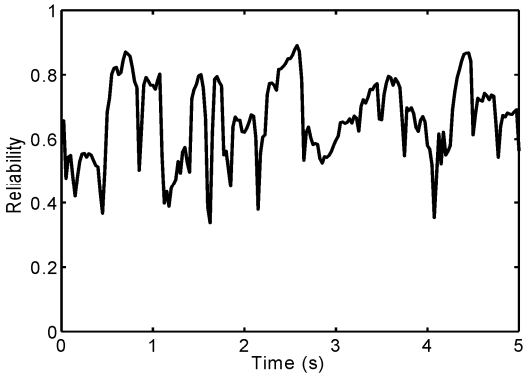


Figure 3.1.4: Reliability signal of music. Very 'clean'. Quite high mean.

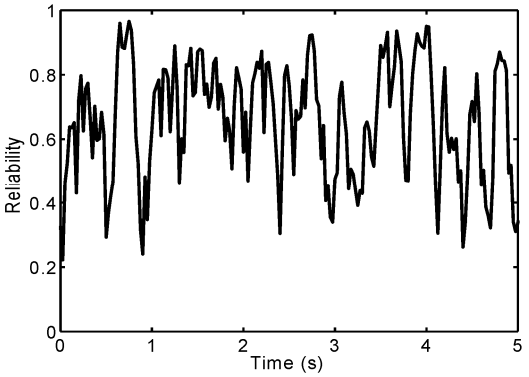


Figure 3.1.5: Reliability signal of speech. Seems more random, but still with a high mean.

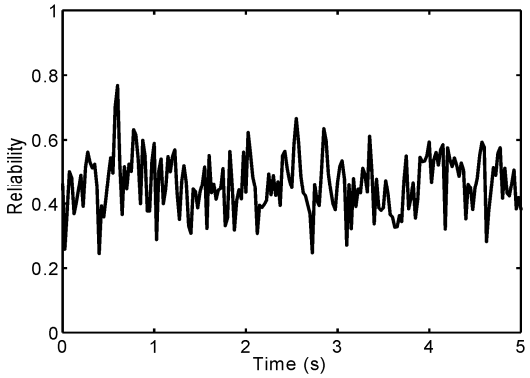


Figure 3.1.6: Reliability signal of noise. Low mean, random, but with smaller variance than speech.

The plots are quite different. The reliability of the music shows more constant behaviour than does the two other classes. This has something to do with the very constant nature of the tone and thus the pitch in music. Noise and speech varies a lot more giving a jagged impression of the plot. It seems the speech has a higher maximum value. This could be due to the fact that in clean speech only a single pitch is present whereas in music more than one is usually present and in the noise none is present. Noise clearly has a lower reliability than the other two classes.

3.2 Reliable windows

In sounds coming from physical objects it is very unlikely that the pitch changes rapidly. If the pitch is changing rapidly up and down in steps of more than 100 Hz it is very likely that it is not the true behaviour of the pitch which has been found, but rather some noise. In music the pitch can change from one tone to another, but it is not likely that the notes changes every pitch window that is separated by 25 ms. This means that when the pitch is relatively smooth over a number of pitch windows the measured pitch is very likely to be the true pitch. And when the pitch jumps a big number of Hz, it is probably because there is no longer a pitch present in the signal, or the pitch algorithm has detected the pitch wrongly, or a new tone is played on the instrument.

It has been noticed that noise often results in a pitch very close to the minimum allowed pitch. Because the lobe width is fixed for all pitches the minimum pitch is the one that covers most of the spectrum. This means that for a flat spectrum this will be the one with the smallest error. With a flat spectrum the pitch does not change very much from pitch window to pitch window and thus seems to be a reliable pitch. This is not the case of course.

To separate pitch windows, with a pitch that seems reliable, from other pitch windows, a term called reliable windows is used. It is simply time frames within the feature window where certain pitch properties are fulfilled.

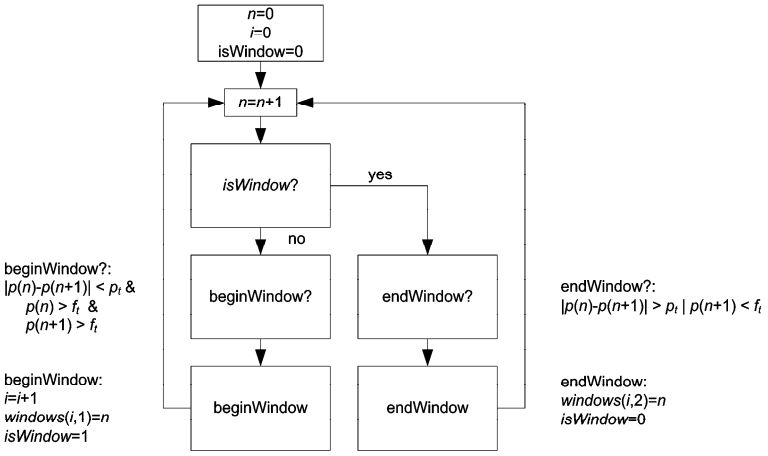


Figure 3.2.1: Algorithm for reliable windows.

Using the procedure above, the feature window is divided into reliable windows. The pitch values within a reliable window are characterized by two things; they are all greater than f_t, and no difference between two adjacent pitchvalues is greater than p_t. The output of the algorithm is a matrix, *windows*, with two rows and a number of columns. The first row has the start index and the second row contains the end index. The number of reliable windows is the number of columns. In the figure, p is the pitch sample, *isWindow* is a boolean variable, n is the pitch sample index, and i is the feature window index.

A vector, R, is generated with the value 0 or 1 for each pitch value specifying if the pitch is included in a reliable window or not. The reliable windows of the signals from figure 3.1.1, figure 3.1.2, and figure 3.1.3 is shown in figure 3.1.2, figure 3.1.3 and figure 3.1.4.

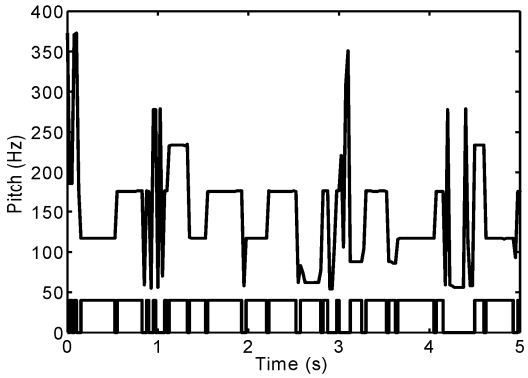


Figure 3.2.2: Reliable windows in the bottom of the plot of the pitch in music.

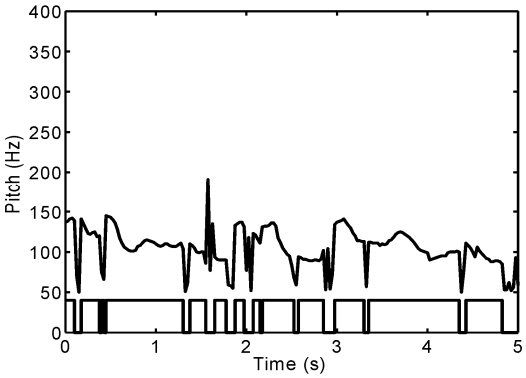


Figure 3.2.3: Reliable windows in the bottom of the plot of the pitch in speech.

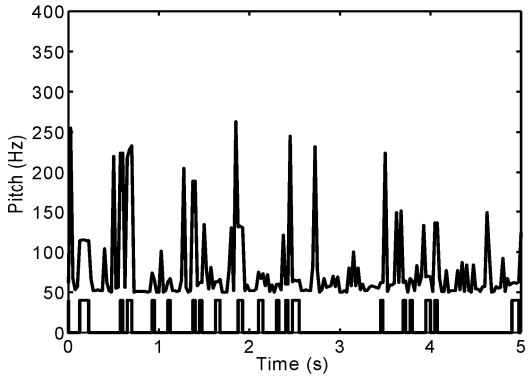


Figure 3.2.4: Reliable windows in the bottom of the plot of the pitch in noise.

The reliable windows also describe the classes by itself. The length of the reliable windows and the number of samples included in a reliable window separates music and speech from noise. Also the behaviour of the pitch within a reliable window can be used. Features are presented next, some uses the reliable windows and some does not.

3.3 Features

In this section the different features of the pitch signal is presented. Whether a feature shows good separation of the classes or not will not be checked thoroughly, but of course that is what they are all designed to do. All the following features are found on 5 second long feature windows. From the pitch detector a pitch value for each 25 ms was returned. This means the features are based on 200 samples each. Feature windows will overlap 4 s. This means that the decision horizon is 5 s, and a decision can be made every second.
The logarithm is taken of some of the features because the distribution of the features then came closer to gaussian. This is the subject of section four. Plots of the feature histograms on the sound database are in appendix.

In the following, I is the number of pitch values in a feature window and i is used to index them. When reliable windows are used, W is number of reliable windows in the feature window and w is used to index them. $pitch_i$ is the i'th pitch value and $reliability_i$ is the i'th reliability value.

3.3.1 Sum of reliable windows

The minimum length of a reliable window is two pitch samples. Because the difference between the two pitch samples is limited to p_t it means that very few reliable windows exist in a random pitch pattern. The property that a flat spectrum gives small pitches and the minimum pitch f_t, creates even fever reliable windows for noisy data. A good measure for identifying pitched signals is the sum of pitches included in reliable windows,

$$sumOfReliableWindows = \sum_{i=1}^{I} R_i \tag{3.2.1}$$

This feature is quite good at separating noise from music and speech.

3.3.2 Length of reliable windows

With noisy data, not only are the reliable windows few they also have quite short duration. This means the length of the reliable windows can give some information of the signal. The individual lengths are found and different features are calculated on this.
The maximum length and the mean length are used. The minimum length is not of much value, because small reliable windows will be present in almost all signals.

$$\begin{aligned} l_w &= windows(w,2) - windows(w,1) + 1 \\ maxWindowLength &= \log\max(l_w) \\ averageWindowLength &= \log\frac{1}{W}\sum_{w=1}^{W} l_w \end{aligned} \tag{3.2.2}$$

3.3.3 Deviation within reliable windows

For separating music from speech a valuable observation can be made from the plots above. The music exhibits a very constant pitch within the reliable windows whereas the speech can change quite much within a window. Thus the difference between maximum and minimum pitch within a reliable window can be used.

The deviation is calculated for each reliable window in the feature window and two features are calculated. The mean and the maximum of the deviations,

$$\begin{aligned} \boldsymbol{p}_w &= \left[pitch_{windows(w,1)}, pitch_{windows(w,1)+1}, \ldots, pitch_{windows(w,2)} \right] \\ d_w &= \max\left(\boldsymbol{p}_w\right) - \min\left(\boldsymbol{p}_w\right) \\ maxDeviation &= \max\left(d_w\right) \\ averageDeviation &= \frac{1}{W}\sum_{w=1}^{W} d_w \end{aligned} \tag{3.2.3}$$

3.3.4 Reliability

The plots of the reliability showed some features worth capturing. Two sets of features have been created. One that only uses the values in the reliable windows and one that uses the entire feature window. The feature based on the reliable windows is simply the mean,

$$averageReliability = \frac{\sum_{i=1}^{I} reliability_i \cdot R_i}{\sum_{i=1}^{I} R_i} \tag{3.2.4}$$

For the feature based on the entire feature window both the mean and the standard deviation is calculated.

$$\begin{aligned} genericReliabilityMean &= \frac{1}{I}\sum_{i=1}^{I} reliability_i \\ genericReliabilityDev &= \log\sqrt{\frac{1}{I-1}\sum_{i=1}^{I}\left(reliability_i - genericReliabilityMean\right)^2} \end{aligned} \tag{3.2.5}$$

3.3.5 Tonality

The notes of classical western music are confined to a subset of frequencies. These frequencies are defined with the reference point of the middle A at 440 Hz. To find the frequency of the rest of the notes simply multiply or divide by $\sqrt[12]{2}$ [Jørgensen, 2003, chap. 2] for each half note up or down. Music is expected to have pitches closer to frequency of these notes than noise and speech. To capture this, the distance to the nearest note is calculated.

The distance between notes is bigger in the higher end of the scale. This means a higher error can be achieved at higher frequencies and in general that noise, which has low pitch, will have better values. To make up for this the distance is calculated in logarithmic space. This makes sense as the frequencies of the notes are exponentially related to each other. To convert a given pitch to a note scale the following function is used,

$$t_i = 12\log_2\left(\frac{pitch_i}{440}\right) \tag{3.2.6}$$

This scale gives integer values if the pitch hits a note and in between values when the pitch is off key. The pitch distance is simply calculated as the distance to the nearest note.

$$d_i = \left| t_i - round\left(t_i\right) \right| \tag{3.2.7}$$

The average of the distances that are in a reliable window is used as a feature,

$$toneDistance = \frac{1}{\sum_i R_i} \sum_i d_i \cdot R_i \tag{3.2.8}$$

and when the entire feature window is used,

$$genericToneDistance = \frac{1}{I} \sum_{i=1}^{I} d_i \tag{3.2.9}$$

Another feature can be found by remembering the nearest note. In most music only a subset of the notes in the possible range is used. Scales is very uncommon in most music, and occurs only infrequently in pieces where they do exist. Speech on the other hand touches more frequencies because the pitch slides up and down hitting many notes. A feature is the number of different notes hit.

$$\begin{aligned} t_i &= round\left(t_i\right) \\ numberOfTones &= length\left(unique\left(\left[t_1, t_2, ..., t_I\right]\right)\right) \end{aligned} \tag{3.2.10}$$

this also has an equivalent that is not dependent on the reliable windows,

$$genericNumberOfTones = length\left(unique\left(\left[t_1, t_2, ..., t_I\right]\right)\right) \tag{3.2.11}$$

Singing and instruments such as guitars and violins are not tuned using a clear reference, like a tuning fork. This means they are not necessarily tuned for 440 Hz. A feature is created that is not dependent on this fixed reference. By finding a mean pitch of each reliable window and calculating the tone distance between each consecutive reliable window,

$$\begin{aligned} d_w &= \left|t_{w+1} - t_w\right| \ , \quad w = \left[1, 2, ..., W-1\right] \\ toneHarmonicDistance &= \frac{1}{W-1} \sum_{w=1}^{W-1} \left|d_w - round\left(d_w\right)\right| \end{aligned} \tag{3.2.12}$$

where t_w is the mean of the tone in the w'th reliable window.

3.3.6 Monotonicity

This feature is used on both the reliability and the pitch signal. Both music and speech is characterized by being dynamic. This means that even though music tends to be constant within reliable windows a single note is seldom held over the complete feature window. This can be the case for noise. For example the humming of a computer would result in a very long reliable window of constant pitch. This is of course not a good example of music and hence if the reliable windows are becoming too long they are probably noise. Speech is characterized by the unvoiced parts with very bad reliability and voiced part with good reliability.

This feature is calculated using the flatness measure, which is normally used as the spectral flatness measure [Jayant, 1984]. It is a way of measuring the deviation from a completely flat plot and is given by the ratio between the geometric and the arithmetic mean,

$$pitchMonotonicity = \frac{e^{\frac{1}{I}\sum_{i=1}^{I}\log pitch_i}}{\frac{1}{I}\sum_{i=1}^{I} pitch_i}$$
$$reliabilityMonotonicity = \frac{e^{\frac{1}{I}\sum_{i=1}^{I}\log reliability_i}}{\frac{1}{I}\sum_{i=1}^{I} reliability_i} \quad (3.2.13)$$

3.3.7 Difference

To catch the feature that speech varies and music has a more constant behaviour the *max*- and *averageDeviation* was created based on the reliable windows. To create a feature that does not depend on the reliable windows the absolute difference between the pitch values is used,

$$d_i = \left| pitch_{i+1} - pitch_i \right| \ , \quad i = [1, 2, ..., I-1] \quad (3.2.14)$$

The number of diff values is one less than the total number of values in the signal.
The mean and standard deviation is calculated on the diff signal,

$$genericAbsDiffMean = \log \frac{1}{I-1}\sum_{i=1}^{I-1} d_i$$
$$genericAbsDiffDev = \sqrt{\frac{1}{I-2}\sum_{i=1}^{I-1}\left(d_i - genericAbsDiffMean\right)^2} \quad (3.2.15)$$

Also a feature based on the histogram of the diff is created. The maximum diff value is 350 because of the frequency range 50 to 400 Hz. Bins are created in a logarithmic fashion as specified by the table below.

1	2	3	4	5	6	7	8
[0;2[	[2;4[	[4;8[	[8;16[	[16;32[	[32;64[	[64;128[	[128;256[

A feature is created for each bin called *genericAbsDiff1-8*. The last bin from 256 to 350 is not included intentionally because it can be calculated from the other 8 bins and caused a singularity in the classification algorithms.

3.3.8 Mean & standard deviation

The mean and standard deviation are use together with the diff values, the reliability and the pitch itself. This is a general feature that is almost always used and is included here as well. The pitch mean and standard deviation are,

$$genericMean = \log \frac{1}{I}\sum_{i=1}^{I} pitch_i$$
$$genericDev = \sqrt{\frac{1}{I-1}\sum_{i=1}^{I}\left(pitch_i - genericMean\right)^2} \quad (3.2.16)$$

3.3.9 MCR - Mean Crossing Rate

MCR is a development of the zero-crossing-rate, ZCR [Saunders, 1996], which is a very used feature in sound classification, where it is used directly on the sound signal. MCR simply counts the number of times the mean value is crossed by the signal. ZCR

is only a good measure if the mean of the signal is 0, and the logical expansion of the ZCR on a positive signal is MCR.

$$genericMCR = \sum_{i=1}^{I-1} MC_i$$
$$MC_i = \begin{cases} 0 \ , & sign\left(pitch_i - \mu_p\right) = sign\left(pitch_{i+1} - \mu_p\right) \\ 1 \ , & otherwise \end{cases} \tag{3.2.17}$$

where μ_p is the mean of the pitch in the feature window and $sign(x)$ is the sign of x.

3.4 Logarithmic distribution of features

Some features show logarithmic behaviour. This means that the data is not gaussian distributed, but rather the logarithm of the data is gaussian. Whether the data is gaussian or not is important because a gaussian distribution is used in the model when doing the actual classifications. Even if the classification model did not use the gaussian distribution it is often convenient to use the logarithm anyways. This is because if all features have nearly the same distribution the decision boundaries between the classes become simpler. Because many data is gaussian distributed, the common distribution is chosen to be the gaussian one. Several ways of measuring gaussianity exist, a few will be presented and one will be used.

3.4.1 Log likelihood

And obvious way of comparing model fit is to take the log likelihood of the complete dataset. The model that fits best is chosen. This works if multiple models are fitted to the same dataset. This does not work however if the dataset is different. When comparing the original dataset with a logarithmic dataset, the datasets are different and it cannot be done. Instead a lognormal distribution could be used. This complicates things because different models must be used for different features. This will not be pursued any further.

3.4.2 Kurtosis

A feature in the search of gaussianity that is often used is the kurtosis. It is the relation between the central fourth order moment and the square of the central second order moment,

$$\kappa = \frac{\mu_4}{\mu_2^2} = \frac{E\left((x-\mu_1)^4\right)}{E\left((x-\mu_1)^2\right)^2} \tag{3.3.1}$$

It gives a value of 3 on a gaussian dataset, and gives lower values for distributions that are more flat and larger when the distribution is more peaked. As can be seen the kurtosis consists of two moments of even magnitude, 2 and 4. This means that it does not distinguish between the left and right tails of the distribution. The lognormal is a distribution that is not distributed equally to both sides of the mean and thus the kurtosis is a bad discriminator. It can result in a value of 3 even though the data set is far from gaussian.

3.4.3 Kolmogorov-Smirnov

The K-S test works from the cumulative distribution of the data. The cumulative distribution of the continuous distribution is given by

$$cdf(x) = \int_{-\infty}^{x} pdf(x) \tag{3.3.2}$$

The cumulative distribution of a data set is a function that increases by 1/N for each point passed in the sorted data

$$cdf(x) = \frac{n(x)}{N} \tag{3.3.3}$$

$n(x)$ is the number of samples below the value x. Both functions are monotically non-decreasing from 0 to 1.

The K-S test is simply the largest distance from the model *cdf* to the dataset *cdf*.

$$KS = \max\left|cdf_{model}(x) - cdf_{data}(x)\right| \quad (3.3.4)$$

The KS value can be looked up in a table to find the goodness of the fit. This is not so important in this case as the only information we want is whether the logarithm of the feature fits better or not. In this case a value is calculated for the original data and the logarithm of the data and the values are compared.

The advantage of this test is that it is independent of the dataset unit and thus can compare the performance of the gaussian distribution on both the original and the logarithm of the features.

Other tests exist for comparing the fit of a model. The Anderson-Darling is an enhancement of the Kolmogorov-Smirnov test. It weights the tails of the distribution more. The Chi square method works on histogram bins and can calculate fits for discrete distributions as well [28]. The Kolmogorov-Smirnov test was used because of its simplicity.

The test was run on all features. In the case where the original and the logarithm of the feature performed equal or close to equal the original feature was selected to increase simplicity. For the feature genericAbsDiff1 the logarithm should have been selected. Because the features genericAbsDiff2-8 was best at the original and they are all part of the same feature the original feature was used in all cases.

Two examples of the distribution of the data and the K-S test is shown below,

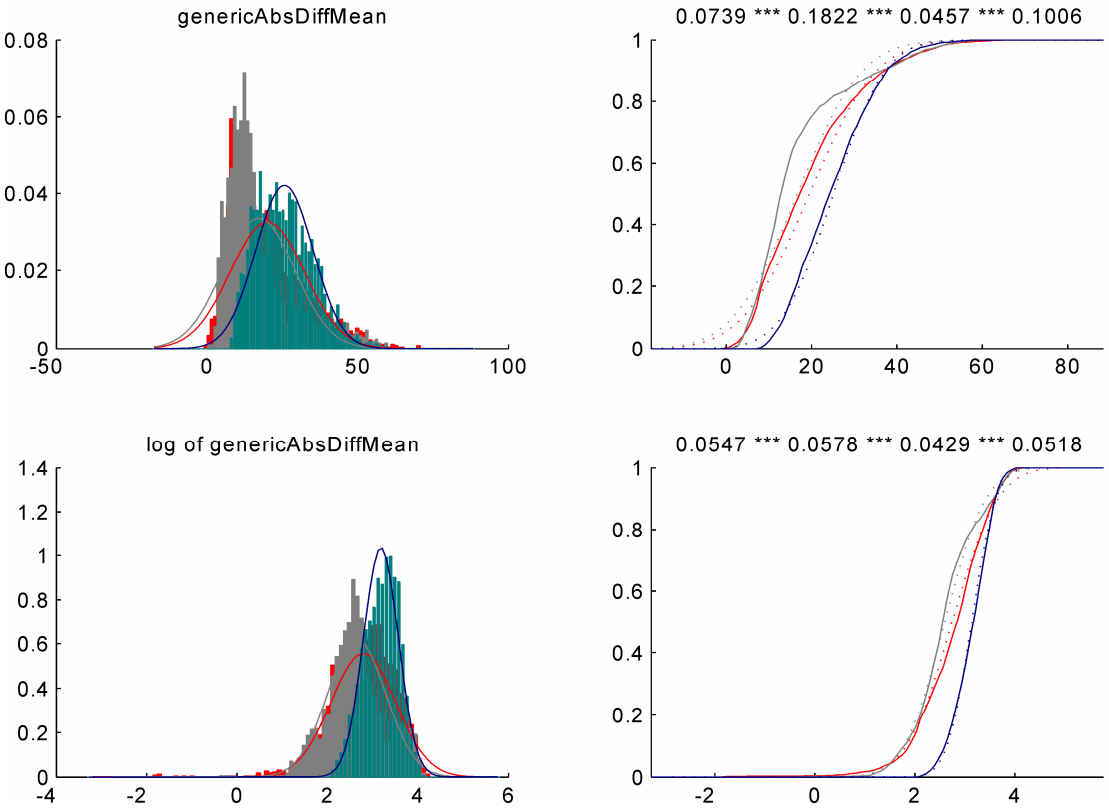


Figure 3.4.1: Feature number 14, *genericAbsDiffMean*. Left is the histograms and right is the K-S test. Numbers on top of K-S test is the maximum difference for music, noise, speech and the mean of the three separated by stars. In the plots music is red, noise is green and speech is grey. The bottom has a better K-S value and the logarithm of the feature is used.

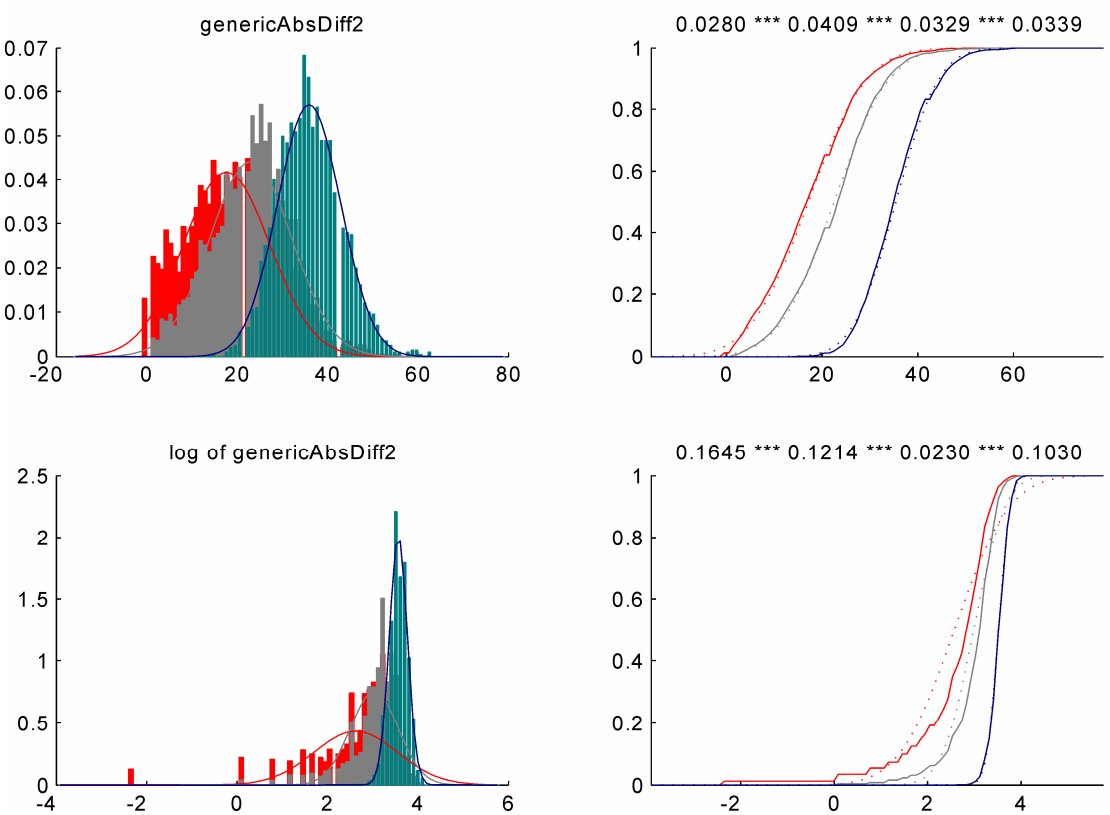


Figure 3.4.2: Same setup as figure above, but for the feature *genericAbsDiff2*. Here the feature it self has a better K-S value than when using the logarithm. The feature will be used by it self.

The features that use the logarithm are *maxWindowLength*, *averageWindowLength*, *genericMean*, *genericAbsDiffMean* and *genericReliabilityDev*.

4 Sound database

A database for training the model and for evaluation of the features is necessary. It must contain samples of each class. Some considerations must be done when creating the database.

General considerations when assembling a training set

For all training problems it is always better to have as large a training set as possible. This is because the parameters will be better estimated the larger a training set used. A large training set also means that each time a feature is evaluated or a new model is trained it takes longer time. This time can actually be quite consuming. Some of the comparisons done in this report took more than two days to run. This means the experiments cannot be undone or redone too many times without much time passing by. A compromise must be found.

It must be decided whether the database should include all scenarios or only the separate classes. When listening to music sometimes someone speaks to you. This would be a mixture of classes. This is of course very relevant as it occurs quite often. Also noise is often present when speaking, this can be in the traffic or at a party. The problem with using mixed classes is that it has to be decided what the target class should be. Is it the speech or the music that is important? This depends on the levels of both signals and will require studies of the thresholds of the mixing classes. This is not the subject of this project and thus only clean classes will be used. It also makes the classification simpler.

Instead of using mixed classes, the clean classes will be sought to cover as much of the class space as possible. Further details are found under each class description.

To facilitate the implementation of the algorithms a common sound format will be used. All sound clips have a sample rate of 10 kHz, a clip length of 30 s, and are in mono. In order to avoid only having the first 30 s of each sound, and because of limited availability of sounds, up to 5 clips are taken from the same original sound clip.

All recordings are provided by Oticon A/S.

Music

Music is perhaps the broadest of the three classes. It ranges from heavy metal, cross electronic and techno, to classical music and choir music which is close to speech. In total 50 minutes of music were used from 20 different song clips. The clips are listed below with short descriptions.

No.	Description	No.	Description
1-5	Choir	51-55	Percussion
6-10	Italian choir	56-60	Classical piano
11-15	Electronic	61-65	Male pop
16-20	Honolulu	66-70	R&B
21-25	Jazz	71-75	Rock
26-30	Macarena	76-80	Female pop
31-35	Middle eastern	81-85	Heavy metal
36-40	New age	86-90	Danish rock
41-45	Opera	91-95	Soft rock
46-50	Pan flute	96-100	Classical violin concert

Noise

Noise is also a quite broad class and it depends somewhat on what is defined as noise. Even though questionable clips have not been used, the class still includes quite different sounds. Especially the noise might cause problems for the classifier. In this project it has been stated that noise is characterized by not having pitch, but some noise do have pitch and sometimes a very constant pitch that could resemble music. For example the humming of a computer could be very tone like and this is actually the case for the lynx helicopter clip as well. A short description of the included sounds, totalling 40 minutes, is included below.

No.	Description	No.	Description
1-5	Café noise	41-45	Photocopy machine
6-10	Car driving, inside	46-50	Transport of beet
11-15	Coffee machine	51-55	Shallow water
16-20	Hotel foyer	56-60	Slaughter house
21-25	Hand mixer	61-65	Boys playing football
26-30	Intensive traffic, outside	66-70	Steel hall
31-35	Lynx helicopter	71-75	Steel melting hall
36-40	Factory noise	76-80	Train station

Speech

The speech class is probably the narrowest class of the three. Unfortunately it is quite difficult to get hold of because not very many speech recordings exist. The list has grown quite big though, covering many languages, females and males, conversations and monologues. Unfortunately children speaking was not found. A total of 42 minutes of speech is used and a listing with brief descriptions is included below.

No.	Description	No.	Description
1-4	Dutch, male	37-38	Danish, female
5-8	English, female	39-44	Russian, male & female
9-12	English, male	45-48	Russian, female
13-16	Italian, female	49-52	Russian, male
17-20	Italian, male	53-56	English, male & female
21-24	English, two females	57-62	American, male & female
25-26	English, two males	63	English, female
27	English, two females	64-65	English, male
28-29	English, two males	66-70	Chinese, male & female
30	Danish, female	71-75	Keele, female
31	English, female	76-79	Keele, male
32-35	Danish, female	80-84	Timit, male
36	English, male		

5 Classification model

The features must be included in a model to complete the classification. Many kinds exist including discriminant functions, K-nearest neighbour, K-means, kernel based methods and neural networks. The Bayes classifier is used in this project, because it is a simple model that shows good performance over a wide range of problems. Further more the parameters of the model can be interpreted and linked to the data directly. The Bayes classifier can be used in different variations of which a few will be described and used.

In the first section the model and two simplifications of it will be described. The results based on the model shows increasing training error for increasing dimensions. This will be the subject of the second section in which a new model will be suggested. Section three reveals some problems in the training of the new model which are solved by doing the training in multiple steps. Section four presents theoretical aspects of generative and discriminative models. Furthermore, is the logistic regression model presented as an example of the latter. Section five compares the performance of the new model against two models: the Bayes classifier (generative) and the logistic regression (discriminative). Finally section six uses a validation set, and Bayes and Akaike's Information Criteria to find an optimal number of features to be used.

5.1 Bayes classification

In this section the Bayes classifier together with the gaussian distribution will be reviewed. Training versus validation sets will briefly be discussed and two different ways of ranking the features are presented. Finally, experiments are run on the database, and the rankings found are verified by searching all combinations on a smaller set of features. A strange increase of the training error with increasing dimensions, is observed.

5.1.1 The model

The Bayesian classifier works by using Bayes' theorem. The joined probability of $\boldsymbol{x}$ and c can be written in two different ways,

$$p(\boldsymbol{x},c) = p(\boldsymbol{x}\,|\,c)\,p(c) = p(c\,|\,\boldsymbol{x})\,p(\boldsymbol{x}) \tag{5.1.1}$$

$\boldsymbol{x}$ is a continuate random vector specifying a measurement point and c is an integer specifying a specific class. Equation (5.1.1) can be rearranged like this,

$$p(c\,|\,\boldsymbol{x}) = \frac{p(\boldsymbol{x}\,|\,c)\,p(c)}{p(\boldsymbol{x})} \tag{5.1.2}$$

This last equation is called Bayes' theorem [Bishop, 2004, chap. 1]. Although simple to realize it has deep implications. With this equation, the probability of a variable c can be found dependent on another variable $\boldsymbol{x}$, without modelling the probability directly. When doing classification, this is exactly what is wanted. The characteristics are measured at a given time, which gives a point in measurement space. With Bayes' theorem the probability of a point coming from a specific class can be found. It should be mentioned that $\boldsymbol{x}$ and c in general could both be vectors either discrete or continuate.

The left hand side of Bayes theorem is called the posterior probability, but what still needs to be found is the variables on the right hand side. $p(c)$ is called the prior probability. It is the probability of a class when no information about the point $\boldsymbol{x}$ is present. When little prior information is present equal probabilities are used. This is often the case and will also be used here. It means that the classification will be done entirely based on the measured point and not be influenced by knowledge besides that. $p(\boldsymbol{x}\,|\,c)$ is called the class conditional probability of $\boldsymbol{x}$, meaning the probability of a given point depending on the class. A model of each class is generated without worrying about the other classes. Often the gaussian distribution model is used, and it will be investigated shortly. $p(\boldsymbol{x})$ works as a normalization factor which ensures that the class probabilities sum to one. When c is a discrete variable it can be found like this,

$$p(\boldsymbol{x}) = \sum_{c=1}^{C} p(\boldsymbol{x}\,|\,c)\,p(c) \tag{5.1.3}$$

where C is the total number of classes. Bayes' theorem can now be written like this,

$$p(c\,|\,\boldsymbol{x}) = \frac{p(\boldsymbol{x}\,|\,c)\,p(c)}{\sum_{c=1}^{C} p(\boldsymbol{x}\,|\,c)\,p(c)} \tag{5.1.4}$$

and in this equation, it is easily verified that the probabilities will sum to one.

As mentioned above the gaussian model will be used for the class conditional probability, $p(\boldsymbol{x}|c)$. The gaussian model has the following probability density function [Johnson, 2002, chap. 4],

$$p(\boldsymbol{x}|\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c)=\frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}_c|}}e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_c)^T\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_c)} \tag{5.1.5}$$

where d is the dimension of the $\boldsymbol{x}$ vector, and $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ are model variables of the gaussian model and are denoted mean and covariance. They are defined by the two expectations,

$$\boldsymbol{\mu}_c=E\{\boldsymbol{x}\}\ ,\quad \boldsymbol{\Sigma}_c=E\left\{(\boldsymbol{x}-\boldsymbol{\mu}_c)(\boldsymbol{x}-\boldsymbol{\mu}_c)^T\right\} \tag{5.1.6}$$

To get probabilities from a density function, an integration over a specified area must be calculated. This is not done in the Bayes classifier. Instead the density values are used directly as if they were probabilities. Strictly speaking they are not, and can in fact have values well over 1. Therefore probabilities are not used any longer, but only likelihoods.

The true distributions of the classes are of course not known and must be estimated based on a training set. The estimation relies on an error function. When dealing with probability models a commonly used one is the maximum likelihood error function. The measurement points are considered to be independent of each other. This means that the joined likelihood of a number of points can be found by the product of their likelihoods,

$$p(\boldsymbol{X}|\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c)=\prod_{n=1}^{N}p(\boldsymbol{x}_n|\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c) \tag{5.1.7}$$

where $\boldsymbol{X}$ is N observations of the variable $\boldsymbol{x}$. It is this likelihood that is maximized and gives the following maximum likelihood estimate of the mean and the unbiased estimate of the covariance [Johnson, 2002, chap. 3],

$$\hat{\boldsymbol{\mu}}_c=\frac{1}{N_c}\sum_{n=1}^{N_c}\boldsymbol{x}_n\ ,\quad \hat{\boldsymbol{\Sigma}}_c=\frac{1}{N_c-1}\sum_{n=1}^{N_c}(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_c)^T \tag{5.1.8}$$

Because the covariance matrix is symmetric by definition, the number of free parameters of the Bayes classifier is,

$$C\frac{d(d+3)}{2} \tag{5.1.9}$$

where C is the total number of classes. Because of the curse of dimensionality and the danger of overfitting the model to the training data, it can be desirable to limit the number of free parameters. Also for computational reasons this can be desirable. The mean is always calculated in the same way, but to limit the quite big number of free parameters in equation (5.1.9), different variations in the calculation of the covariance are used. One way to limit the parameters is to use a common covariance matrix for all classes. This limits the free parameters to,

$$Cd+\frac{d(d+1)}{2} \tag{5.1.10}$$

The joined covariance matrix is found by combining the individual covariances like this [Johnson, 2002, chap. 11],

$$\hat{\boldsymbol{\Sigma}}=\sum_{c'=1}^{C}\frac{N_{c'}-1}{\sum_{c''=1}^{C}(N_{c''}-1)}\hat{\boldsymbol{\Sigma}}_{c'} \tag{5.1.11}$$

or by calculating it directly,

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N-C}\sum_{n=1}^{N}(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^T \tag{5.1.12}$$

where N is the number of points from all classes. When evaluating the sum in equation (5.1.12) each term uses the $\boldsymbol{\mu}_c$ from the class that $\boldsymbol{x}_n$ targets.

Another variation of the gaussian model assumes that the dimensions of the data within each class are independent of each other. This means that the covariance reduces to a diagonal matrix and greatly reduces the number of free parameters which is given by,

$$2Cd \tag{5.1.13}$$

This approach is sometimes called Naive Bayes because of the independence assumption. The covariance is given by,

$$\hat{\boldsymbol{\Sigma}}_c = diag\left(\frac{1}{N_c - 1}\sum_{n\in c}(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)\circ(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)\right) \tag{5.1.14}$$

$\circ$ is the element wise (Hadamard) product and *diag* forms a matrix with the specified vector in the diagonal and zero elsewhere.

When evaluating the Bayes classification likelihood of a set of points, the error is again found by assuming independent points and using the product of the likelihoods,

$$p(\boldsymbol{c} \mid \boldsymbol{X}) = \prod_{n=1}^{N} p(c_n \mid \boldsymbol{x}_n) \tag{5.1.15}$$

c_n specifies the class that $\boldsymbol{x}_n$ targets and $\boldsymbol{c}$ is a vector containing all c_n. Instead of using the likelihood directly the negative logarithm of the likelihood is often used as an error function. The likelihood function is always positive and the logarithm function is monotonically increasing. This means that discrimination between models is not affected by the logarithm and it makes the likelihood computations simpler,

$$E = -\log p(\boldsymbol{c} \mid \boldsymbol{X}) = -\sum_{n=1}^{N} \log p(c_n \mid \boldsymbol{x}_n) \tag{5.1.16}$$

and the gaussian model is simplified by the logarithm. Furthermore the error function decreases the better the classification, and the minimum error obtainable is 0. The error is referred to as the negative log likelihood.

When classifying a point this function is optimized,

$$q_n = \max_c p(c \mid \boldsymbol{x}_n, \theta) \tag{5.1.17}$$

where θ is the Bayes model with the mean, covariances and prior likelihoods and q_n is the assigned class. The equation simply states that the point will be assigned to the class with the biggest posterior likelihood.
Another error function that is often used is the classification error. This is the number of points that is assigned to the wrong class divided by the total number of points,

$$E_{miss} = \frac{1}{N}\sum_{n=1}^{N} e_n \quad , \quad e_n = \begin{cases} 1 \ , & q_n \neq c_n \\ 0 \ , & q_n = c_n \end{cases} \tag{5.1.18}$$

The classification error is a bit more informative than the negative log likelihood, but it is discrete. This makes analytical reviewing more difficult. The classification error is reported together with the negative log likelihood through out the project.

5.1.2 Training data

When using the Bayes classifier, the model parameters must somehow be found. This is done using a training set containing a number of points for which the classes are already known. It consists of a vector $\boldsymbol{c}$ with N class targets and a matrix $\boldsymbol{X}$ with N measurement points. The training can be divided into two different parts. One part is to fit the model parameters in an optimal way. The other part is finding the optimal complexity of the model. The more complex the model, the more accurate the training data can be fitted. The goal is not to fit the training data, but rather to fit the underlying relations generating the training data, and this is probably not achieved just by making a more and more complex model. This means that even though the error found on the training data might be worse for a simpler model, it can actually be a better model of the underlying relations. This causes a split of the error into a training error based on the dataset used for the parameter optimization and a test error or validation error based on a different set of points [Bishop, 2004, chap 9]. For now, only the optimization of the parameters and thus the training error will be used and later the complexity of the model will be investigated.

5.1.3 Feature ranking

The model is now in place. The features found in the previous chapter could simply be used and a model would be available. It is not convenient to use all of the features though. Some features are better at classifying the data than others. Some of the features are very similar and are expected to be correlated, for example the tone distance. Two features exist of this kind, one based on the reliable windows and one that uses the entire feature window. Even though they should not give the exact same values, the two features are expected to be very correlated.

The task is to rank the features in some way. The goal is to find the best model of each dimension, meaning the best model using only one feature, the best model using two features and so on. There is no easy way to find the best subset of features. The only way to be certain is to test all combinations. Some features compliment each other in unpredictable ways and gives an optimal classification, whereas when used separately they provide little information. For 28 features it is an enormous task to test all combinations (2^{28}-1 combinations exist) and is seldom a feasible way of solving the problem. Several other methods exist though. For example the covariance matrix can be used to see what features are correlated and which are not. This gives a good estimate of how much correlation exists on a pair wise level. It can also be investigated if two features when used by them selves miss the same points or if they compliment each other. If they miss the same points one of them can be left out. The most commonly known methods are forward selection and backward elimination, which are both quite methodical. They will be presented next.

Forward selection

The forward selection algorithm [Bishop, 2004, chap. 9] starts by finding the best feature for doing classification on its own. Then the feature that compliments the first best is found by trying all the remaining features together with the first feature. Next, the third feature that compliments the first two features best is found and so on. It can also be explained by starting with two vectors of features. At first one is empty and the other contains all features. Then by using each feature in the second vector for classification on its own the best is found, added to the first vector and removed from the second vector. Next all remaining features in the second vector are added one by

one to the first vector and the first vector is used for classification. Again the best feature is found and added to the first vector and removed from the second vector. This procedure is continued until no features are left in the second vector. Now the first vector contains a prioritized numbering of the features.

Backward elimination

Backward elimination [Bishop, 2004, chap. 9] works in a similar, but reversed way. It starts by classifying with all features. Then all features are removed one by one to find the feature that degrades the classification the least and this feature is taken out and put in a feature vector. This is continued until only one feature is left and again a prioritized vector is the result. Because none of the methods assures the optimal subsets they do not necessarily give the same ranking of the features.

Both ways of selecting features results in a dramatic decrease in search space. Instead of doing 2^{28}-1 comparisons only 28+27+…+1 comparisons has to be performed. For the 28 features this is 406 comparisons compared to 268.435.455 comparisons in the exhaustive search. The danger of using these procedures is that an optimal selection of features is neglected and a suboptimal result is found. The performance of the selection schemes used will be evaluated later in this section.

5.1.4 Experimental results

The models are used on the data from the database presented in the previous chapter. It contains about two hours of sound of the three classes - music, noise and speech. That makes a total of 6600 samples. This database will be used through out the project.

Forward selection

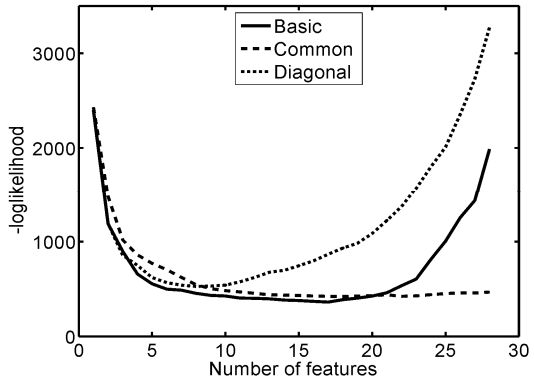


Figure 5.1.1: Training error. An unexpected increase in the error is observed for increasing dimensions.

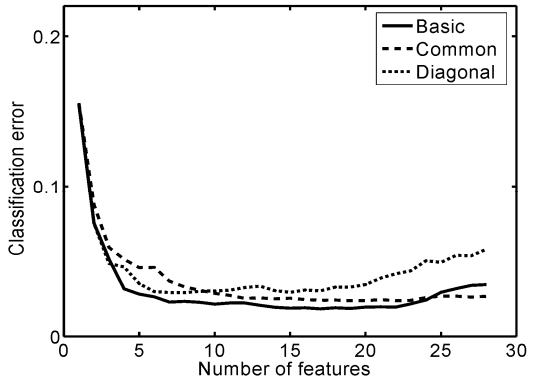


Figure 5.1.2: Classification error. The same increase in error is observed.

The plot of the negative log likelihood does not give a certain conclusion about which is the better variation. The basic variation gives the smallest error until about 20 features. It is not surprising that the basic variation is the best, because it has far more free parameters and should be able to fit the data better. What is surprising is the increase in error at the end. Since this is strictly the training error without any test or validation, the error is expected to decrease the more information is available and the more complex the model gets. The error increases in the end for all the variations, but is worst for the diagonal case and best for the common covariance.

Backward elimination

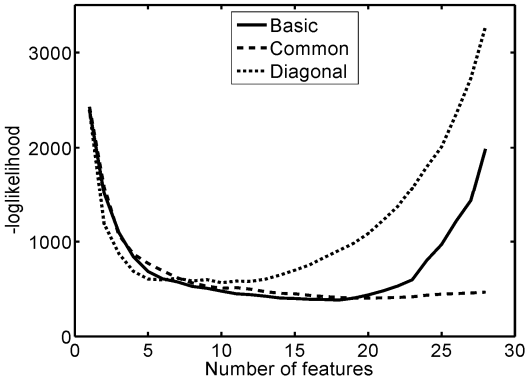


Figure 5.1.3: Training error. The values are almost the same as for forward selection which suggest near optimal parameters are found.

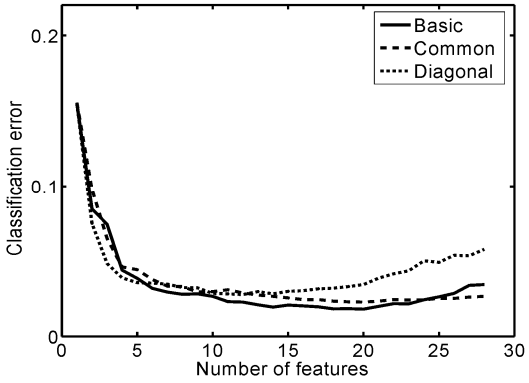


Figure 5.1.4: Classification error.

The backward elimination of the features gives very similar results. This is good for a selection point of view, and gives a hint that the subsets found by these algorithms are also close to the subsets found by the exhaustive search. The diagonal variation shows better performance here with fewer features than it did in the former. The same trend of increasing error with more features is the same as before. This will be the topic of the next section.

The prioritized sequence of the features for both ranking schemes and all variations is,

Forward selection	
Basic	28 25 4 15 16 20 19 22 18 27 2 12 7 6 24 9 8 14 3 26 1 5 17 21 23 13 10 11
Common	28 18 25 19 5 15 22 21 4 12 1 7 26 2 13 10 3 9 11 17 6 27 23 8 14 16 24 20
Diagonal	28 25 18 15 22 4 26 21 14 17 1 9 7 19 27 8 5 13 6 12 23 24 2 10 16 20 3 11
Backward elimination	
Basic	28 16 15 7 19 4 27 2 22 18 24 21 17 8 6 9 12 25 26 1 3 14 5 13 10 20 23 11
Common	28 19 25 4 13 12 18 21 1 22 6 11 27 7 3 14 2 10 5 9 16 23 17 15 24 26 8 20
Diagonal	28 25 4 15 22 21 6 13 19 7 17 26 14 18 9 8 27 5 12 1 23 24 2 10 16 20 3 11

The priority does vary over the different runs, but are similar in general terms. They all agree that feature 28 is the best feature and most that feature 11 is the worst. Differences, in the choice of features, between the different covariance variations are expected, for example because the diagonal assumes independent features which the other two do not. The forward selection and backward elimination on the same variation should give the same result. The basic variation agrees on 8 out of 10 and 12 out of 15 features. The diagonal variation agrees on 8 out of 10 and 11 out of 15 features. The common variation agrees on 6 out of 10 and 13 out of 15 features. The most important thing is that the errors are almost the same. The differences, in the selection of the features, are a sign that some features give the same amount of information, and which feature is chosen is not important.

Verification of feature selection

To investigate if a suboptimal subset of features is selected using the ranking schemes, a validation will be performed. Because of the huge number of combinations possible it is impossible to check all. Instead the best 15 features are selected based on the 6 experiments from above. This smaller number of features can be exhaustively

searched in not too much time. By comparing the achieved error rates with the ones from the optimal solution it can be indicated if the ranking schemes are effective or not. Of course it is still only an indication and to be sure the complete search has to be done with all features.
A plot of both the negative log likelihood and the miss classification rate is shown below. The 15 features that are used are: 1, 2, 4, 6, 7, 12, 13, 15, 18, 19, 21, 22, 25, 27, and 28. Numbering can be found in appendix.

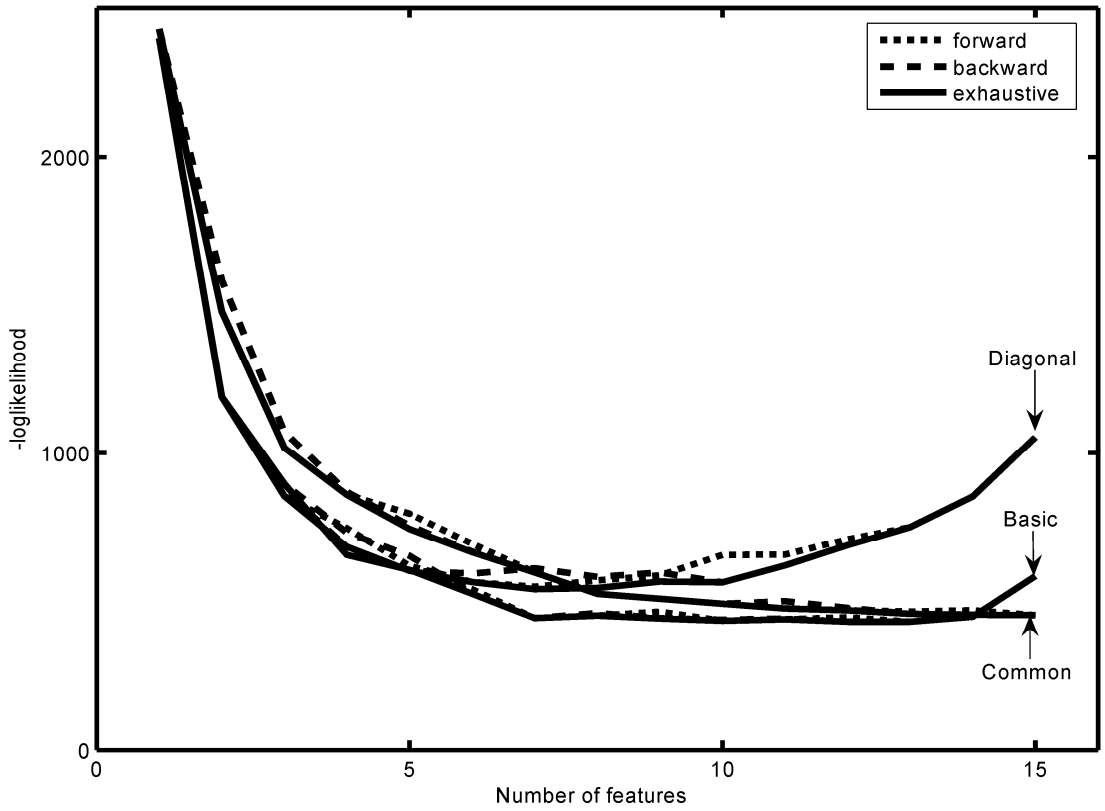


Figure 5.1.5: Training error. The exhaustive search gives only a slight improvement and only for some dimensions. This shows that the ranking schemes are effective.

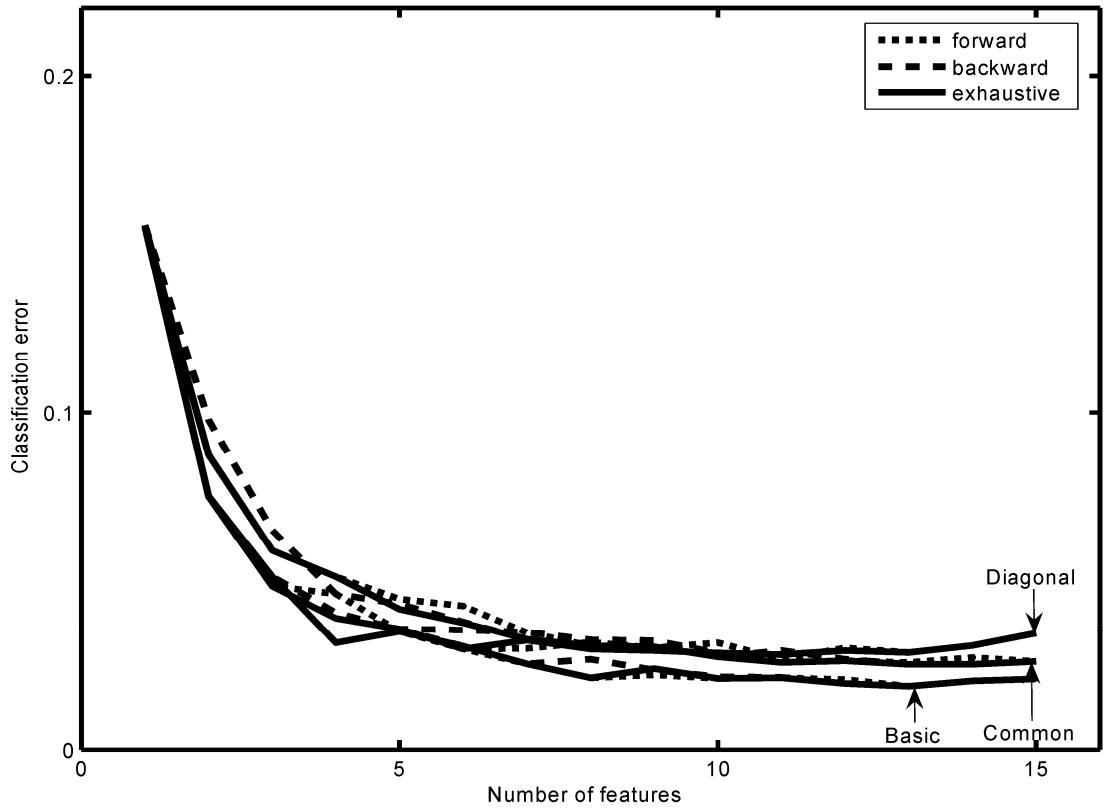


Figure 5.1.6: Classification error. Again the exhaustive search gives very little improvement.

As can be seen from the figures the classification error of the optimal subset is very close to the classification error found by the selection schemes. At many places on the 'Number of features' axis do the methods result in the same error as the exhaustive search, i.e. the results from the selection schemes can be trusted as good subsets.

5.2 Training a gaussian model discriminatively

In this section the phenomenon with the increasing training error seen in the previous section will be explained. It will be shown that it is caused by training the model generatively and a new method for training the model discriminatively is derived. The terms generative and discriminative will be explained in section four.

5.2.1 Increasing training error

In the previous section an interesting result was achieved in the ranking of features. It seemed the error started to increase when a certain number of features was exceeded. Further more with many features the error was smaller for the simpler model than the more complex model. This is a bit strange because the variation with common covariance is included in the basic variation, so at least equal error should be obtained.

In order to find an explanation for this, it is necessary to investigate what is being optimized and what the error function is. The error function is specified by the negative sum of the posterior log likelihoods given by,

$$E = -\sum_{n=1}^{N} \log\left(p\left(c_n \mid \boldsymbol{x}_n\right)\right) \tag{5.2.1}$$

This function can be split into the three classes,

$$\begin{aligned} E &= -\sum_{N_1} \log\left(p\left(c_1 \mid \boldsymbol{x}_{n_1}\right)\right) - \sum_{N_2} \log\left(p\left(c_2 \mid \boldsymbol{x}_{n_2}\right)\right) - \sum_{N_3} \log\left(p\left(c_3 \mid \boldsymbol{x}_{n_3}\right)\right) \\ &= -\sum_{N_1} \log\left(\frac{p\left(\boldsymbol{x}_{n_1} \mid c_1\right)}{\sum_{c'=1}^{C} p\left(\boldsymbol{x}_{n_1} \mid c'\right)}\right) - \sum_{N_2} \log\left(\frac{p\left(\boldsymbol{x}_{n_2} \mid c_2\right)}{\sum_{c'=1}^{C} p\left(\boldsymbol{x}_{n_2} \mid c'\right)}\right) - \sum_{N_3} \log\left(\frac{p\left(\boldsymbol{x}_{n_3} \mid c_3\right)}{\sum_{c'=1}^{C} p\left(\boldsymbol{x}_{n_3} \mid c'\right)}\right) \end{aligned} \tag{5.2.2}$$

From this equation it can be seen that the parameters of all three classes influences the error in all points and not only the ones targeted for that class. As was explained previously the optimization only optimizes the error of a single class without accounting for the dependency to the other classes. This is what causes the increase in the training error. The model is simply not optimized for classification, but is optimized for the within class likelihood.

A small two-class one-dimensional problem is designed to visualize the problem.

Two classes have been constructed, consisting of 1000 points. Both are squared gaussian distributions with zero mean and variance one constructed like this,

$$C_1 = \left(N(0,1)\right)^2 \quad , \quad C_2 = -2\left(N(0,1)\right)^2 \tag{5.2.3}$$

They are plotted in figure 5.2.1 together with their maximum likelihood gaussian models. Of course they do not fit very well because the datasets are not gaussian, but still it is the best way a gaussian model can be fit to each class. Figure 5.2.2 shows the class likelihood found using Bayes.

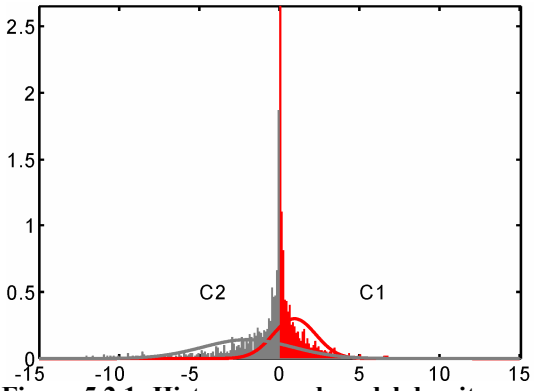


Figure 5.2.1: Histogram and model density function of simple example.

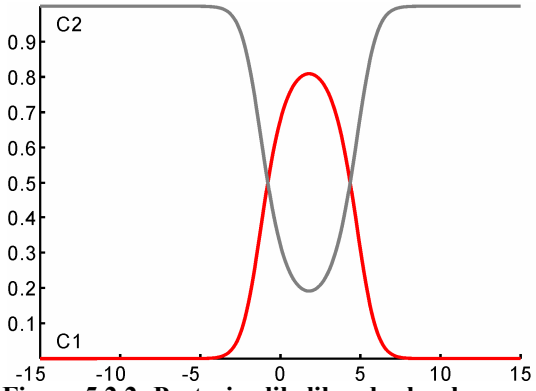


Figure 5.2.2: Posterior likelihoods clearly result in non optimal decision boundaries.

Misclassifications are generated just left of zero and again to the right where C_2 is preferred even though no points exist in this part. The misclassification around 0 can be removed simply by moving both mean values a little to the right, so clearly the classification task has not been optimized when optimizing the individual classes.

5.2.2 Maximum likelihood in classification

Previously it was shown that the problem with the error was due to the fact that the error function, which is maximized during training, is not the error function used to evaluate the classification model. In this section the classification error function will be trained instead and this should result in better classification. When optimizing parameters the derivative of the error function is found and set to 0. First the error function is rearranged to facilitate the derivations,

$$
\begin{aligned}
E &= -\sum_{n=1}^{N} \log\left(p\left(c_n \mid \boldsymbol{x}_n\right)\right) \\
&= -\sum_{n=1}^{N} \log\left(\frac{p\left(\boldsymbol{x}_n \mid c_n\right) p\left(c_n\right)}{\sum_{c'=1}^{C} p\left(\boldsymbol{x}_n \mid c'\right) p\left(c'\right)}\right) = \sum_{n=1}^{N} \log\left(\sum_{c'=1}^{C} \frac{p\left(\boldsymbol{x}_n \mid c'\right) p\left(c'\right)}{p\left(\boldsymbol{x}_n \mid c_n\right) p\left(c_n\right)}\right) \\
&= \sum_{n=1}^{N} \log\left(\sum_{c'=1}^{C} e^{\log\left(\frac{p(\boldsymbol{x}_n|c')p(c')}{p(\boldsymbol{x}_n|c_n)p(c_n)}\right)}\right) = \sum_{n=1}^{N} \log\left(\sum_{c'=1}^{C} e^{\log\left(p(\boldsymbol{x}_n|c')\right)-\log\left(p(\boldsymbol{x}_n|c_n)\right)+\log\left(\frac{p(c')}{p(c_n)}\right)}\right) \\
&= \sum_{n=1}^{N} \log\left(\sum_{c'=1}^{C} e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right), \quad Q_c = \log\left(p\left(\boldsymbol{x}_n \mid c\right)\right), \quad \gamma_{c_1,c_2} = \log\left(\frac{p\left(c_1\right)}{p\left(c_2\right)}\right)
\end{aligned}
\tag{5.2.4}
$$

The derivatives with regard to the parameters in the gaussian model are found. In stead of doing the complete evaluation for both parameters a substitute variable, u_c, is used at first,

$$\begin{aligned}\frac{\partial E}{\partial u_c} &= \sum_{n=1}^{N}\left(\sum_{c'=1}^{C} e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)^{-1}\sum_{c'=1}^{C}\left(e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\left(\delta_{c',c}\frac{\partial Q_{c'}}{\partial u_c}-\delta_{c_n,c}\frac{\partial Q_{c_n}}{\partial u_c}\right)\right)\\ &= \sum_{n=1}^{N}\left(\sum_{c'=1}^{C} e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)^{-1}\left(e^{Q_c-Q_{c_n}+\gamma_{c,c_n}}\frac{\partial Q_c}{\partial u_c}-\delta_{c_n,c}\frac{\partial Q_c}{\partial u_c}\sum_{c'=1}^{C} e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)\\ &= \sum_{n=1}^{N}\left(\sum_{c'=1}^{C} e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)^{-1}\left(e^{Q_c-Q_{c_n}+\gamma_{c,c_n}}-\delta_{c_n,c}\sum_{c'=1}^{C}\left(e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)\right)\frac{\partial Q_c}{\partial u_c}\end{aligned} \tag{5.2.5}$$

where $\delta_{x,y}$ is the Dirac delta function and gives 1 if $x = y$ and 0 otherwise. The equation is a weighted sum of the derivatives,

$$\begin{aligned}\frac{\partial E}{\partial u_c} &= \sum_{n=1}^{N} w_{c,n}\frac{\partial Q_c}{\partial u_c}\\ w_{c,n} &= \left(\sum_{c'=1}^{C} e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)^{-1}\left(e^{Q_c-Q_{c_n}+\gamma_{c,c_n}}-\delta_{c_n,c}\sum_{c'=1}^{C}\left(e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)\right)\end{aligned} \tag{5.2.6}$$

The weights can be written back to the likelihoods which simplifies the equation,

$$\begin{aligned}w_{c,n} &= \left(\sum_{c'=1}^{C} e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)^{-1}\left(e^{Q_c-Q_{c_n}+\gamma_{c,c_n}}-\delta_{c_n,c}\sum_{c'=1}^{C}\left(e^{Q_{c'}-Q_{c_n}+\gamma_{c',c_n}}\right)\right)\\ &= p(c_n\mid \boldsymbol{x}_n)\left(\frac{p(\boldsymbol{x}_n\mid c)\,p(c)}{p(\boldsymbol{x}_n\mid c_n)\,p(c_n)}-\delta_{c_n,c}\frac{1}{p(c\mid \boldsymbol{x}_n)}\right)\\ &= p(c\mid \boldsymbol{x}_n)-\delta_{c_n,c}\end{aligned} \tag{5.2.7}$$

In the gaussian distribution two sets of parameters exist to optimize: the mean and the covariance. The Q function looks like this,

$$Q_c = -\frac{d}{2}\log(2\pi)-\frac{1}{2}\log\left(\left|\hat{\boldsymbol{\Sigma}}_c\right|\right)-\frac{1}{2}(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_c)^T\,\hat{\boldsymbol{\Sigma}}_c^{-1}(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_c) \tag{5.2.8}$$

The covariance matrix and the mean are treated separately. First the mean,

$$\begin{aligned}\frac{\partial Q_c}{\partial \hat{\boldsymbol{\mu}}_c} &= \hat{\boldsymbol{\Sigma}}_c^{-1}(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_c)\\ \frac{\partial E}{\partial \hat{\boldsymbol{\mu}}_c} &= \sum_{n=1}^{N} w_{c,n}\hat{\boldsymbol{\Sigma}}_c^{-1}(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_c)\end{aligned} \tag{5.2.9}$$

If set to zero it gives,

$$\hat{\boldsymbol{\mu}}_c = \frac{\sum_{n=1}^{N} w_{c,n}\boldsymbol{x}_n}{\sum_{n=1}^{N} w_{c,n}} \tag{5.2.10}$$

The covariance matrix is a symmetric matrix and for this reason the derivative gets a bit complicated [Petersen, 2005, chap. 2]. It is given by,

$$\begin{aligned}\frac{\partial Q_c}{\partial \hat{\boldsymbol{\Sigma}}_c} &= -\hat{\boldsymbol{\Sigma}}_c^{-1} + \frac{1}{2}\hat{\boldsymbol{\Sigma}}_c^{-1} \circ \boldsymbol{I} + \hat{\boldsymbol{\Sigma}}_c^{-1}(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^T \hat{\boldsymbol{\Sigma}}_c^{-1} - \\ &\qquad \frac{1}{2}\hat{\boldsymbol{\Sigma}}_c^{-1}(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^T \hat{\boldsymbol{\Sigma}}_c^{-1} \circ \boldsymbol{I} \\ &= \hat{\boldsymbol{\Sigma}}_c^{-1}\left((\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^T \hat{\boldsymbol{\Sigma}}_c^{-1} - \boldsymbol{I}\right) - \\ &\qquad \frac{1}{2}\left(\hat{\boldsymbol{\Sigma}}_c^{-1}\left((\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^T \hat{\boldsymbol{\Sigma}}_c^{-1} - \boldsymbol{I}\right)\right) \circ \boldsymbol{I} \\ &= \boldsymbol{M} - \frac{1}{2}\boldsymbol{M} \circ \boldsymbol{I} \quad , \quad \boldsymbol{M} = \hat{\boldsymbol{\Sigma}}_c^{-1}\left((\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^T \hat{\boldsymbol{\Sigma}}_c^{-1} - \boldsymbol{I}\right) \\ \frac{\partial E}{\partial \hat{\boldsymbol{\Sigma}}_c} &= \sum_{n=1}^{N} \boldsymbol{w}_{c,n} \frac{\partial Q_c}{\partial \hat{\boldsymbol{\Sigma}}_c} = \sum_{n=1}^{N} \boldsymbol{w}_{c,n}\left(\boldsymbol{M} - \frac{1}{2}\boldsymbol{M} \circ \boldsymbol{I}\right)\end{aligned} \tag{5.2.11}$$

$\circ$ is the Hadamard product which is the element wise product of two matrices. Because of this product, the equation becomes very complicated to solve for equal to zero.

When little information of the prior likelihoods is present they are often assumed to be identical and thus go out of the equations. For completeness, though, the derivatives are found,

$$\frac{\partial E}{\partial p(c)} = \frac{1}{p(c)} \sum_{n=1}^{N} p(c \mid \boldsymbol{x}_n) - \delta_{c=c_n} \tag{5.2.12}$$

Details are in appendix.
This cannot be solved for equal to zero. In the experiments the priors are assumed equal and are not part of the optimization algorithm.

The derivative of the covariance could not be solved when set to zero, and a gradient approach will be used instead. Several gradient based methods exist, but the most basic method is simply called gradient descent. A gradient descent method is a simple way of checking if better performance can be achieved. It does not have fast convergence, but if good results appear it can be optimized in different ways. The gradient descent algorithm works by taking small steps in the direction of the negative derivative. If an appropriate step size is chosen it results in a (local) minimum.

$$\begin{aligned}\frac{\partial E}{\partial \boldsymbol{\mu}_c} &= \sum_{n=1}^{N} w_{c,n} \boldsymbol{\Sigma}_c^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_c) = \sum_{n=1}^{N}\left(p(c \mid \boldsymbol{x}_n) - \delta_{c_n,c}\right)\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_c) \\ \frac{\partial E}{\partial \hat{\boldsymbol{\Sigma}}_c} &= \sum_{n=1}^{N} w_{c,n}\left(\boldsymbol{M} - \frac{1}{2}\boldsymbol{M} \circ \boldsymbol{I}\right) = \sum_{n=1}^{N}\left(p(c \mid \boldsymbol{x}_n) - \delta_{c_n,c}\right)\left(\boldsymbol{M} - \frac{1}{2}\boldsymbol{M} \circ \boldsymbol{I}\right) \\ \boldsymbol{M} &= \hat{\boldsymbol{\Sigma}}_c^{-1}\left((\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^T \hat{\boldsymbol{\Sigma}}_c^{-1} - \boldsymbol{I}\right)\end{aligned} \tag{5.2.13}$$

The new mean and variance are updated like this [Bishop, 2004, chap. 7],

$$\begin{aligned}\hat{\boldsymbol{\mu}}_{c,new} &= \hat{\boldsymbol{\mu}}_{c,old} - \alpha \frac{\partial E}{\partial \hat{\boldsymbol{\mu}}_c} \\ \hat{\boldsymbol{\Sigma}}_{c,new} &= \hat{\boldsymbol{\Sigma}}_{c,old} - \alpha \frac{\partial E}{\partial \hat{\boldsymbol{\Sigma}}_c}\end{aligned} \tag{5.2.14}$$

where α is a constant controlling the step size.
As this is an iterative approach an initial guess is needed to start the process. Depending on the error surface as a function of the parameters, this can have small or big influence on the end result. A first attempt uses the within class maximum likelihood estimates of the parameters.

Results show that it is indeed possible to gain better performance this way. On the small, two class, one-dimensional problem the algorithm performs like this,

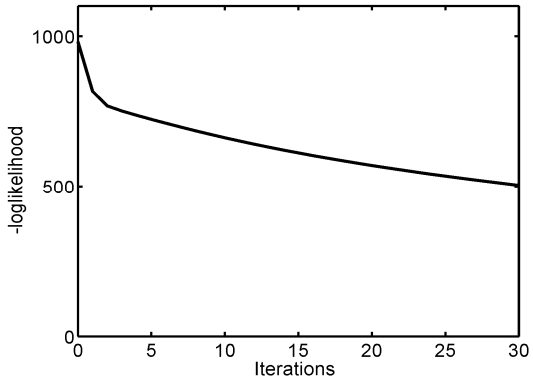


Figure 5.2.3: Training error of new model with gradient descent.

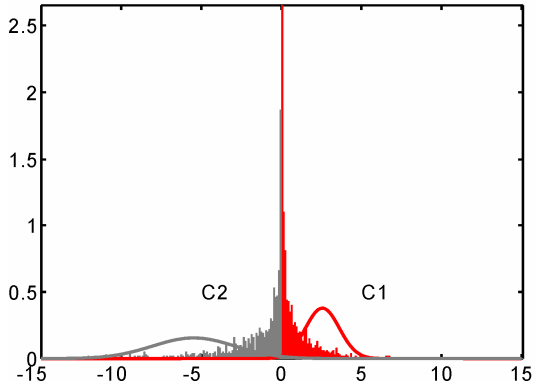


Figure 5.2.4: Density functions after training.

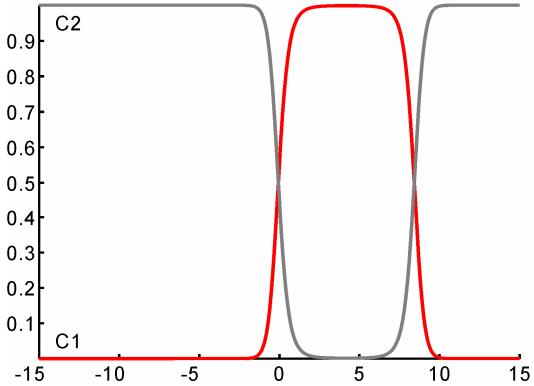


Figure 5.2.5: Posterior likelihoods after training are clearly improved.

The classification problem is clearly performing better. Convergence of the gradient descent algorithm is slow and many improvements exist, for example using line search to optimize the step size and using variations of the second derivative to optimize the search direction. An algorithm from immoptibox [27] is implemented. It is based on a quasi-Newton method and uses an approximate inverse Hessian (matrix of the second derivatives). This algorithm results in figure 5.2.6 and figure 5.2.7,

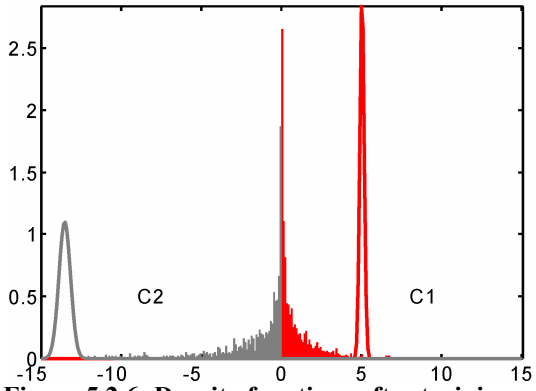


Figure 5.2.6: Density functions after training with advanced algorithm.

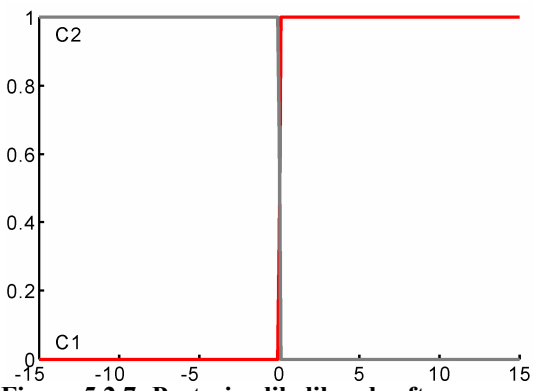


Figure 5.2.7: Posterior likelihoods after training are near optimal.

The classification in figure 5.2.7 is clearly optimal and the two gauss models are clearly far from the original classes. A problem was visualized with the procedure though. To achieve the sharp separation the variances becomes smaller and smaller and the means further and further apart. This means no defined minimum is present and the algorithm ends in a singular point. This is a general problem of the new model and training will be investigated further in the next section.

To show the connection between the new and the original way of training, it is shown that the parameters optimized for classification are identical to the ones optimized for the within class likelihood when the actual distribution of the data is gaussian. The derivative is divided into the parts from each class,

$$\begin{aligned}\frac{\partial E}{\partial u_c} &= \sum_{n=1}^{N}\left(p(c\mid \boldsymbol{x}_n)-\delta_{c_n,c}\right)\frac{\partial Q}{\partial u_c}\\ &= \sum_{c'=1}^{C}\sum_{n\in N_{c'}}\left(p(c\mid \boldsymbol{x}_n)-\delta_{c',c}\right)\frac{\partial Q}{\partial u_c}\end{aligned} \tag{5.2.15}$$

When the actual distribution of the data is known the inner sum can be exchanged with an integration over the distribution,

$$\frac{\partial E}{\partial u_c} = \sum_{c'=1}^{C}\int\left(p(c\mid \boldsymbol{x})-\delta_{c',c}\right)\frac{\partial Q}{\partial u_c}\,p(\boldsymbol{x}\mid c')\,p(c')\,d\boldsymbol{x} \tag{5.2.16}$$

by using Bayes' theorem this can be rewritten,

$$\begin{aligned}\frac{\partial E}{\partial u_c} &= \int\sum_{c'=1}^{C}\left(p(c\mid \boldsymbol{x})-\delta_{c',c}\right)\frac{\partial Q}{\partial u_c}\,p(c'\mid \boldsymbol{x})\,p(\boldsymbol{x})\,d\boldsymbol{x}\\ &= \int p(\boldsymbol{x})\frac{\partial Q}{\partial u_c}\sum_{c'=1}^{C}\left(p(c\mid \boldsymbol{x})-\delta_{c',c}\right)p(c'\mid \boldsymbol{x})\,d\boldsymbol{x}\\ &= \int p(\boldsymbol{x})\frac{\partial Q}{\partial u_c}\left(\sum_{c'=1}^{C}p(c\mid \boldsymbol{x})\,p(c'\mid \boldsymbol{x})-p(c\mid \boldsymbol{x})\right)d\boldsymbol{x}\\ &= \int p(\boldsymbol{x})\frac{\partial Q}{\partial u_c}\,p(c\mid \boldsymbol{x})\left(\sum_{c'=1}^{C}p(c'\mid \boldsymbol{x})-1\right)d\boldsymbol{x}=0\end{aligned} \tag{5.2.17}$$

This shows that if the data is gaussian the found derivatives are already zero when initializing with the within class estimates of the parameters and thus the parameters are already optimal.
If the data is not gaussian the parameters optimized for within class likelihood is not necessarily optimal for classification, not even when the covariance is limited as in the variations with common or diagonal covariance. This can quite easily be verified by looking at figure 5.2.1 together with the following arguments. Diagonal covariance is the same as the basic algorithm in the one-dimensional case and thus can be improved. If common covariance is used it is the (Mahalanobis) distance to the means that decides the classes and in the simple example the means are not symmetric around 0 as should be the case.

The previous results showed that whenever the actual data is not distributed as the model being used, it does not necessarily result in the maximum likelihood of classification when using the within class maximum likelihood estimates of the parameters. One can of course argue whether it makes sense to use gaussian models if the data is not distributed like that. In this simple example it is difficult to defend the use of a gaussian model, but when data are more gaussian or when some dimensions

are gaussian and some are not it makes perfectly sense to enhance the classification using the newly developed algorithm.

It should be mentioned that if common covariance is used the evaluations of the derivative no longer holds. In the derivations it has been assumed that the covariance in each class is independent. With common covariance instead the following derivative should be used,

$$\frac{\partial E}{\partial \boldsymbol{\Sigma}} = \sum_{n=1}^{N} \frac{\sum_{c'=1}^{C} e^{Q_{c'} - Q_{c_n} + \gamma_{c',c_n}} \left(\frac{\partial Q_{c'}}{\partial \boldsymbol{\Sigma}} - \frac{\partial Q_{c_n}}{\partial \boldsymbol{\Sigma}} \right)}{\sum_{c''=1}^{C} e^{Q_{c''} - Q_{c_n} + \gamma_{c'',c_n}}} = \sum_{n=1}^{N} \sum_{c'=1}^{C} w_{c',n} \left(\frac{\partial Q_{c'}}{\partial \boldsymbol{\Sigma}} - \frac{\partial Q_{c_n}}{\partial \boldsymbol{\Sigma}} \right)$$

$$w_{c',n} = \frac{e^{Q_{c'} - Q_{c_n} + \gamma_{c',c_n}}}{\sum_{c''=1}^{C} e^{Q_{c''} - Q_{c_n} + \gamma_{c'',c_n}}} = p\left(c' \mid \boldsymbol{x}_n\right) \tag{5.2.18}$$

$$\frac{\partial Q_{c'}}{\partial \hat{\boldsymbol{\Sigma}}} - \frac{\partial Q_{c_n}}{\partial \hat{\boldsymbol{\Sigma}}} = \boldsymbol{M} - \frac{1}{2} \boldsymbol{M} \circ \boldsymbol{I}$$

$$\boldsymbol{M} = \hat{\boldsymbol{\Sigma}}^{-1} \left(\left(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_{c'} \right) \left(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_{c'} \right)^T - \left(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_{c_n} \right) \left(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_{c_n} \right)^T \right) \hat{\boldsymbol{\Sigma}}^{-1}$$

5.3 Training and initialization of the new model

In this section the new model will be used on the sound database, and especially the initialization and training of the parameters will be investigated. A ranking of the features will be done in the end of this section.

5.3.1 Training

A first run is performed on the data to get a first impression of how the new procedure works. The basic variation (covariance for each class) and the sequence from the forward selection are used. The parameters are initialized with the within class maximum likelihood estimates.

New model, basic, training 1

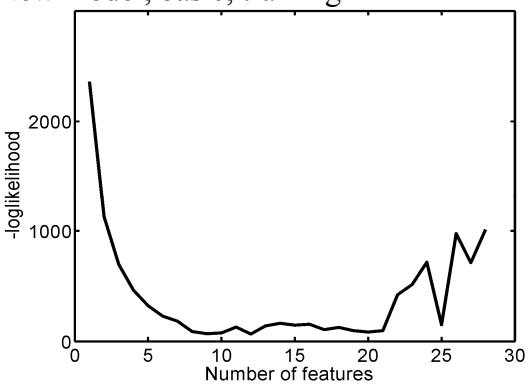


Figure 5.3.1: Training error. Improved error, but still shows increase for increasing dimensions.

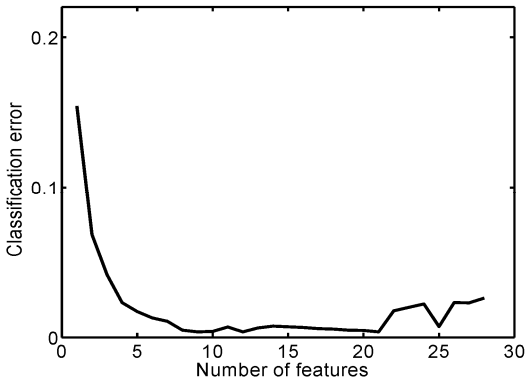


Figure 5.3.2: Classification error.

Although the error has been improved somewhat it still increases for higher numbers of features. This time it has been optimized for classification so this should be impossible.

To ensure the training error does not increase, each time an extra dimension is added, the model is initialized so to give at least the same likelihood as the previous model. This can be done by using the parameters from the previous model. The extra parameters needed for the extra dimension is initialized to be equal for all classes. This implies that the class conditional likelihoods are multiplied by a constant if compared to the previous model. Because the same new parameters are used for all classes, the same constants will be multiplied and they go out of the equation when using Bayes' theorem. When using this procedure the error is ensured not to increase for increasing dimensions. The result is shown in figure 5.3.3,

New model, basic, training 2

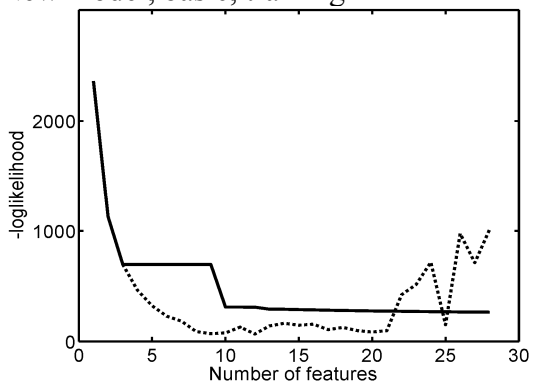


Figure 5.3.3: Training error. No increase for increasing dimensions, but a worse result.

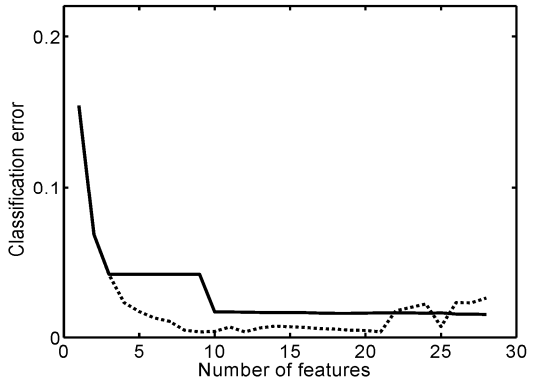


Figure 5.3.4: Classification error.

The error does not increase, but another problem occurs. The error reaches a far worse value than in the previous experiments. It seems the error space as a function of the parameters is quite complex. When the dimension is added the parameters of the previous model are highly optimized and should have small gradients. On the other hand the new parameters are not optimized at all and should have big gradients. This can cause some problems for the algorithm.
In order to get the new parameters up to level they are trained separately first. This is done simply by setting the derivatives of the rest to 0. Afterwards the complete model is trained. The result was the following,

New model, basic, training 3

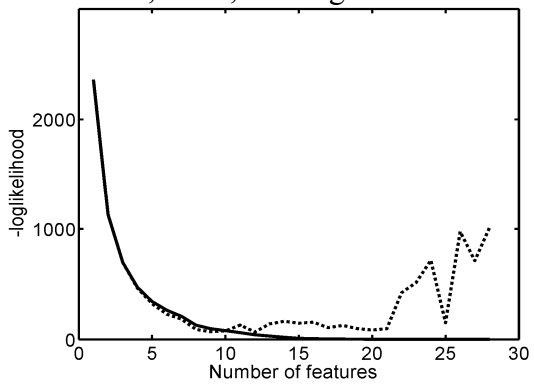


Figure 5.3.5: Training error. Shows good performance. Slightly bigger than method 1 for some dimensions.

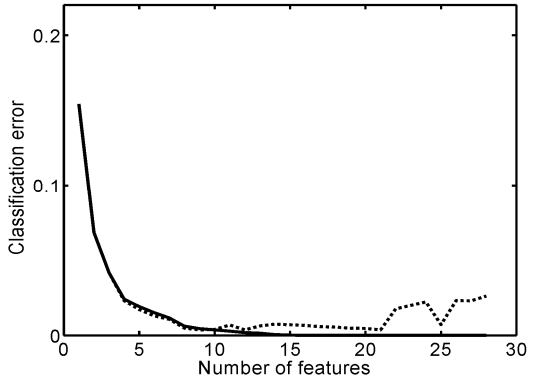


Figure 5.3.6: Classification error.

This looks like quite promising. It can, however, be seen that training method 1 is slightly better with dimensions between 5 and 10. The final algorithm compares the results generated by training method 3 and 1, and chooses the better. The algorithm now gives,

New model, basic, training 4

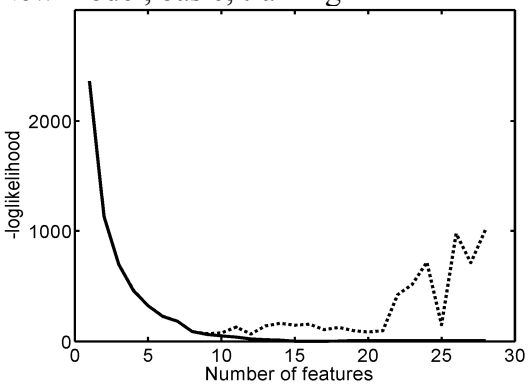


Figure 5.3.7: Training error. The best result obtained.

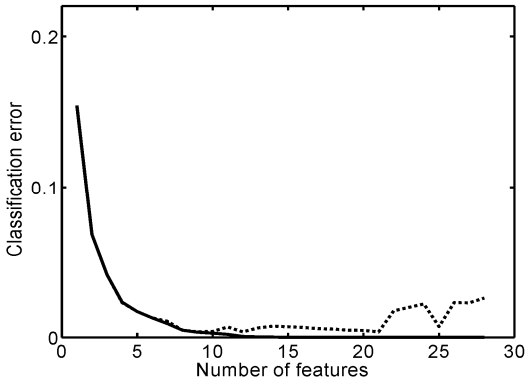


Figure 5.3.8: Classification error.

This is the best achieved in any run. The classification error is on 1.7 % with only 5 features and 0.3 % with 10 features.

The common covariance variation shows only small increase for increasing dimensions when no training was used, but in order to get the best results it was still necessary to use training method 4. Only the result of training method 4 is shown.

New model, common covariance, training 4

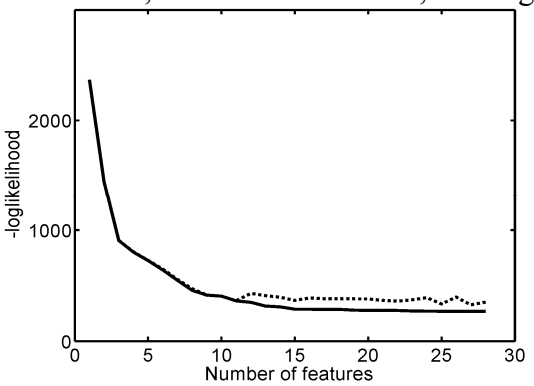


Figure 5.3.9: Training error method 4 in full compared to method 1 in dashed.

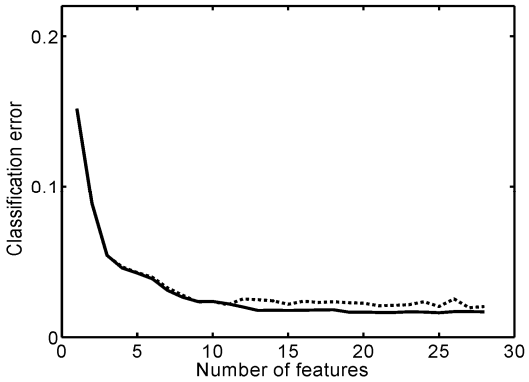


Figure 5.3.10: Classification error.

Finally the variation with diagonal covariance was run. It too had to be trained the advanced way to get the best result,

New model, diagonal covariance, training 4

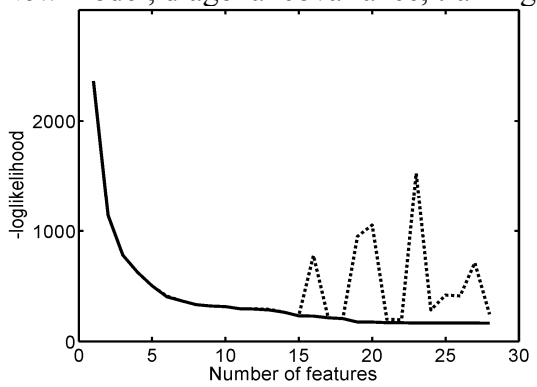


Figure 5.3.11: Training error method 4 in full compared to method 1 in dashed.

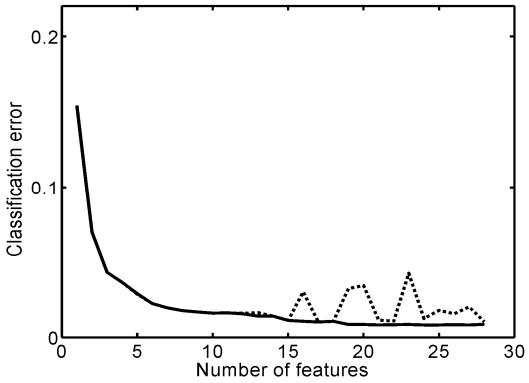


Figure 5.3.12: Classification error.

When training this way it is certified that the training is decreasing for increasing dimensions. Further more the error of the model with the most parameters has smaller error than those with fewer parameters as was initially expected.

5.3.2 Initialization

It seems that the initialization is quite important for how well the parameters can be found. To investigate the dependence further, randomization in the initialization were examined. This showed some difficulties though. The mean is not restricted in any way, but randomization of these parameters did not show any differences in the results and it must be concluded that the problem lies with the covariance matrices. The parameters of the covariance matrix are not free. First the matrix must be symmetric, but this is not a real restriction as it only decreases the number of parameters and not the values of them. Second the covariance matrix must be positive definite. This is a bigger issue. Also there were some numerical issues. When the Mahalanobis distance from the sample point to the mean becomes bigger there is a danger that the likelihood value is so small that it numerically is rounded to 0. If this happens for all three classes there is no way of assigning the point. In high dimensions this is more likely to happen because a single point is more and more unlikely. Experiments were run with different ways of assuring a positive covariance, but no good results were obtained. Either the different randomizations differed so much that it made no sense at all or the algorithm found its way back to the same minimum each time.

A main issue for the model is the possibility of getting invalid model parameters. This happens when the covariance becomes nonpositive definite. This is a problem that is not treated in the update equations. It seems that the gradient can actually be pointing in the direction of these values and thus halt the line search for the other dimensions as well. Error surfaces plotted together with the gradients supported this perception. The means can take any real value and cannot cause any problems. An approach where the dimensions will be trained by themselves will be tried. It trains a single dimension of the covariance at a time and the mean each time. The result is shown below, the result from training method 4 in dashed,

Training of basic 5

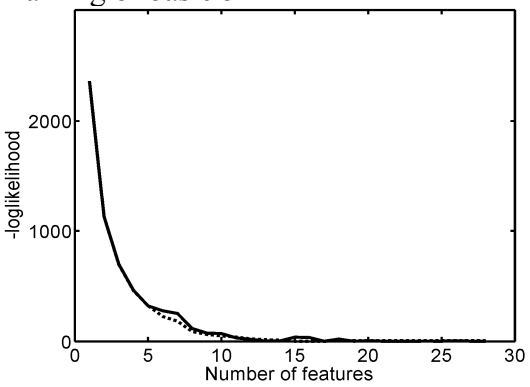


Figure 5.3.13: Training error. Method 5 in full, method 4 in dashed. Method 5 works nearly as well as method 4.

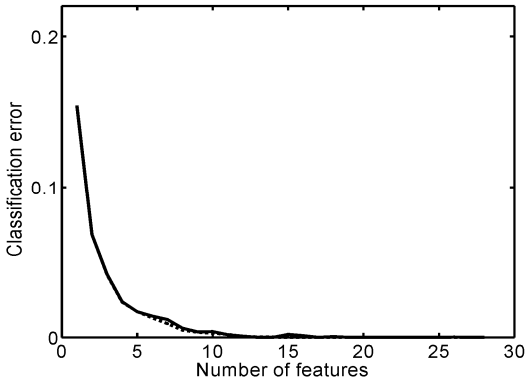


Figure 5.3.14: Classification error.

The results are not quite as good as training method 4, but it is close and more importantly it performs a lot faster. This is necessary when doing the runs with validation sets which will be done later.

5.3.3 Feature ranking

Because the training of the new model is so much slower than that of the original Bayes classifier, the ranking of the features has only been done with the forward selection scheme. The new model was trained using training method 3, which gave the following results,

Basic

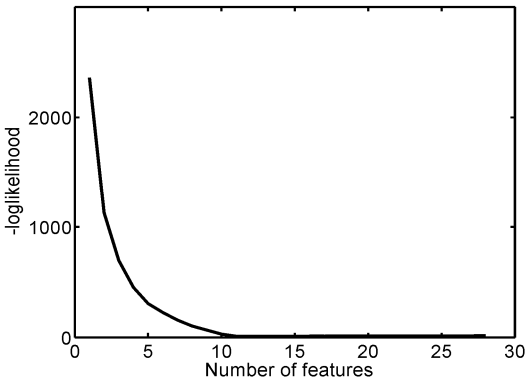


Figure 5.3.15: Training error of basic variation of new model. After about 10 features an optimal result is obtained.

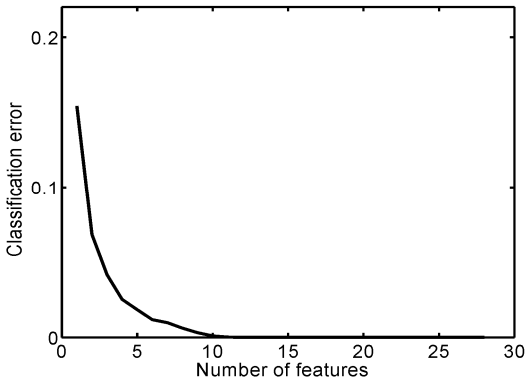


Figure 5.3.16: Classification error. After 10 features no classification error is present.

Common

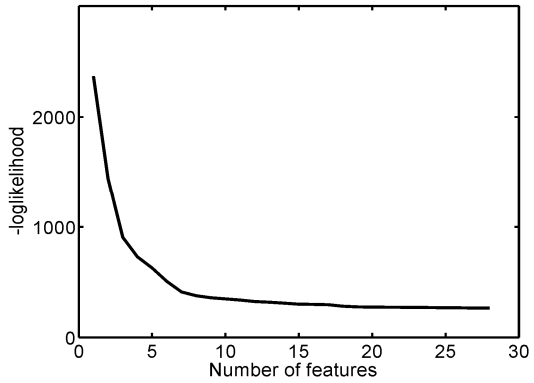


Figure 5.3.17: Training error of common variation of the new model. Error keeps decreasing. Nearly constant after 10 features.

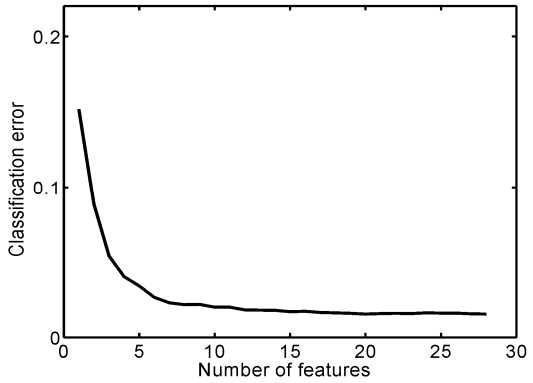


Figure 5.3.18: Classification error. Nearly constant after 10 features.

Diagonal

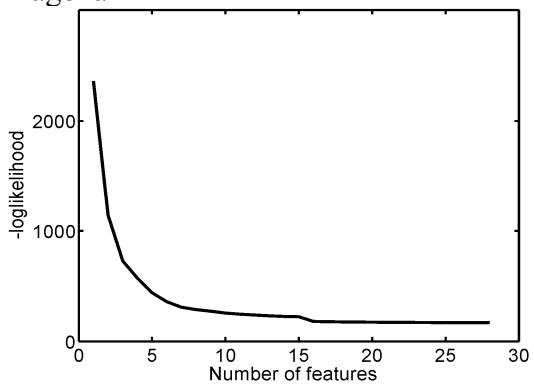


Figure 5.3.19: Training error of diagonal variation of the new model. Nearly constant after 15 features.

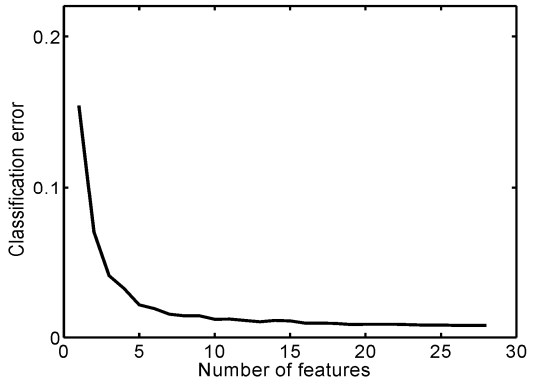


Figure 5.3.20: Classification error. Nearly constant after 10 features.

The ranking of the features were the following,

Forward selection	
Basic	28 25 4 22 19 15 14 6 12 23 7 11 20 9 21 10 5 27 18 13 24 17 16 8 26 1 2 3
Common	28 18 25 4 15 16 20 26 17 7 1 22 8 12 11 5 2 13 19 6 3 23 14 10 24 21 27 9
Diagonal	28 25 4 15 6 16 22 1 20 19 21 7 13 23 8 5 2 3 9 12 27 14 24 18 10 11 17 26

When compared to the rankings achieved for the original Bayes classifier it agrees in many of the selected features. Because of the increased training time, the verification of the ranking is not possible within reasonable time. Because the verification of the previous rankings showed very little discrepancy, these results are considered to be good approximations of the optimal subsets as well.

5.4 Generative vs. discriminative models

The new model retrieved in the two previous sections has to do with a known problem in statistics. It has to do with the difference between generative (informative) models and discriminative models. The generative model tries to describe the classes. The name generative is used because with this kind of model it is possible to generate synthetic data once the model is obtained, and informative is used because it gives information about the data. The discriminative does not describe the data. The only thing this kind of model worries about is to discriminate between the classes so they can be classified correctly [Ng, 2002]. Even though the two families of models are different, they exist in pairs. An example is given next, and some characteristics of the model families in general are given and the new model is set in relation to them. Finally, the equivalent discriminant models are given for the three variations of the covariance of the generative (Bayes) model.

5.4.1 Example of a generative discriminative pair

If a classification problem consisted of two gaussian classes with different means and the same covariance matrix the generative model would find the mean of the classes and the joined covariance by the following equations.

$$\begin{aligned} \hat{\boldsymbol{\mu}}_1 &= \frac{1}{N_1}\sum_{n|c_n=1}\boldsymbol{x}_n \quad , \quad \hat{\boldsymbol{\mu}}_2 = \frac{1}{N_2}\sum_{n|c_n=2}\boldsymbol{x}_n \\ \hat{\boldsymbol{\Sigma}} &= \frac{1}{N_1+N_2-2}\left(\sum_{n|c_n=1}(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_1)(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_1)^T + \sum_{n|c_n=2}(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_2)(\boldsymbol{x}_n-\hat{\boldsymbol{\mu}}_2)^T\right) \end{aligned} \tag{5.4.1}$$

Then it would use the Bayes criterion to find the posterior likelihoods like this,

$$p(c_1 \mid x) = \frac{p(x \mid c_1)\,p(c_1)}{p(x \mid c_1)\,p(c_1) + p(x \mid c_2)\,p(c_2)} \tag{5.4.2}$$

and choose the class with the biggest posterior. The posterior likelihoods of the two classes share the denominator and thus it can be left out. A discriminant is formed,

$$\lambda = \frac{p(c_1 \mid \boldsymbol{x})}{p(c_2 \mid \boldsymbol{x})} = \frac{p(x \mid c_1)\,p(c_1)}{p(x \mid c_2)\,p(c_2)} \tag{5.4.3}$$

If the discriminant is larger than one, class 1 is chosen, class 2 is chosen otherwise. The logarithm of the discriminant is as good a discriminant because the logarithm is monotically increasing and the discriminant is always larger than 0. The logarithm of equation (5.4.3) together with the gaussian distributions with common covariance gives,

$$\begin{aligned}\log\lambda &= \log\left(p(\boldsymbol{x}\mid c_1)\right)-\log\left(p(\boldsymbol{x}\mid c_2)\right)+\log\left(\frac{p(c_1)}{p(c_2)}\right)\\ &= -\frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_1)^T\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_1)+\frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_2)^T\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_2)+\log\left(\frac{p(c_1)}{p(c_2)}\right)\\ &= \hat{\boldsymbol{\mu}}_1^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_1+\hat{\boldsymbol{\mu}}_2^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_2-2\hat{\boldsymbol{\mu}}_1\hat{\boldsymbol{\Sigma}}^{-1}\boldsymbol{x}-2\hat{\boldsymbol{\mu}}_2\hat{\boldsymbol{\Sigma}}^{-1}\boldsymbol{x}+\log\left(\frac{p(c_1)}{p(c_2)}\right) \qquad (5.4.4)\\ &= \hat{\boldsymbol{\mu}}_1^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_1+\hat{\boldsymbol{\mu}}_2^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_2+\log\left(\frac{p(c_1)}{p(c_2)}\right)-\left(2\hat{\boldsymbol{\mu}}_1\hat{\boldsymbol{\Sigma}}^{-1}+2\hat{\boldsymbol{\mu}}_2\hat{\boldsymbol{\Sigma}}^{-1}\right)\boldsymbol{x}\\ &= \beta_0+\boldsymbol{\beta}_1^T\boldsymbol{x}\end{aligned}$$

As can be seen, this is a simple linear equation that if it is larger than 0, class 1 is chosen and class 2 is chosen otherwise. In the generative model the constants β_0 and $\boldsymbol{\beta}_1$ are found through the means and the joined covariance,

$$\beta_0 = \hat{\boldsymbol{\mu}}_1^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_1+\hat{\boldsymbol{\mu}}_2^T\hat{\boldsymbol{\Sigma}}^{-1}\hat{\boldsymbol{\mu}}_2+\log\left(\frac{p(c_1)}{p(c_2)}\right),\quad \boldsymbol{\beta}_1 = -\left(2\hat{\boldsymbol{\mu}}_1\hat{\boldsymbol{\Sigma}}^{-1}+2\hat{\boldsymbol{\mu}}_2\hat{\boldsymbol{\Sigma}}^{-1}\right) \qquad (5.4.5)$$

The two constants β_0 and $\boldsymbol{\beta}_1$ in the linear discriminant can be found directly without worrying about means and covariances. It can be done by training the discriminant directly. This is what separates the discriminative from the generative model. The generative finds the distributions of each class and through them finds the linear constants, whereas the discriminative finds the constants directly. For a two class problem of dimension d, the generative model has $2+2d+d^2$ variables whereas the discriminative model can do the same classification with only $1+d^2$. Further more the discriminative model does not assume that the data is gaussian distributed, but simply finds the best linear separation related to an error function. A way of training the discriminative model is to use ± 1 for target classes and use a simple least squares error function [Ruck, 1990].

5.4.2 Characteristics of generative and discriminative models

Many papers have been written about whether the discriminative or the generative approach should be used [Ng, 2002], [Bouchard, 2003]. It seems that in many cases the discriminative model gives the best separation [Bach, 2005], but the opposite is also claimed [Efron, 1975]. The differences in opinion come largely from different assumptions and the general characteristics will be summarized in the following.

One argument in favour of the discriminative model is that something that is in fact not needed to solve the problem should not be modelled. Only the posterior likelihoods or the decision boundaries are needed and therefore it is a detour to find the class distributions first.

One thing that separates the problem in two is if the model assumption of the generative models is correct or not. If the class conditional model does not correspond to the distribution of the training data, the performance of the generative models can be severely affected. This was shown in section 5.2. This is to the advantage of the discriminative models as they do not have a corresponding assumption and works equally well on all distributions [Rubinstein, 1997]. This means the discriminative models are more robust to variations in the distribution of the data.

The asymptotic behaviour of the two models as the number of samples goes to infinity is interesting. The asymptotic test error of the discriminative models is always lower than or equal to that of the generative models [Ng, 2002], and this is why many problems show better performance in the discriminative case. Often the generative models reach its asymptote quicker than the discriminative models, and thus, even though being asymptotical worse, they can give better performance when limited samples are available [Rubinstein, 1997]. It is visualized in figure 5.4.1.

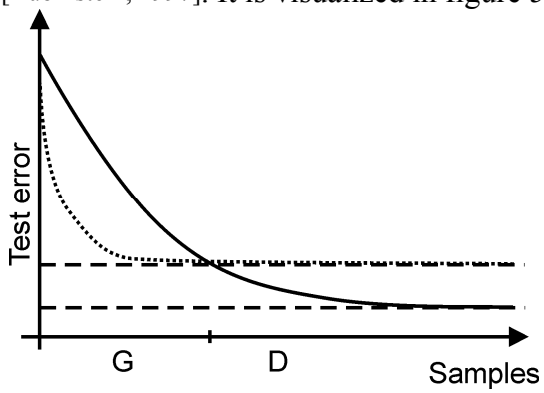


Figure 5.4.1: The asymptotic behaviour of generative models, dotted, and discriminative models, full. On the x axis regions of preferred models are marked.

A feature of the generative models is that it gives a likelihood value as well as the classification. This is good when a measure of the reliability of the classification is needed and also makes it possible to change the misclassification cost function after the model has been trained. The discriminative models do not give a likelihood directly, but post processing can be used to obtain likelihoods as in logistic regression. The likelihood from the generative models is more reliable though because they estimate the distributions of the data [Abou-Moustafa, 2004]. When post processing is used the training of the discriminative models become nonlinear and an iterative approach must be used instead.

The training of the generative models is straightforward. Only a single sweep through the data is necessary and it is only done within the classes. This means classes can be trained independently of other classes. The discriminative models trains over all classes at once, and, when nonlinear post processing is used, must be done in an iterative fashion that is much more computationally demanding.

The discriminative models only discriminate between two models. This means that if more classes exist, post processing is necessary again. This can be done in different ways. The logistic regression used on multiple classes gives the softmax function. Another approach is to form a tree of two class models [Frank, 2004]. The tree approach will be used when comparing performance with the new model. The generative model handles any number of classes directly.

5.4.3 New model in relation to generative and discriminative models

What is done in the new model cannot be characterized specifically as generative or discriminative. The reason why it is not generative is that it does not describe the data any longer. As was seen in the simple example it was not the best gaussian approximation that was found, and if used for generating data they will not be any like

the original data. Because it is not the discriminant that is trained and the parameters of the generative model are maintained, it can not be called discriminative either. It can be called training a generative model discriminatively.

5.4.4 Logistic regression and quadratic discriminant

As mentioned previously, post processing can be used to make the discriminant model return likelihoods instead of only classifications. A function that transforms from the likelihood interval, 0 to 1, to the linear interval of $\pm\infty$ is the logit function [Komarek, 2004, chap. 4],

$$\operatorname{logit}\left(p(x)\right)=\log\frac{p(x)}{1-p(x)}\cong\beta_0+\boldsymbol{\beta}_1^T\boldsymbol{x} \tag{5.4.6}$$

which is then modelled by the linear discriminant as indicated by the approximation sign in equation (5.4.6). Using the logit means post processing of the outputs of the linear network must be done to get the likelihood,

$$p(x)=\frac{e^{\beta_0+\boldsymbol{\beta}_1^T\boldsymbol{x}}}{1+e^{\beta_0+\boldsymbol{\beta}_1^T\boldsymbol{x}}} \tag{5.4.7}$$

This gives a nonlinearity in the model and thus makes a linear solution impossible. Instead an iterative gradient approach is used, which makes the training more complicated than the generative model. Models using the logit are called logistic regression and the training of these models, as well as the linear models, is assured to converge to the global minimum with a proper training algorithm [Hastie, 1991, chap. 6].

The three variations of the gaussian model all have discriminative counterparts. For the variation with common covariance the result was a linear discriminant as found in equation (5.4.4).
For the basic variation the discriminant is,

$$\begin{aligned}
\log\lambda &= \log\left(p(\boldsymbol{x}\mid c_1)\right)-\log\left(p(\boldsymbol{x}\mid c_2)\right)+\log\left(\frac{p(c_1)}{p(c_2)}\right)\\
&= -\frac{1}{2}\log\left(\left|\hat{\boldsymbol{\Sigma}}_1\right|\right)-\frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_1)^T\,\hat{\boldsymbol{\Sigma}}_1^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_1)\\
&\quad+\frac{1}{2}\log\left(\left|\hat{\boldsymbol{\Sigma}}_2\right|\right)+\frac{1}{2}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_2)^T\,\hat{\boldsymbol{\Sigma}}_2^{-1}(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_2)+\log\left(\frac{p(c_1)}{p(c_2)}\right)\\
&= \frac{1}{2}\log\left(\frac{\left|\hat{\boldsymbol{\Sigma}}_2\right|}{\left|\hat{\boldsymbol{\Sigma}}_1\right|}\right)-\frac{1}{2}\hat{\boldsymbol{\mu}}_1^T\hat{\boldsymbol{\Sigma}}_1^{-1}\hat{\boldsymbol{\mu}}_1+\frac{1}{2}\hat{\boldsymbol{\mu}}_2^T\hat{\boldsymbol{\Sigma}}_2^{-1}\hat{\boldsymbol{\mu}}_2+\log\left(\frac{p(c_1)}{p(c_2)}\right)+\\
&\quad\left(\hat{\boldsymbol{\mu}}_1\hat{\boldsymbol{\Sigma}}_1^{-1}-\hat{\boldsymbol{\mu}}_2\hat{\boldsymbol{\Sigma}}_2^{-1}\right)^T\boldsymbol{x}+\frac{1}{2}\boldsymbol{x}^T\left(\hat{\boldsymbol{\Sigma}}_2^{-1}-\hat{\boldsymbol{\Sigma}}_1^{-1}\right)\boldsymbol{x}\\
&= \beta_0+\boldsymbol{\beta}_1^T\boldsymbol{x}+\boldsymbol{x}^T\boldsymbol{B}_2\boldsymbol{x}
\end{aligned} \tag{5.4.8}$$

This is a quadratic discriminant.
For the diagonal case a diagonal matrix can be put in the equation above. The constant and the linear term do not change, but the quadratic term simplifies to a diagonal matrix. This means that the discriminant becomes,

$$\log\lambda=\beta_0+\boldsymbol{\beta}_1^T\boldsymbol{x}+\boldsymbol{\beta}_2^T(\boldsymbol{x}\circ\boldsymbol{x}) \tag{5.4.9}$$

$\circ$ is the element wise (Hadamard) product.

The logistic regression is only defined for linear weights, but as can be seen from the equations above it is only the inputs that are quadratic and the problem remains linear. This means the quadratic terms are simply added as additional inputs.
As this is not the main subject of this project a prewritten method of a logistic regression model is found in the NetLab package [6] which uses the iteratively reweighted least squares algorithm for training which assures convergence.

5.5 Comparison of new model against generative and discriminative models

The classification results using the new model were quite good with a very high classification rate, but a more interesting thing is how the new model compares to the existing models. The new model is a mixture of two methods, the generative and the discriminative, therefore it seems appropriate to compare against these two. The equivalent models were found in the previous sections. The generative model does not use training and the discriminative model was assured to converge to a global minimum, so no further work has to be done in the training of the parameters in these two algorithms to make the comparison fair. For the new model training method 4 is used together with the feature sequence from the ranking of the new model.

Basic

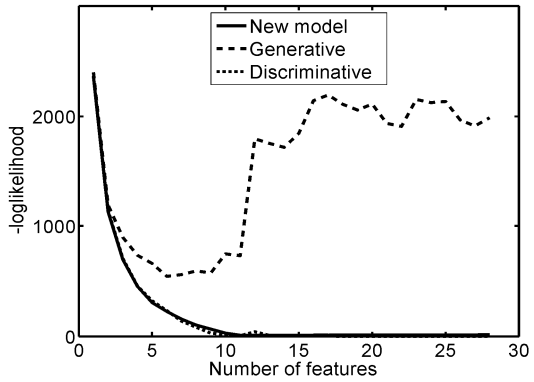


Figure 5.5.1: Training error. The new model and the discriminative model are very close to each other. The generative is clearly worse.

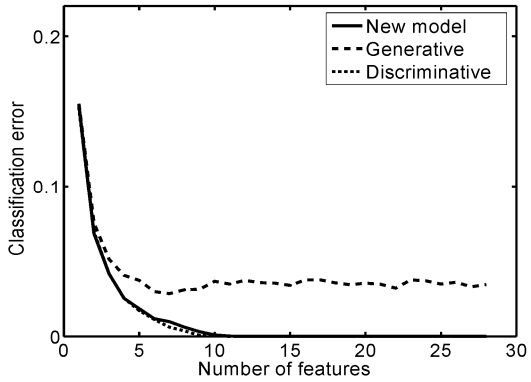


Figure 5.5.2: Classification error.

Common covariance

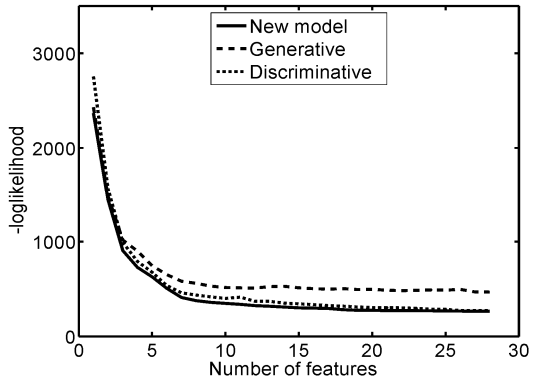


Figure 5.5.3: Training error. The same trend appears here. The new model and the discriminative models are close and the generative is the worst.

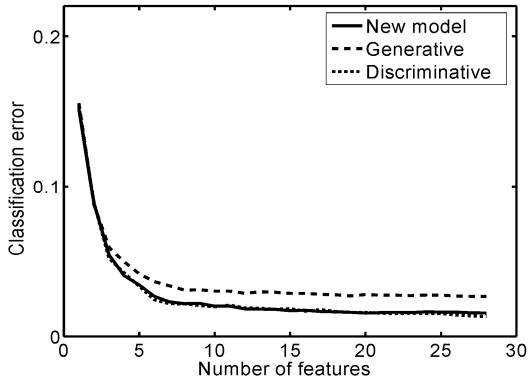


Figure 5.5.4: Classification error.

Diagonal covariance

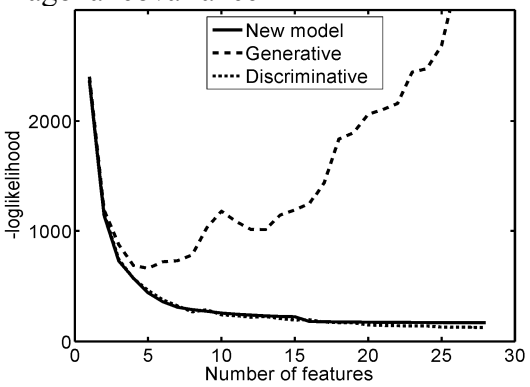


Figure 5.5.5: Training error. Again the generative is clearly worse than the two others.

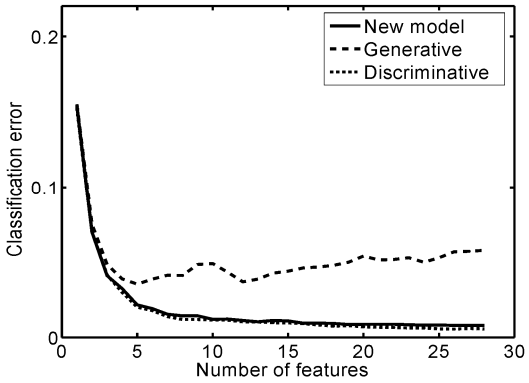


Figure 5.5.6: Classification error.

The new algorithm performs quite well. The generative model is clearly worse than both others in all variations. It is close with common covariance though. The discriminative and the new model perform very similar, sometimes the discriminative is slightly better, other times the new model is best. The similar classification performance will be investigated further in chapter 6.

5.6 Feature selection using the new model

In a previous section it is mentioned that training a model consists of at least two steps. The first is the training of the model parameters. This has been covered in the previous sections and a comparison has also been made. The other step of training involves choosing the dimension of the model. The dimension is chosen to optimize generalizability. This means the model's ability to classify new points rather than the points in the training set. The classes of the training set are already known, so classification of the training set is not the objective of the algorithm. The goal is the ability to classify new points with unknown classes. The more dimensions, the better the classification is still true in theory, also for classifying new points, but the requirements of the training set increases as more dimensions are included. For a training set to be large enough, enough points must be present in all parts of feature space. When the feature space increases so must the training set size. This is called the curse of dimensionality. If the training set is not large enough there is a risk of overfitting the model to the training set and loosing generalizability of the model. For a fixed size of training set, as is the case here, it means that a large number of dimensions are not necessarily desirable.
A number of different ways exist to find the point where overfitting occurs, and two of them will be used here. One uses another set of known points to verify the performance against, the other way is based on information criteria. Two of the variations of the information criteria will be used, the Akaike's Information Criteria and the Bayes Information Criteria. In the end a final model is suggested based on the results.

5.6.1 Validation set

The overfitting of the training data and the ability to classify new points must be tested. An obvious way of doing this is to test the found algorithm on a set of new points. The new points are usually found by dividing the training set in half into a validation set and a training set. The model is trained using the training set and then a validation error is calculated using the model on the validation set. The model can not overfit the validation set because it is not part of the training.
The training error keeps decreasing for increasing dimensions. The validation error on the other hand will usually decrease at first like the training error, but when the model starts to overfit the training data, the validation error will increase. Right before this point the optimal dimension is found. When this point has been identified, the complete database is used to find the optimal parameters.
Both the training set and the validation set must be of a certain size in order to make the results valid. This causes a problem when only a limited set of samples is available, and it might not be enough if only half of the points can be used for training. In these cases a leave-1-out [Bishop, 2004, chap. 9] approach can be used. This approach works by taking only a single sample out the training set, training the model and calculate the validation error of the single sample. This is repeated with all samples of the training set being taken out once. The validation error is the sum of all the validation errors found. This way you get a maximum number of training samples while still having a large validation set. This of course has the disadvantage of training the model over and over again which can take some time.
The leave-1-out approach is hard to use in this case for two reasons. The first reason is the obvious time it consumes. If the test is to give a valid result it has to be retrained

independently for each new test. Should the model find the best parameters, it needs to use training method 5 which consumes a lot of time. The other reason has to do with the independency of the samples. To make the validation set valid, it must be independent of the training set. The features are based on overlapping windows which means that the points are highly correlated with each other. The database consists of 30 s samples of sound, but some of the samples are taken from the same song or sentence. This again causes another level of correlation of the points in the training set.

The clips appear in order in the training set. This means that for the same song the clips appear in series in the training set. To save time more than one sample is taken out in each iteration. To lose some of the correlation the validation set is taken out as a series of samples instead of random samples all over the training set. This means that correlation only exists between the end points of the series and is minimized in the remaining validation set. 3 times 3 clips of 30 s are taken out in each iteration, 3 from each class. This makes a compromise of training set size and computation time. It should be mentioned though that clips from the same song will still occur in both sets, and the results must be considered with care. The plot looks like this,

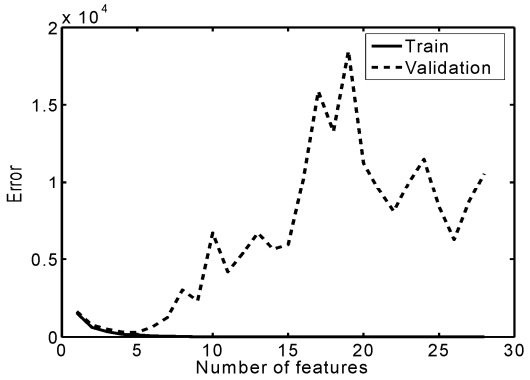


Figure 5.6.1: Validation and training error of the basic variation of the new model.

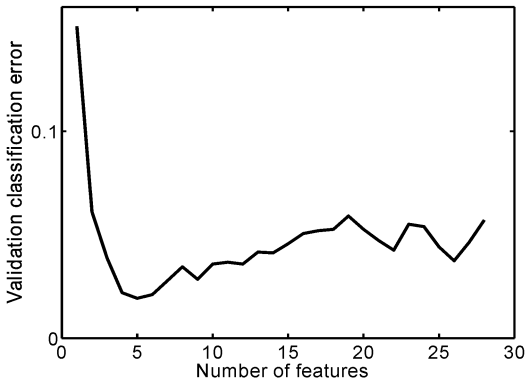


Figure 5.6.2: Validation classification error of the basic variation of the new model.

The validation error increases dramatically when more than 5 features are used. Based on this the point of overfitting is of course 5 features, but the increase in validation error is too big and the validation set too dependent for the result to be trusted on its own. Fortunately another method exist which will be presented in the next subsections.

5.6.2 Bayes factors

When multiple models exist for the same problem it is necessary to find the best one. The best one meaning the model that is most likely to describe the problem [Schwarz, 1978], [Kass, 1995]. In the case of two competing models this can be written with the use of Bayes as,

$$p(H_k \mid \boldsymbol{X}, \boldsymbol{c}) = \frac{p(\boldsymbol{c} \mid H_k, \boldsymbol{X})\, p(H_k, \boldsymbol{X})}{p(\boldsymbol{c} \mid H_1, \boldsymbol{X})\, p(H_1, \boldsymbol{X}) + p(\boldsymbol{c} \mid H_2, \boldsymbol{X})\, p(H_2, \boldsymbol{X})} \tag{5.6.1}$$

H_1 and H_2 are the two models, $\boldsymbol{X}$ is the training data and $\boldsymbol{c}$ the target classes. The odds in favour of model H_1 can be written as the ration between the two probabilities,

$$\frac{p(H_1 \mid \boldsymbol{X}, \boldsymbol{c})}{p(H_2 \mid \boldsymbol{X}, \boldsymbol{c})} = \frac{p(\boldsymbol{c} \mid H_1, \boldsymbol{X})}{p(\boldsymbol{c} \mid H_2, \boldsymbol{X})} \frac{p(H_1, \boldsymbol{X})}{p(H_2, \boldsymbol{X})} \tag{5.6.2}$$

The posterior odds are equal to the prior odds times a constant. This constant is what is called the Bayes factor and is given by,

$$B_{12} = \frac{p(\boldsymbol{c} \mid H_1, \boldsymbol{X})}{p(\boldsymbol{c} \mid H_2, \boldsymbol{X})} \tag{5.6.3}$$

If no prior knowledge of the problem exists and the prior probabilities are set equal, the Bayes factor gives the posterior odds in favour of a given model directly. Often models are compared to a reference model which is then called H_0.
If the model depends on a set of parameters, which they often do, the likelihood used is not simply the maximum likelihood estimate, but should be a marginal likelihood which means an integration on the parameter space has to be performed. The integration has the form,

$$p(\boldsymbol{c} \mid \boldsymbol{X}, H_k) = \int p(\boldsymbol{c} \mid \boldsymbol{X}, H_k, \boldsymbol{\theta}_k) \cdot \pi(\boldsymbol{\theta}_k \mid \boldsymbol{X}, H_k)\, d\boldsymbol{\theta}_k \tag{5.6.4}$$

π is a prior distribution of the model parameters which form the vector, $\boldsymbol{\theta}$. The prior distribution can be very hard to give and many different schemes for evaluating the integral have been suggested.
The Schwarz criterion is given by,

$$S_{12} = \log p\left(\boldsymbol{c} \mid \boldsymbol{X}, H_1, \hat{\boldsymbol{\theta}}_1\right) - \log p\left(\boldsymbol{c} \mid \boldsymbol{X}, H_2, \hat{\boldsymbol{\theta}}_2\right) - \frac{1}{2}(d_1 - d_2)\log(n) \tag{5.6.5}$$

where $\hat{\boldsymbol{\theta}}_k$ is the maximum likelihood estimate of H_k, and can be viewed as an approximation of the logarithm of the Bayes factor because it satisfies [Kass, 1995],

$$\frac{S_{12} - \log(B_{12})}{\log(B_{12})} \to 0 \quad , \quad n \to \infty \tag{5.6.6}$$

This gives an approximation of the Bayes factor with the likelihoods already calculated. When comparing more than one model S_{12} can be divided into two factors representing each model,

$$\begin{aligned} S_{12} &= S_1 - S_2 \\ S_k &= \log p\left(\boldsymbol{c} \mid \boldsymbol{X}, H_k, \hat{\boldsymbol{\theta}}_k\right) - \frac{1}{2} d_k \log(n) \end{aligned} \tag{5.6.7}$$

The model with the largest S_k will then be the preferred model. When S_k is multiplied with minus two the Bayes Information Criterion is found,

$$BIC = -2 \cdot \log p\left(\boldsymbol{c} \mid \boldsymbol{X}, H_k, \hat{\boldsymbol{\theta}}_k\right) + d_k \log(n) \tag{5.6.8}$$

The model with the smallest *BIC* value is the preferred model.

5.6.3 Akaike's information criterion

Akaike's information criterion [Akaike, 1973] is defined as follows,

$$AIC = -2 \log p\left(\boldsymbol{c} \mid \boldsymbol{X}, H_k, \hat{\boldsymbol{\theta}}_k\right) + 2d_k \tag{5.6.9}$$

It is very similar to *BIC* and the penalty constant differs only by $\frac{1}{2}\log d_k$. The AIC is found in a quite different way though. The derivation is quite advanced and will not be repeated here, but it is based on maximizing the expected log likelihood which is defined by,

$$E_{\boldsymbol{C},\hat{\theta}}\left\{\log p\left(\boldsymbol{c} \mid \hat{\boldsymbol{\theta}}\right)\right\} = E_{\hat{\theta}}\left\{\int_{\boldsymbol{C}} p(\boldsymbol{c} \mid \boldsymbol{\theta}) \log p\left(\boldsymbol{c} \mid \hat{\boldsymbol{\theta}}\right) d\boldsymbol{c}\right\} \tag{5.6.10}$$

where $\hat{\boldsymbol{\theta}}$ are the estimated parameters, $\boldsymbol{\theta}$ are the true parameters and E_x is the expected value with respect to the distribution of x. It means that the *AIC* finds the

model that maximizes the likelihood over all possible estimated parameter sets on all possible test sets, i.e. the model that minimizes the generalization error. *BIC* finds the most probable model for a given training set, which is not quite the same.
As can be seen from the equations, *BIC* will tend to favour models of smaller dimension than *AIC* will for training sets bigger than 8 samples.

BIC is dependent on the number of independent samples which goes directly into the equation. To decrease the dependency between points in the training set, no overlap is used in the feature windows which causes the training set to decrease five times. The model has been trained on this set and the *BIC* and *AIC* values have been found. The training error has been doubled for the values to be comparable. The plots look like this,

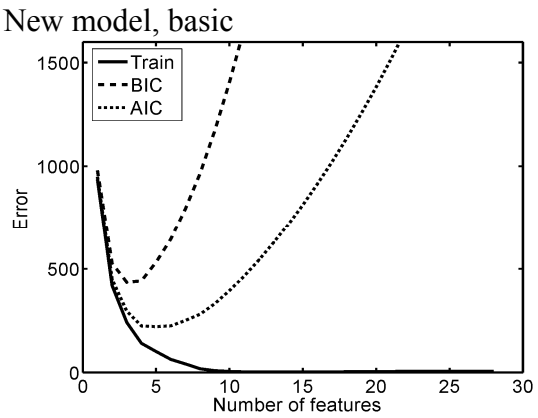


Figure 5.6.3: *BIC* & *AIC* compared to double the training error. *BIC* suggests 3 and *AIC* 5 features.

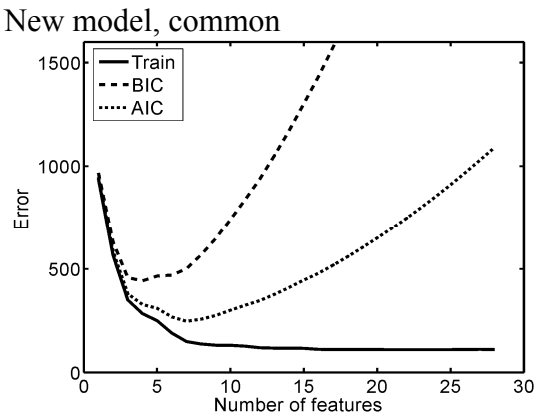


Figure 5.6.4: *BIC* & *AIC* compared to double the training error. *BIC* suggests 4 and *AIC* 7 features.

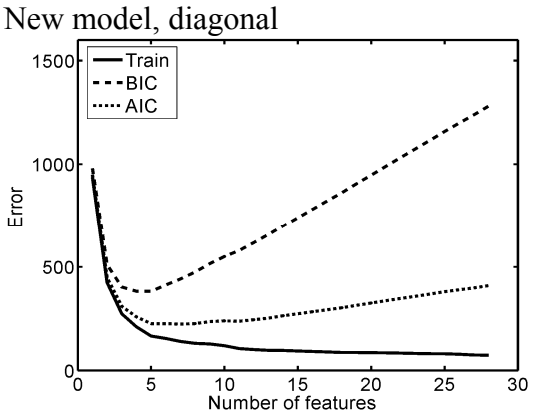


Figure 5.6.5: *BIC* & *AIC* compared to double the training error. *BIC* suggests 5 and *AIC* 7 features.

The *BIC* values favour smaller dimensions than *AIC* as was expected. As no general rule exist of which value to trust, the results of the *AIC* and *BIC* from the basic algorithm are compared to the result of the validation set. In the basic variation, the validation set favours 5 features as does the *AIC*. It could seem that *BIC* is too drastic in its choice of simplicity. This can be explained with the fact that the training set, although better than the full set, still has dependencies between the samples. This means that the penalty is too big for *BIC*. The diagonal minimum *AIC* and the basic

minimum *AIC* is very close to each other with only a small margin favouring the basic variation.

5.6.4 Final model

Based on the *BIC*, *AIC* and the validation set, the basic variation of the new model with only 5 features is chosen. These 5 features are selected using the forward selection scheme and they are,

28 genericReliabilityDev
25 genericToneDistance
4 averageDeviation
22 genericAbsDiff7
19 genericAbsDiff4

When originally investigating the features these was not the ones that would have been chosen, but when looking at the histograms of the features they all show a very good separation. The classification error with the chosen model is 1.8 % which is quite low. Using the validation set it was 1.9 % which indicates that very little overfitting has occurred with this number of features. The final model is investigated in the next chapter.

6 Evaluation of the final model

A final model has been created and in this section the model will be investigated. In the first section the features selected for the model will be presented. In the second section the misclassifications that are still present will be reviewed. This is done by looking at the confusion matrix and by inspecting the sounds that cause the misclassifications. The misclassifications done by logistic regression and the new model are compared. In section three the effects of decreasing the accuracy of the pitch detector is evaluated.

6.1 Investigation of chosen features

The features that were selected for the final model are in ranking order from best to worst,

28 genericReliabilityDev
25 genericToneDistance
4 averageDeviation
22 genericAbsDiff7
19 genericAbsDiff4

The features are presented and described individually below,

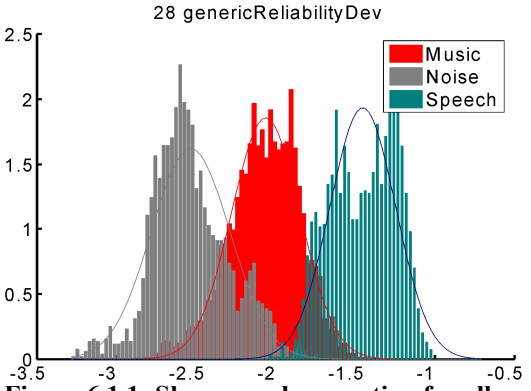


Figure 6.1.1: Shows good separation for all classes.

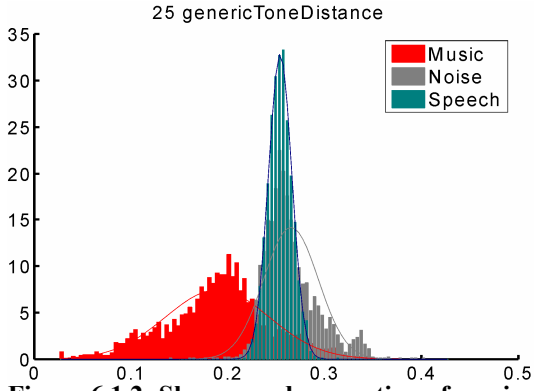


Figure 6.1.2: Shows good separation of music.

Feature 28 clearly shows very good separation of the three classes. The logarithm has been taken of the classes so the values cannot be used directly, but the ordering is directly explainable. Noise is described badly by pitch and therefore constantly has a low value which results in very low deviation in the values. Speech is sometimes described very well by pitch (voiced regions) and sometimes very bad (unvoiced regions). This dual behaviour causes the deviation to be very large. Music is somewhere in between with a quite constant behaviour, but still not as constant as noise. Feature 25 captures music nicely whereas noise and speech are on top of each other. This is as anticipated because this was exactly what the feature was designed for.

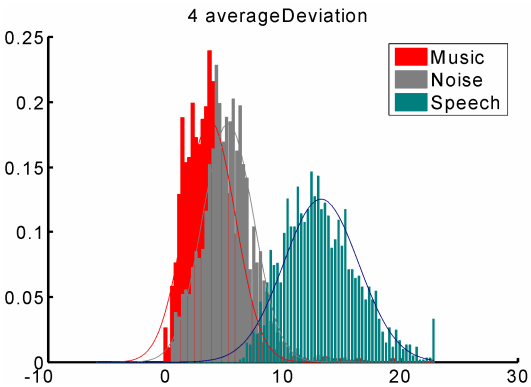


Figure 6.1.3: Shows good separation of speech.

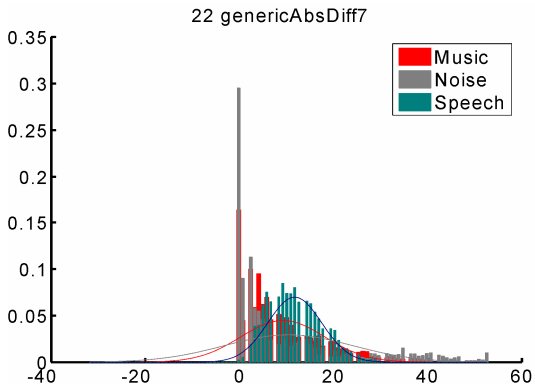


Figure 6.1.4: The reason that this feature is chosen is not apparent.

Feature 4 is the only feature in the final model that is based on the reliable windows. It shows very good separation of the speech which has much slopeness inside the reliable windows. Both music and noise have little slopeness inside the reliable windows, music because the tones are quite constant and noise because the windows are so small that the pitch can not vary much. It is not quite clear why feature 22 is chosen. Even though the classes are distinguishable, they are on top of each other.

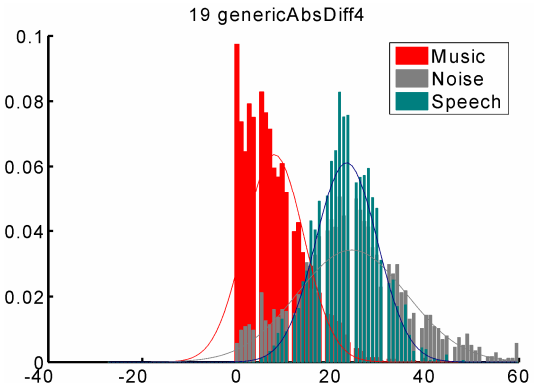


Figure 6.1.5: Good separation of music.

Feature 19 shows good separation of the music. This feature is one of the histogram bins of the differences. Bin four covers the interval 8-16. Speech probably has a large value here because it covers the steps in the normal speech variation of the pitch. Why noise has the same value as speech is not directly explainable.

3 D scatter plots are made to show the separation of the classes.

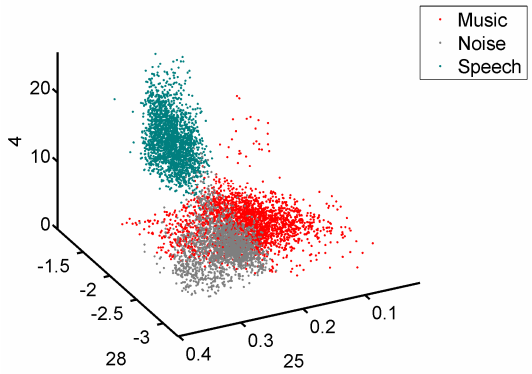


Figure 6.1.6: 3 D scatter plot of the three best features. Speech is obviously better separated than music and noise.

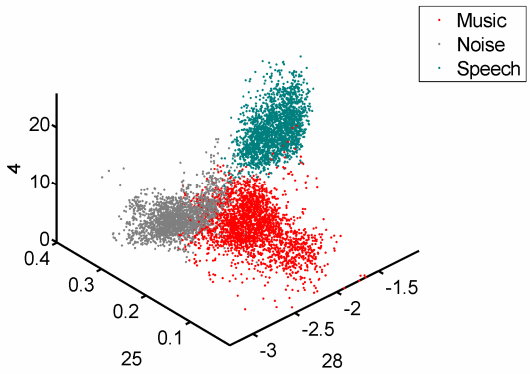


Figure 6.1.7: Same data as on the left, but from another angle. Noise and music are mixed.

The classes are clearly different in placement. The speech only has very little overlapping with the two other classes, while music and noise is more entangled. This is not surprising. The music and noise covers a lot of genres, and the classes contain very different sound clips. Speech is much more clearly distinguished. Even if the classifications were done manually, it would probably be easier to classify the speech than noise and music.
Plots of the remaining feature combinations are in appendix.

The model distributions and class boundaries can be seen in the next plots. The distribution is found by taking the feature specific entry from the complete covariance

and mean. The decision boundaries are found by using this distribution and therefore the misclassified point can be inside the decision boundaries of the plot.

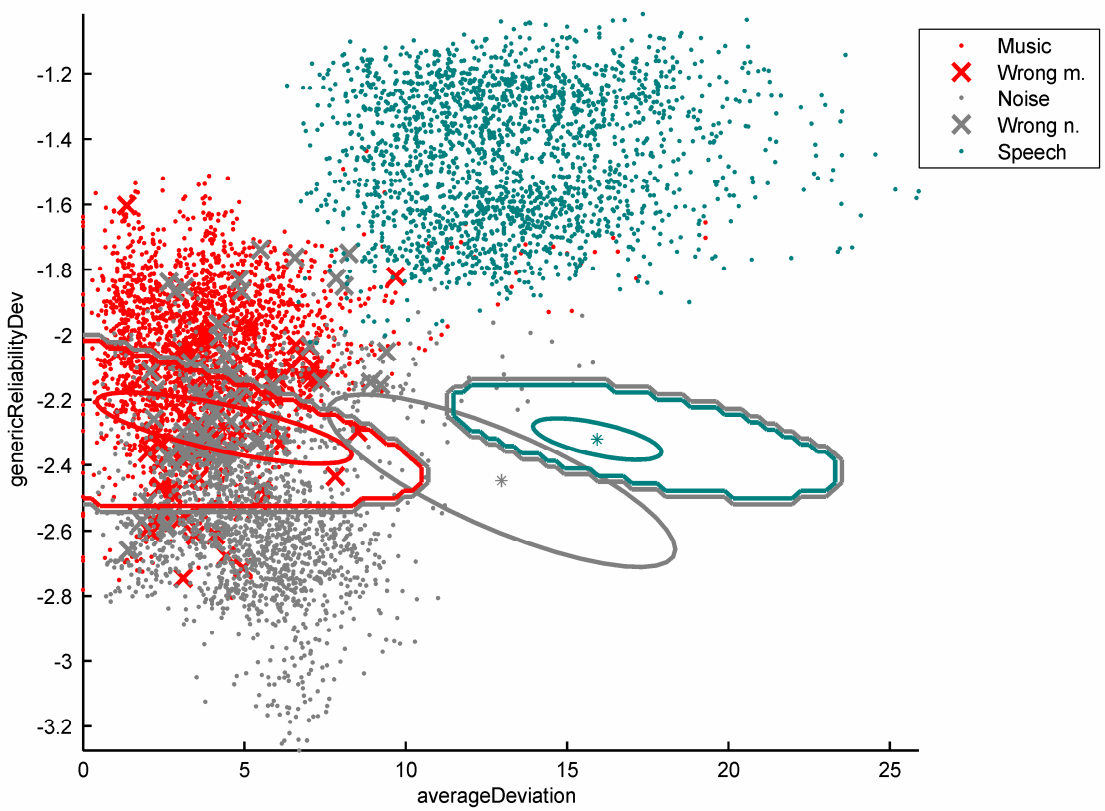


Figure 6.1.8: Two dimensions of a five dimensional model. Stars and ellipses are mean and covariances respectfully. Jagged lines are decision boundaries based on the two dimensions. Wrong classifications based on five dimensions are marked with x. No wrong speech existed.

The plot shows some very interesting features. The mean of the speech is no where near the mean that is expected and that would have been returned by the typical Bayes classifier. This plot is taken out of a 5 dimensional model, and of course the classification is not done based on what is shown here, but the remaining scatter plots show the same behaviour, although not as pronounced as in figure 6.1.8. Many of the misclassified points are inside their respective decision boundaries and many correctly classified points are outside.

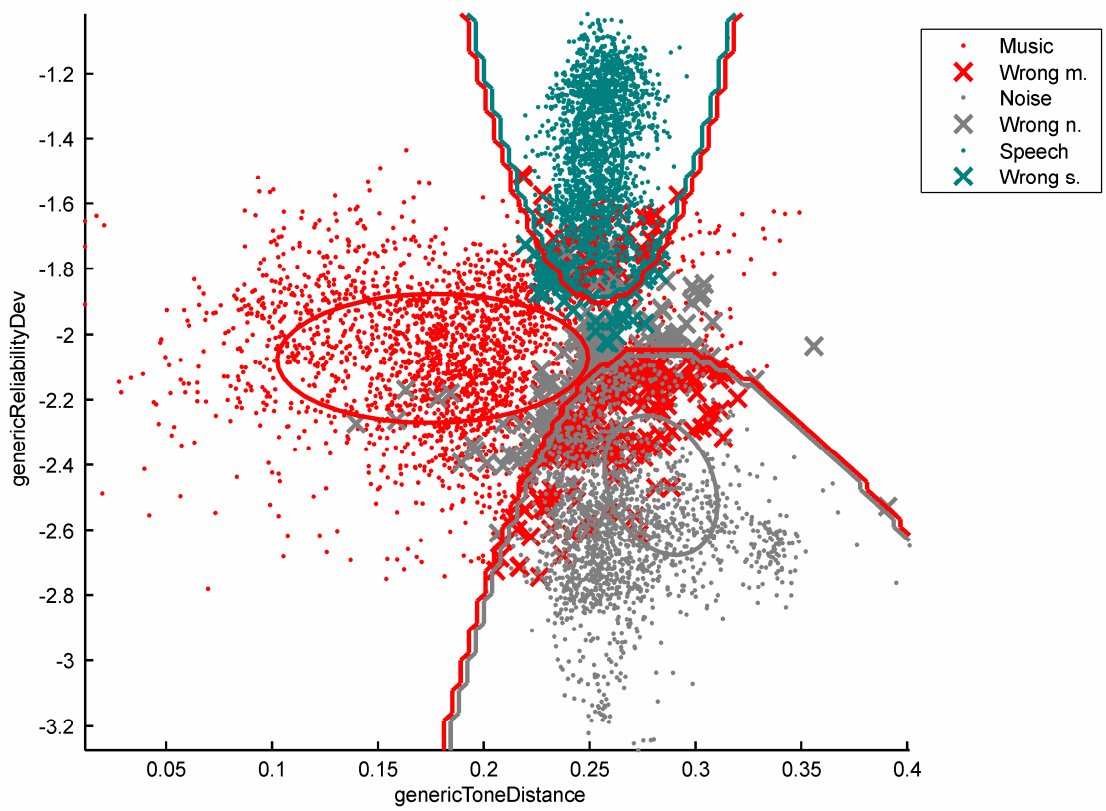


Figure 6.1.9: Two-dimensional model. Stars and ellipses are mean and covariances respectfully. Jagged lines are decision boundaries. Wrong classifications are marked with x. Scatter plot of three-dimensional model can be found in appendix.

When smaller dimensional models of 2 and 3 features are used, the expected behaviour is observed, figure 6.1.9. The explanation for this is as follows. The smaller dimensional models use all its power to classify for example 90 % of the points correctly. Then an extra dimension is added. If the within class parameters are used, the 10 % that lies outside the decision boundaries will probably also be close to or outside the decision boundary for the new dimension. This means that some small part of the model will give some extra information, but the most part will just make the model even more sure of the 90 % already classified points. When the parameters are optimized using all classes and dimensions the extra dimension can be used to reach the 10 % more directly. When doing the training it must be considered not to damage the classification of the 90 %, but this is taken care of by the update equations. This can cause dimensions that have been trained in a higher dimensional space to appear as above when looked at for themselves.

6.2 Investigation of misclassifications

To investigate the misclassifications of the final model the confusion matrix is created. The confusion matrix shows the relation between the real classes and the one the model selects.

		Predicted class		
		Music	Noise	Speech
True class	Music	2441	59	0
	Noise	62	1938	0
	Speech	0	0	2100

A very interesting property here is that no speech is misclassified and no sample is misclassified as speech. The only overlap exists between music and noise. This was also noticed in the scatter plots of the previous section. This is a very nice property, because speech is the most critical class to classify correctly. When speech is misclassified you loose information, for example the first word in a sentence. When misclassifying music and noise nothing crucial is lost.

Sounds causing the misclassifications

It is important to know what kind of sounds is causing the misclassifications. This way the model can eventually by improved on these specific issues. Some of the issues are caused by a failing pitch detector and some are because the pitch has problems with describing the problem. Some plots are shown, but the most important way of investigating the problems is by listening to the sounds. They will be described to get an understanding of the problems. Some of the misclassifications are just a single sample of an entire song. This will not be investigated. More interestingly, some sound clips cause many failures and these will be described.

In music two different reasons for misclassifications exist. When the music consists almost only of percussion instruments the music is classified as noise. This is not surprising. It can happen in drum solos in rock music, but is also present in some other genres. Some pieces starts with a percussion intro.

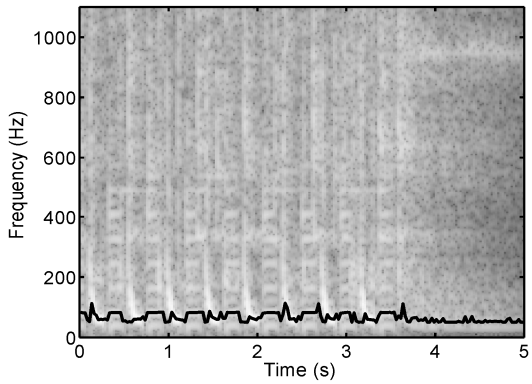


Figure 6.2.1: Misclassified window from percussion music. Very little pitch structure is present in the plot, nor when listened to.

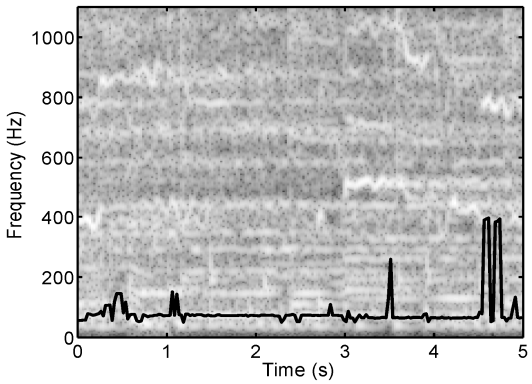


Figure 6.2.2: Misclassified window from new age music. Pitch is present when listened to, but is not apparent in the spectrum.

One of the percussion pieces consists solely of percussion instruments and one of the misclassified frames is shown in figure 6.2.1. It is clear that there is not much pitch to detect, and it must be concluded that the pitch is not a good characteristic for classification here.
Another problem with music is when many instruments are playing simultaneously. An example is shown in figure 6.2.2. The pitch detector has a hard time detecting a single pitch even though a pitch seems to dominate when listened to. It happens many places and especially rock and heavy metal, with electric guitars, drums, a synthesizer and the singer, has misclassified windows caused by this, but also classical music, like the classical violin concert, shows some problems. With a more advanced pitch detector, that can identify more than one pitch in the signal, it is possible that misclassifications can be corrected.

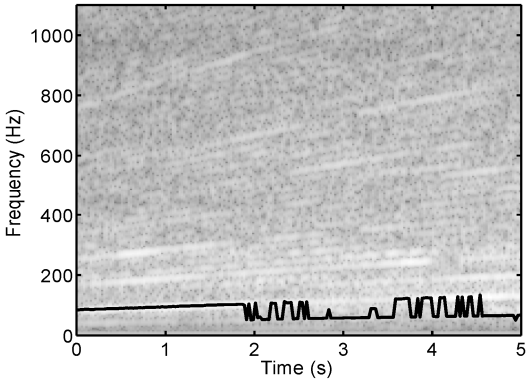


Figure 6.2.3: Misclassified window from traffic noise. The increasing pitch is from the engine of an accelerating car.

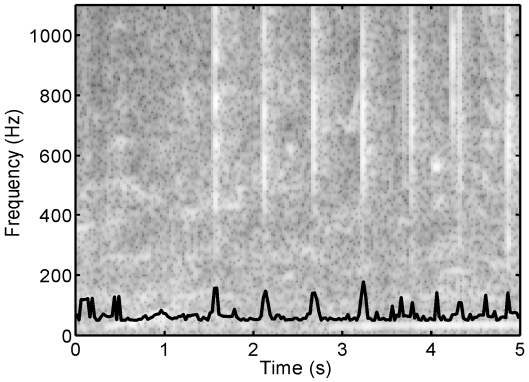


Figure 6.2.4: Misclassified window from hotel foyer noise. The spikes are high heals walking pass.

The misclassifications done in noise can also be divided in two parts. Some recordings of traffic are included in the noise class and they include the sound of cars accelerating. This causes the pitch from the engine to increase as shown in figure 6.2.3 and gives the classifier problems. The other part of misclassifications is when something close to white noise, for example many people speaking, are mixed with a more clear pitch. This also gives the classifier problems. An example from a hotel foyer where foot steps are present is shown in figure 6.2.4.

Logistic regression used together with the new model

In chapter 5 the logistic regression (discriminative family of models) and the new model showed very similar performance. It could be interesting to investigate if they misclassify the same samples or not. If not, they could be used together to improve performance. If they do misclassify the same points, it is an indication that they are basically the same model.

The new model has been trained with the almost optimal method 3. The number of samples misclassified by both the logistic regression model and the new model is plotted in figure 6.2.5. The logistic regression model is slightly better than the new model. Therefore the benefit of using both models instead of just the logistic regression is shown in figure 6.2.6.

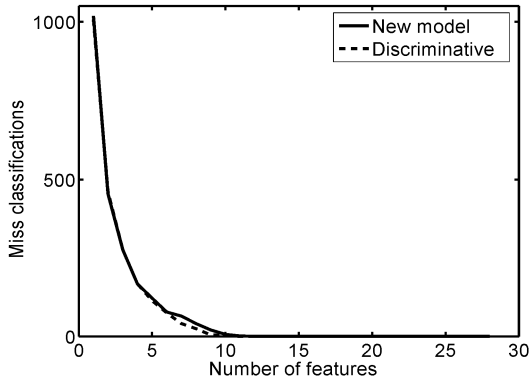


Figure 6.2.5: The two models have almost the same number of miss classifications with the discriminative model being slightly better.

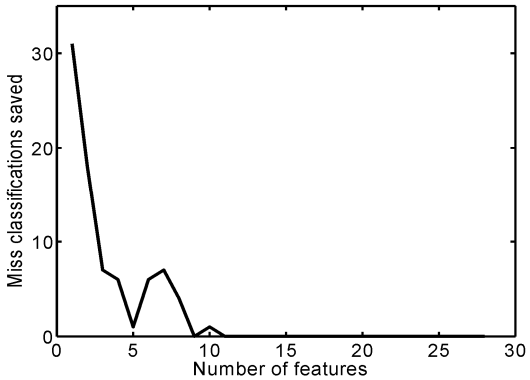


Figure 6.2.6: The number of misclassifications saved by using both models is very little compared to the total number.

The saved misclassifications are very few compared to the total number of misclassifications and a combination of both models yields only a little advantage. This indicates that the decision boundaries are close to one another and that they are basically the same model. This is not completely surprising given that they have the same order of complexity in the same feature space.

6.3 Effect of different FFT sizes on the classification system

The pitch detector that was selected in chapter 2 uses a quite large FFT. This is probably the most important aspect, making it difficult to include it in a hearing aid. This section makes a small comparison of the effect of the FFT size on the complete system. The final model is used, but retrained. This means that only the FFT size is changed and besides the retraining, everything is kept the same. This means that the comparison cannot be taken as a true comparison of a complete system using a certain FFT size. If this was the objective, a new ranking should be performed, because some features might be more influenced by the change than others. It should though give an indication of the penalty of using a smaller FFT.

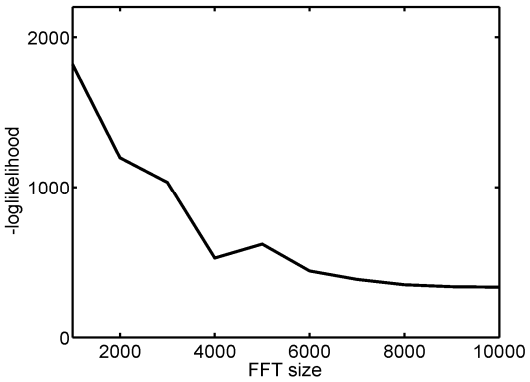


Figure 6.3.1: The training error is clearly a function of the FFT size. An unexplained bump is present at 5000.

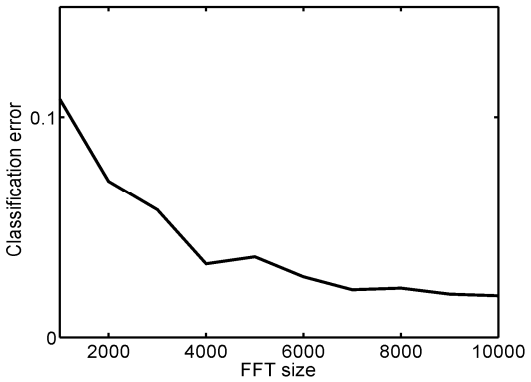


Figure 6.3.2: Also the classification error shows a clear dependence on FFT size and the same bump exists.

A clear dependence is shown between the FFT size and the classification error. From 8000 and up not much is lost. Even though the dependency exists it can also be seen that with a very small FFT of 1000 a classification error of only about 10% can be obtained. The plots indicate that not much will be gained if increasing the FFT size beyond the size of 10000 because the slope is gradually approaching zero. Which size to use must be an implementation specific decision.

7 Conclusion

The conclusion is divided in two sections. In the first section the results from the work in this project will be presented, and in the second ideas for future work will be suggested.

7.1.1 Results from this project

A new pitch detector was suggested based on a combination of two existing algorithms working in the frequency domain. It was compared against two other algorithms working in the time domain. A comparison was set up, and the new pitch detector showed better performance than the two others. The comparison was not a general comparison, because it was specifically tailored for using the pitch detector in classification. Focus was on computational burden and overall hit rate, and not exact accuracy. The other two algorithms will probably show better accuracy if sub Hz precision is desired. The real difference between the chosen algorithm and the other two was speed. The other two algorithms could, possibly, have obtained the same hit rate as the selected algorithm, but it would simply consume too much time. In the comparison that came closest in performance, the selected method was over 100 times faster than the faster of the other two. The time used by the new pitch detector for extracting the pitch, was 0.4 times the length of the signal.

Based on the pitch signal and the error coming from the selected pitch detector a number of features were found. True pitch values were separated from false with the use of reliable windows. Many features showed good separation, but the selection of features were not done until a classification model was set up. The Kolmogorov-Smirnov test was used to examine how gaussian the features were distributed. Both the features and the logarithm of the features were examined, and the one closest to gaussian was used.

First the Bayes classifier was used for classification. Three variations of covariance were used, a covariance for each class, common covariance for all classes and diagonal covariance for each class. All three variations showed an increase in training error when the numbers of features exceeded a certain value. This is quite strange because under maximum likelihood training this is not possible. It was shown that when training the Bayes classifier with a gaussian distribution and the data is not gaussian distributed it no longer results in maximum likelihood classification. A new model was suggested which assures maximum likelihood. There is an issue with the training of this model, though. It was circumvented, by training in a stepwise manner. The new model was put into perspective with the comparison of generative and discriminative models. Through literature studies, the generative model was found to be preferred, when the distribution of the model is the same as that of the data, and if training samples were limited. The discriminative model shows similar or better performance when enough training samples are available and when the distribution of the model is not the same as the data. The new model falls in between the two categories, being a discriminately trained generative model. The new model was compared to the original Bayes classifier, a generative model, and the logistic regression model which is of the discriminative class. The new model was clearly

better than the original and it showed comparable performance to the logistic regression model.
A final classification model was suggested using only five features, a covariance for each class and using the new model. The five features were based on the standard deviation of the pitch error, the distance to musical notes, the average slopeness inside the reliable windows, and two bins of a histogram of the difference between pitch measurements. The final model had a validation classification error of 1.9 %. The project showed that the pitch is indeed a good feature for sound classification, and it showed that with few, but well chosen, features a simple model can give very good results. Further more no speech samples were misclassified in the final model, which is a very nice property.

To round off things, the misclassifications of the two models, the new one and logistic regression was compared. They misclassified almost the same points, which suggests that the models share the same decision boundaries.
The influence of the size of the FFT in the pitch detector, and thereby the pitch accuracy, on the classification was investigated. It showed a clear dependence, but it also showed that accuracy beyond that used in the project would only give little extra information.

7.1.2 Future work

The work in this project presents a rather new way of using the pitch and therefore many things are still unsolved. First of all many, pitch detection algorithms exist and very few of them have been reviewed with classification in mind. The pitch detection is the most time consuming step, if the training of the model is not considered, and therefore would be an obvious place to optimize. The HPS algorithm is very fast and might be usable on its own. Also the length of the pitch detection window could be varied. The Bayesian pitch detector and HMUSIC could probably achieve comparative resolution with smaller windows.

With the features an obvious study is of the length of the feature window. This directly affects the decision horizon which is quite critical especially for speech. The first word is quite important for the understanding of a sentence.
A database with less dependence between the clips would also be desirable. Instead of using 5 clips from each song, it could be random if the clip was taken from the beginning, the middle or the end. It was only in the validation step real trouble was observed, so this would probably not change the results that much. The results would be more reliable, though.

There were problems with the training of the new model. They were solved, but in stepwise fashion. It could be nice with a more clean way of training the new model. This would probably also cut down training time.

If the complete system, in spite of all the optimizations, can not be fit in a hearing aid, it could also be interesting to fit the system in a mobile phone or on a PDA. There is much unused computational power in these devices, and the amount of information to be transferred to the hearing aid is very small, only a class every second.

8 Bibliography

[1] K.T. Abou-Moustafa, C.Y. Suen, *A generative-discriminative hybrid for sequential data classification*, Proc. Int. Conf. on Acoustics, Speech, and Signal Processing, Montreal 2004, vol. 5, pp. 805-808

[2] H. Akaike, *Information theory and the extension of the maximum likelihood principle*, 2nd Int. Symposium on Information Theory, 1973, republished in *Breakthroughs in Statistics*, Springer-Verlag, 1993, vol. 1, pp. 610-624

[3] F.R. Bach, M.I. Jordan, *Discriminative training of hidden Markov models for multiple pitch tracking*, Proc. Int. Conf. on Acoustics, Speech, and Signal Processing, Philadelphia 2005, vol. 5, 489-492

[4] F.R. Bach, M.I. Jordan, *Blind one-microphone speech separation: A spectral learning approach*, Neural Information Processing Systems 17, MIT Press, 2005, pp. 65-72

[5] C.M. Bishop, *Neural networks for pattern recognition*, Oxford University Press, 2004

[6] C.M. Bishop, *Netlab neural network software*, ver. 3.3, http://www.ncrg.aston.ac.uk/netlab/, 2004

[7] G. Bouchard, B. Triggs, *The trade-off between generative and discriminative classifiers*, Proc. Computational Statistics, Prague 2004, pp. 721-728

[8] M.C. Büchler, *Algorithms for sound classification in hearing instruments*, Ph.d. thesis, Swiss Federal Institute of Technology, Zurich, 2002

[9] M.G. Christensen, S.H. Jensen, S.V. Andersen, A. Jakobsson, *Subspace-based fundamental frequency estimation*, Proc. 12th European Signal Processing Conf., Vienna 2004, pp. 637-640

[10] P. de la Cuadra, A. Master, C. Sapp, *Efficient Pitch Detection Techniques for Interactive Music*, Proc. Int. Computer Music Conference, Havana 2001

[11] B. Efron, *The efficiency of logistic regression compared to normal discriminant analysis*, Journal of the American Statistical Association, 1975, vol. 70, no. 352, pp. 892-898

[12] Festvox, *CSTR US KED Timit*, http://festvox.org/dbs/dbs_kdt.html

[13] E. Frank, S. Kramer, *Ensembles of nested dichotomies for multi-class problems*, Proc. 21st Int. Conf. on Machine learning, Banff 2004, pp. 305-312

[14] L.K. Hansen, F.Å. Nielsen, J. Larsen, *Exploring fMRI data for periodic signal components*, Artificial Intelligence in Medicine, 2002, vol. 25, pp. 35-44

[15] T.J. Hastie, R.J. Tibshirani, *Generalized additive models*, Monographs on Statistics and Applied probability, Chapman & Hall, 1991

[16] N.S. Jayant, P. Noll, *Digital coding of waveforms. Principles and applications to speech and video*, Prentice-Hall, 1984

[17] R.A. Johnson, D.W. Wichern, *Applied multivariate statistical analysis*, 5th Ed., Prentice-Hall, 2002

[18] C. Jørgensen, *Stemning og musikalsk konsonans: et matematisk modelleringsprojekt*, IMFUFA, RUC, 2003

[19] R.E. Kass, A.E. Raftery, *Bayes Factors*, Journal of the American Statistical Association, 1995, vol. 90, no. 430, pp. 773-795

[20] P. Komarek, *Logistic regression for data mining and high-dimensional classification*, Ph.d. thesis, Robotics Institute, CMU, 2004

[21] P. Langley, S. Sage, *Induction of selective Bayesian classifiers*, Proc. 10th Conf. on Uncertainty in Artificial Intelligence, Seattle 1994, pp 399-406

[22] L. Lu, H. Jiang, H.J. Zhang, *A robust audio classification and segmentation method*, Proc. 9th ACM Int. Conf. Multimedia, New York 2001, pp. 203-211

[23] G.F. Meyer, *Keele Pitch Database*, http://www.liv.ac.uk/Psychology/HMP/projects/pitch.html, 2005

[24] T.P. Minka, *Algorithms for maximum-likelihood logistic regression*, Technical report, CMU, CALD, 2001

[25] National Center for Voice and Speech, http://www.ncvs.org/ncvs/tutorials/voiceprod/tutorial/influence.html

[26] A.Y. Ng, M.I. Jordan, *On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes*, Neural Information Processing Systems 14, MIT Press, 2002, pp. 841-848

[27] H.B. Nielsen, *immoptibox*, http://www2.imm.dtu.dk/~hbn/immoptibox/, ver. 1.2, IMM, DTU, 2004

[28] NIST/SEMATECH, *e-Handbook of Statistical Methods*, http://www.itl.nist.gov/div898/handbook, 2005, section 1.3.5

[29] T.F. Pedersen, *Bayesian analysis of rotating machines: A statistical approach to estimate and track the fundamental frequency*, Ph.d. thesis, IMM, DTU, 2003

[30] K.B. Petersen, M.S. Pedersen, *The matrix cookbook*, IMM, DTU, 2005

[31] S. Pfeiffer, S. Fischer and W. Effelsberg, *Automatic audio content analysis*, Proc. 4th ACM Int. Conf. Multimedia, Boston 1996, pp. 21-30

[32] T. Poulsen, *Taleforståelighed*, 3rd Ed., Laboratoriet for Akustik, DTH, 1993

[33] L. Rabiner, B.H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993

[34] Y.D. Rubinstein, T. Hastie, *Discriminative vs informative learning*, Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining, Newport Beach 1997, pp. 49-53

[35] D.W. Ruck, S.K. Rogers, M. Kabrisky, M.E. Oxley, B.W. Suter, *The multilayer perceptron as an approximation to a Bayes optimal discriminant function*, IEEE Transactions on Neural Networks, 1990, vol. 1, issue 4, pp. 296-298

[36] J. Saunders, *Real-time discrimination of broadcast speech/music*, Proc. Int. Conf. on Acoustics, Speech, and Signal Processing, Atlanta 1996, vol. 2, pp. 993-996,

[37] E. Scheirer, M. Slaney, *Construction and evaluation of a robust multifeature speech/music discriminator*, Proc. Int. Conf. on Acoustics, Speech, and Signal Processing, Munich 1997, vol. 2., pp. 1331-1334

[38] R.O. Schmidt, *Multiple emitter location and signal parameter estimation*, IEEE Transactions on Antennas and Propagation, 1986, vol. 34, no. 3, pp. 276-280

[39] G. Schwarz, *Estimating the dimension of a model*, The Annals of Statistics, 1978, vol. 6, no. 2, pp 461-464

[40] W.A. Sethares, R.D. Morris, J.C. Sethares, *Beat tracking of musical performances using low-level audio features*, IEEE Transactions on speech and audio processing, 2005, vol. 13., no. 2, pp. 275-285

[41] E. Wold, T. Blum, D. Keislaer, J. Wheaton, *Content-based classification, search, and retrieval of audio*, IEEE Multimedia, 1996, vol. 3, no. 3, pp. 27-36

9 Appendix

A Table of constants

Pitch detector

Detection range: 50 – 400 Hz
Precision: 1 Hz
Pitch window size: 100 ms
Pitch window overlap: 75 ms
Sampling frequency: 10 kHz
Samples pr. pitch window: 1000
FFT size: 10000
$R = 5$
Number of harmonics modelled: 5, 10 and 15
$e_M = 10$

Feature extraction

Feature window size: 5 s
Feature window overlap: 4 s
Pitch samples pr. feature window: 200
$f_t = 60$
$p_t = 15$

Sound database

Sound clip length: 30 s
Sampling frequency: 10 kHz
Number of channels: 1

B Derivation of equation (2.2.8)

$$\frac{\partial E_A}{\partial A_0} = \left(A_0 - \frac{3A_1 - A_2}{2} \right) + \frac{1}{2}\left(\frac{A_0 + A_2}{2} - A_1 \right) + A_0 - S(\omega_0) = 0 \Leftrightarrow$$

$$S(\omega_0) = \frac{9}{4} A_0 - 2A_1 + \frac{3}{4} A_2$$

$$\frac{\partial E_A}{\partial A_1} = \frac{3}{2}\left(\frac{3A_1 - A_2}{2} - A_0 \right) + \left(A_1 - \frac{A_0 + A_2}{2} \right) + \frac{1}{2}\left(\frac{A_1 + A_3}{2} - A_2 \right) + A_1 - S(2\omega_0) = 0 \Leftrightarrow$$

$$S(2\omega_0) = -2A_0 + \frac{9}{2} A_1 - \frac{7}{4} A_2 + \frac{1}{4} A_3$$

$$\frac{\partial E_A}{\partial A_i} = \left(A_i - \frac{A_{i-1} + A_{i+1}}{2} \right) + \frac{1}{2}\left(\frac{A_{i-2} + A_i}{2} - A_{i-1} \right) + \frac{1}{2}\left(\frac{A_i + A_{i+2}}{2} - A_{i+1} \right) + A_i - S\left((i+1)\omega_0\right) = 0 \Leftrightarrow$$

$$S\left((i+1)\omega_0\right) = \frac{1}{4}A_{i-2} - A_{i-1} + \frac{5}{2}A_i - A_{i+1} + \frac{1}{4}A_{i+2}$$

$$\frac{\partial E_A}{\partial A_{N-1}} = \frac{1}{2}\left(\frac{A_{N-3} + A_{N-1}}{2} - A_{N-2} \right) + \left(A_{N-1} - \frac{A_{N-2} + A_N}{2} \right) + \frac{3}{2}\left(\frac{3A_{N-1} - A_{N-2}}{2} - A_N \right) + A_{N-1} - S\left(N\omega_0\right) = 0 \Leftrightarrow$$

$$S\left(N\omega_0\right) = \frac{1}{4}A_{N-3} - \frac{7}{4}A_{N-2} + \frac{9}{2}A_{N-1} - 2A_N$$

$$\frac{\partial E_A}{\partial A_N} = \frac{1}{2}\left(\frac{A_{N-2} + A_N}{2} - A_{N-1} \right) + \left(A_N - \frac{3A_{N-1} - A_{N-2}}{2} \right) + A_N - S\left((N+1)\omega_0\right) = 0 \Leftrightarrow$$

$$S\left((N+1)\omega_0\right) = \frac{3}{4}A_{N-2} - 2A_{N-1} + \frac{9}{4}A_N$$

C Derivation of equation (2.3.8)

$$\log\left(P\left(\boldsymbol{y} \mid \omega_0, K\right)\right) = \frac{1}{2}\log\left(\left|\boldsymbol{V}_P\right| a^d\right) - \frac{1}{2}\log\left(\left|\boldsymbol{V}\right| {a_P}^{d_P}\right)$$

$$= \frac{d}{2}\log(a) - \frac{1}{2}\log\left(\left|{\boldsymbol{V}_P}^{-1}\right|\right) + \frac{1}{2}\log\left(\left|\boldsymbol{V}^{-1}\right|\right) - \frac{d_P}{2}\log\left(a_P\right)$$

$$= \frac{3}{2}\log\left(\sigma_y^2\right) - \frac{1}{2}\log\left(\left|\frac{Tr\left(\boldsymbol{XX'}\right)}{T}\boldsymbol{I} + \boldsymbol{X'X}\right|\right) + \frac{1}{2}\log\left(\left|\frac{Tr\left(\boldsymbol{XX'}\right)}{T}I\right|\right) - \frac{3+T}{2}\log\left(\left(T+1\right)\sigma_y^2 - \boldsymbol{y'X}\left(\frac{Tr\left(\boldsymbol{XX'}\right)}{T}\boldsymbol{I} + \boldsymbol{X'X}\right)^{-1}\boldsymbol{X'y}\right)$$

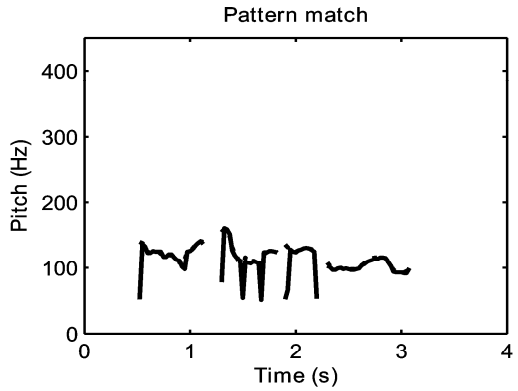
$$= \frac{3}{2}\log\left(\sigma_y^2\right) - \frac{1}{2}\log\left(\left|\frac{Tr\left(\boldsymbol{XX'}\right)}{T}\boldsymbol{I} + \boldsymbol{X'X}\right|\right) + \frac{K}{2}\log\left(\frac{Tr\left(\boldsymbol{XX'}\right)}{T}\right) - \frac{3+T}{2}\log\left(\left(T+1\right)\sigma_y^2 - \boldsymbol{y'X}\left(\frac{Tr\left(\boldsymbol{XX'}\right)}{T}\boldsymbol{I} + \boldsymbol{X'X}\right)^{-1}\boldsymbol{X'y}\right)$$

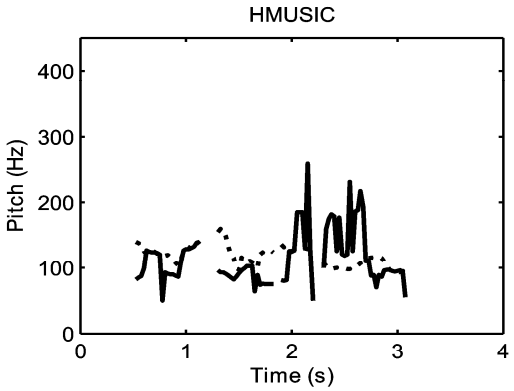
D Derivation of equation (2.3.10)

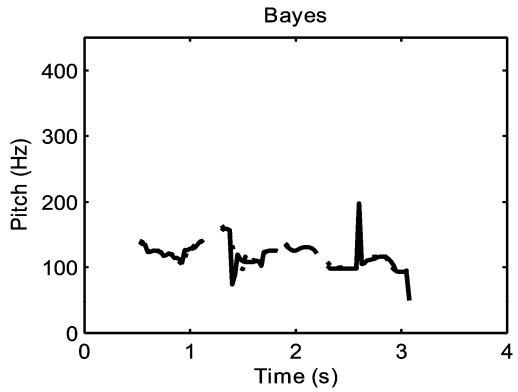
$$\log\left(P(\boldsymbol{y}\mid\omega_0,K)\right)=\frac{3}{2}\log\left(\sigma_y^2\right)-\frac{1}{2}\log\left(\left|\frac{Tr(\boldsymbol{XX}')}{T}\boldsymbol{I}+\boldsymbol{X}'\boldsymbol{X}\right|\right)+\frac{K}{2}\log\left(\frac{Tr(\boldsymbol{XX}')}{T}\right)$$
$$-\frac{3+T}{2}\log\left((T+1)\sigma_y^2-\boldsymbol{y}'\boldsymbol{X}\left(\frac{Tr(\boldsymbol{XX}')}{T}\boldsymbol{I}+\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\boldsymbol{X}'\boldsymbol{y}\right)$$
$$=\frac{3}{2}\log\left(\sigma_y^2\right)-\frac{3+T}{2}\log\left((T+1)\sigma_y^2\right)$$

E Pitch comparison

Timit pitch 1, 5 harmonics

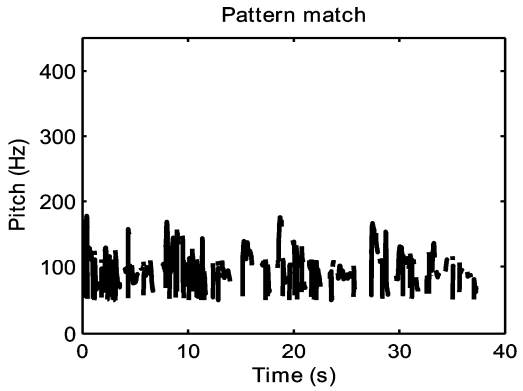


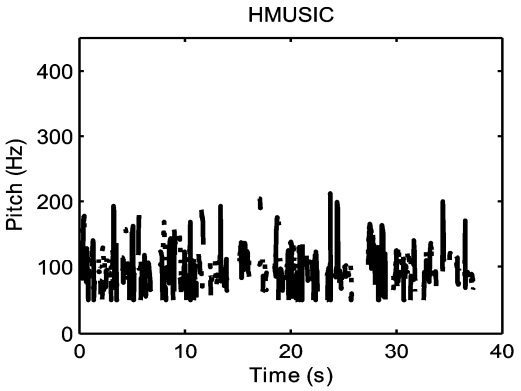


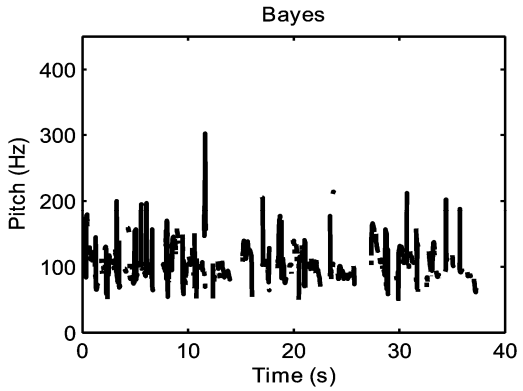


	Missrate	Accuracy
Pattern match	0.10	1.42
HMUSIC	0.64	2.85
Bayes	0.08	1.94

Keele pitch male 1, 5 harmonics

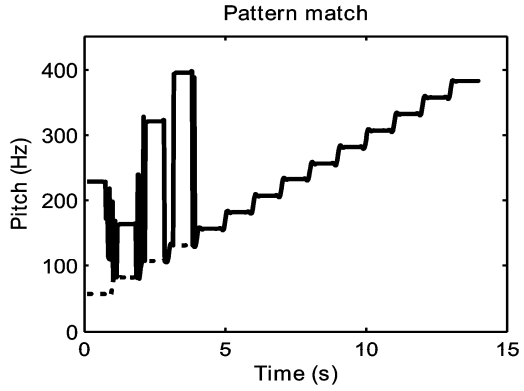


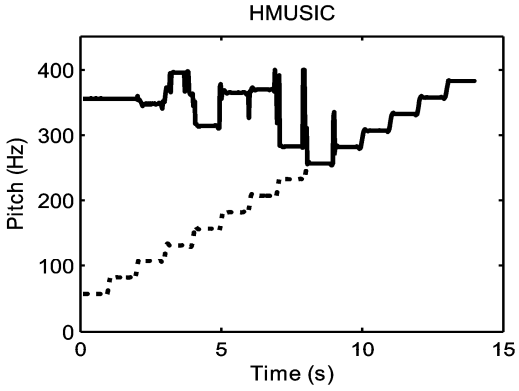


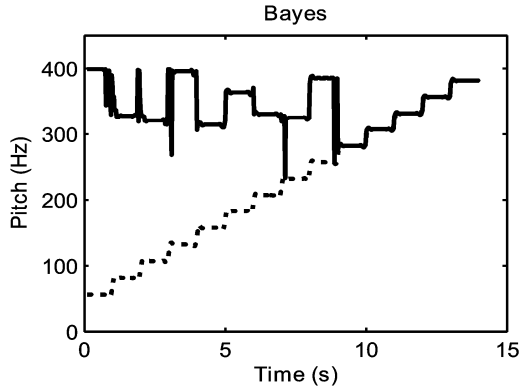


	Missrate	Accuracy
Pattern match	0.10	1.57
HMUSIC	0.64	3.35
Bayes	0.17	2.23

Synthetic pitch 2, envelope 2, no noise, 5 harmonics

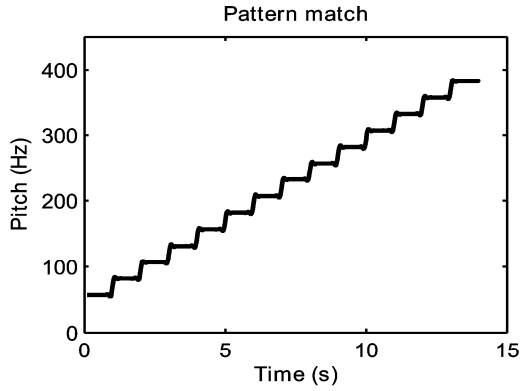


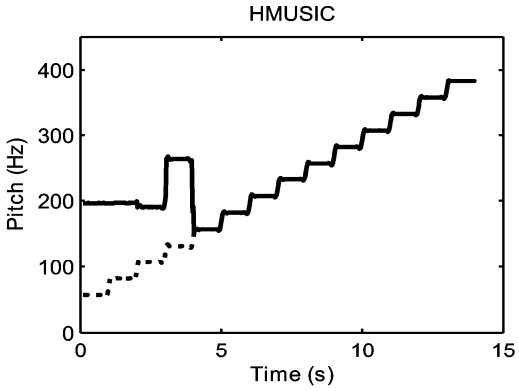


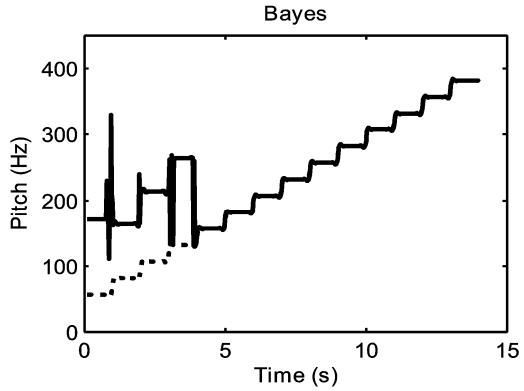


	Missrate	Accuracy
Pattern match	0.22	0.22
HMUSIC	0.57	0.29
Bayes	0.64	0.56

Synthetic pitch 2, envelope 3, no noise, 5 harmonics

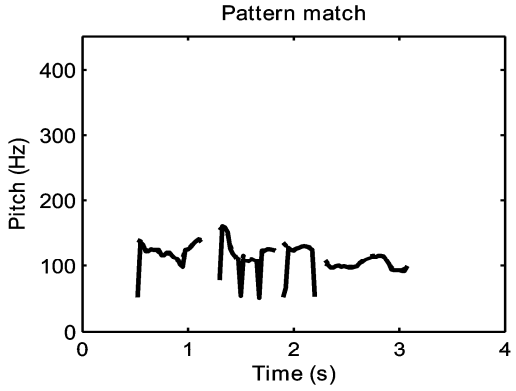


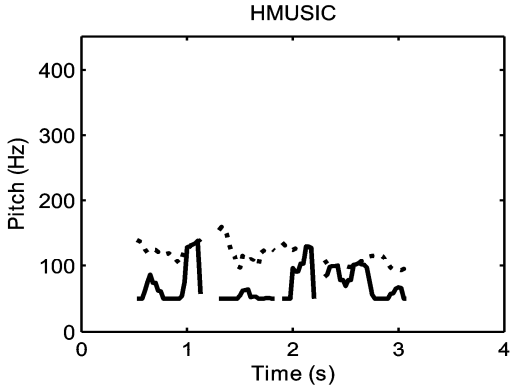


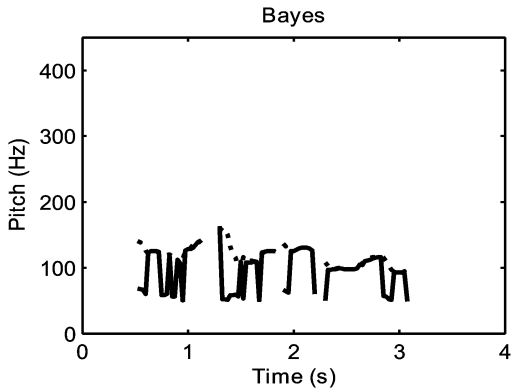


	Missrate	Accuracy
Pattern match	0.00	0.17
HMUSIC	0.28	0.33
Bayes	0.27	0.48

Timit pitch 1, 10 harmonics

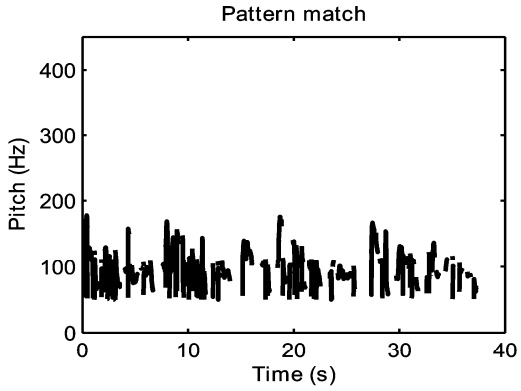


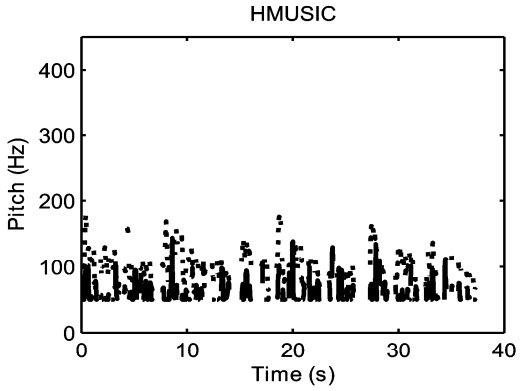


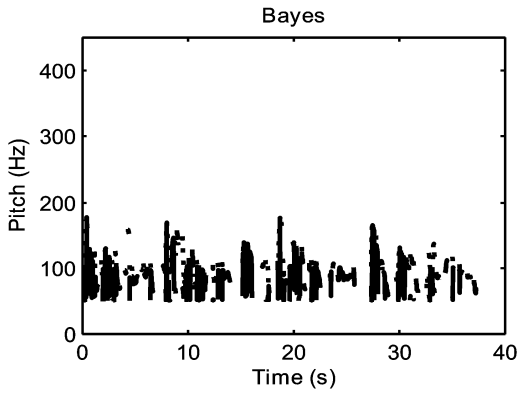


	Missrate	Accuracy
Pattern match	0.10	1.42
HMUSIC	0.82	2.06
Bayes	0.33	1.60

Keele pitch male 1, 10 harmonics

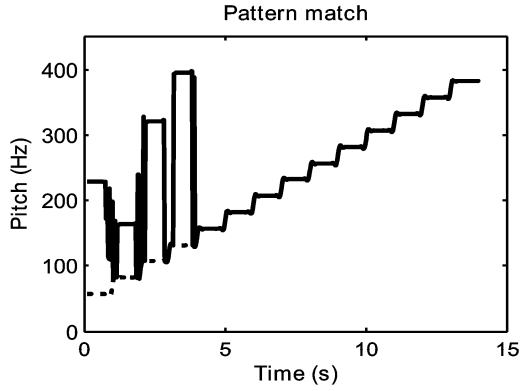


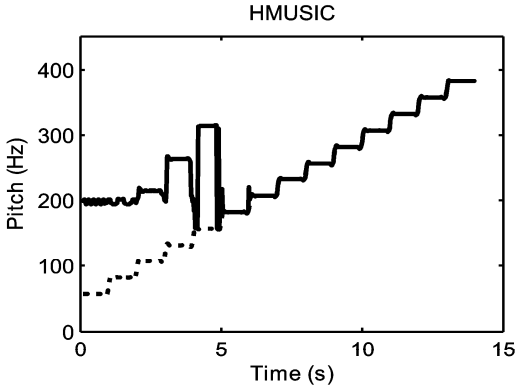


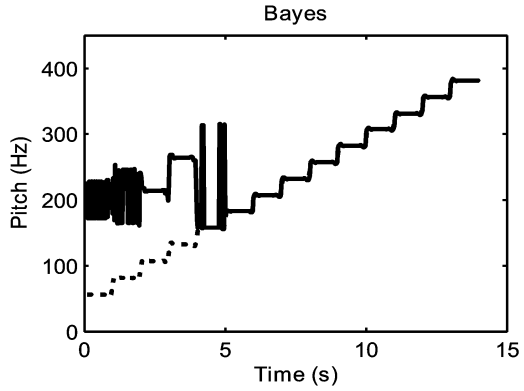


	Missrate	Accuracy
Pattern match	0.10	1.57
HMUSIC	0.91	3.96
Bayes	0.30	2.07

Synthetic pitch 2, envelope 2, no noise, 10 harmonics

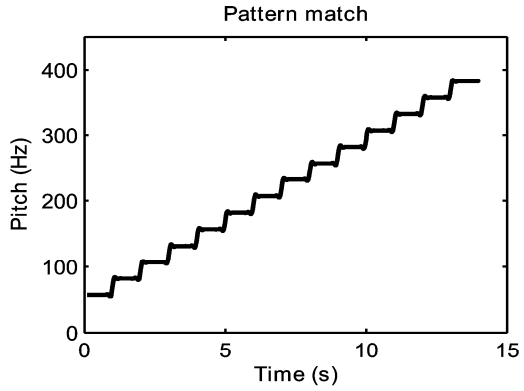


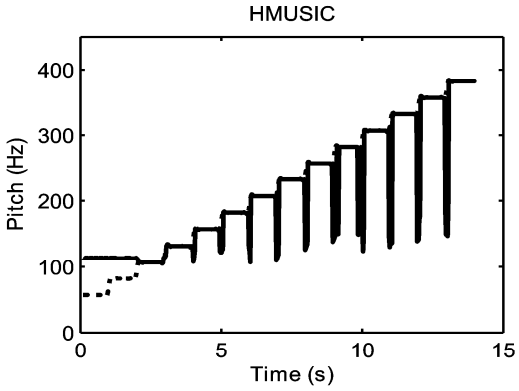


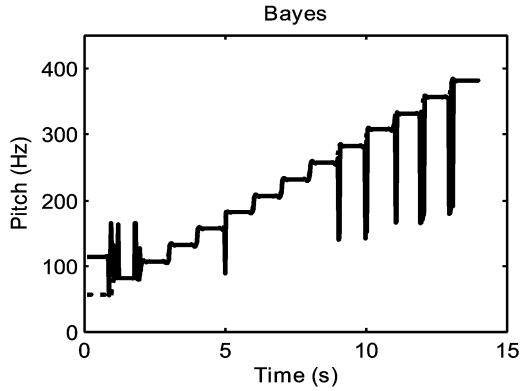


	Missrate	Accuracy
Pattern match	0.22	0.22
HMUSIC	0.34	0.42
Bayes	0.29	0.63

Synthetic pitch 2, envelope 3, no noise, 10 harmonics

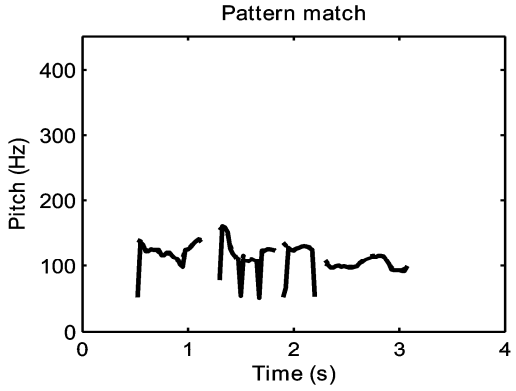


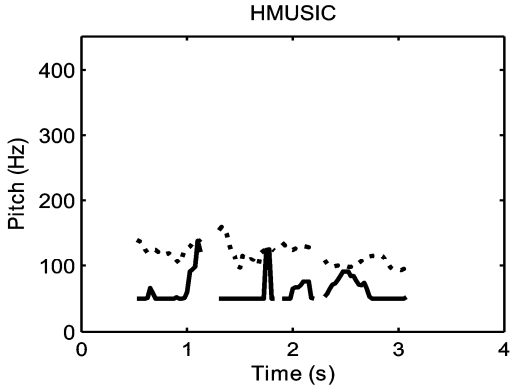


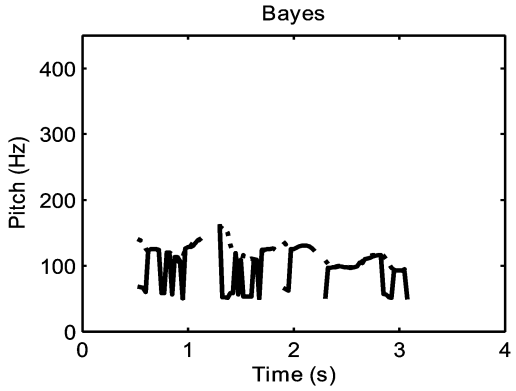


	Missrate	Accuracy
Pattern match	0.00	0.17
HMUSIC	0.23	0.31
Bayes	0.10	0.44

Timit pitch 1, 15 harmonics

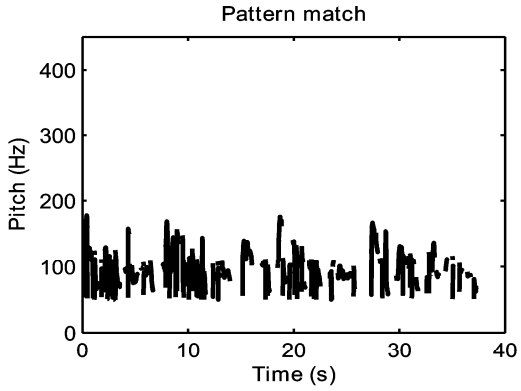


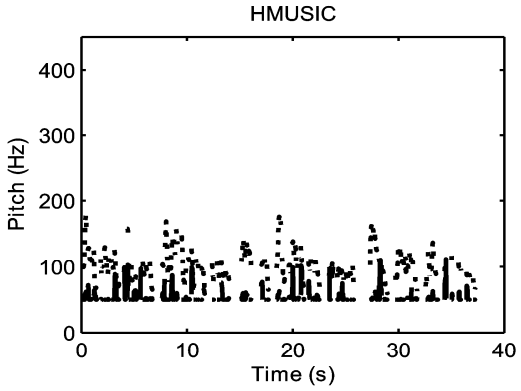


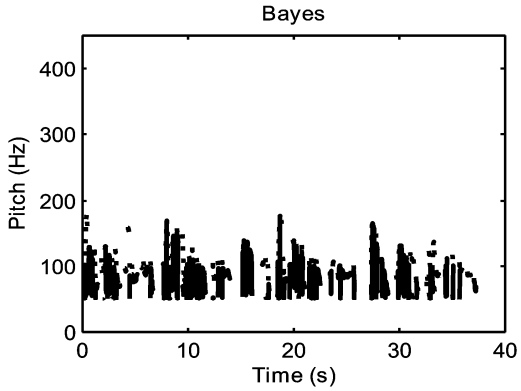


	Missrate	Accuracy
Pattern match	0.10	1.42
HMUSIC	0.93	4.67
Bayes	0.32	1.65

Keele pitch male 1, 15 harmonics

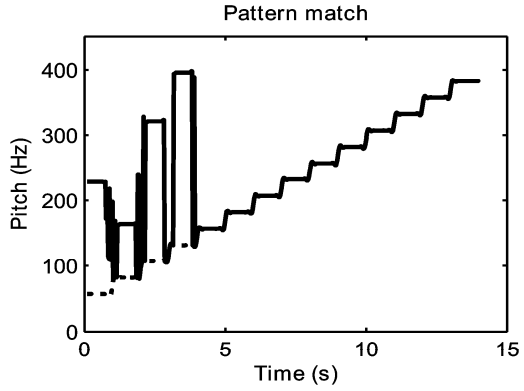


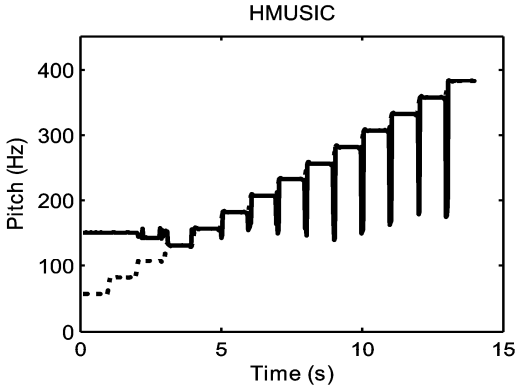


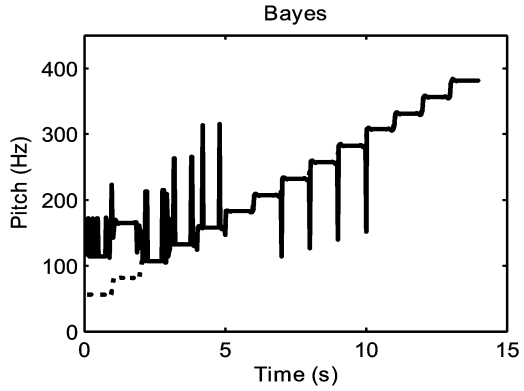


	Missrate	Accuracy
Pattern match	0.10	1.57
HMUSIC	0.97	4.40
Bayes	0.37	2.01

Synthetic pitch 2, envelope 2, no noise, 15 harmonics

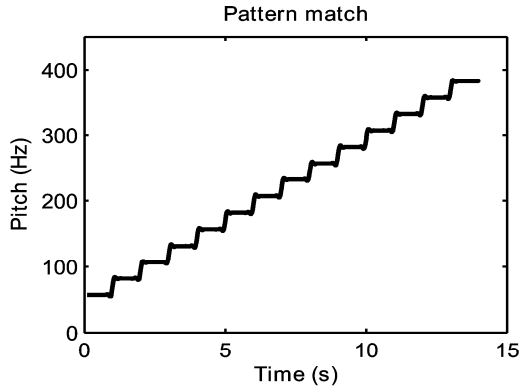


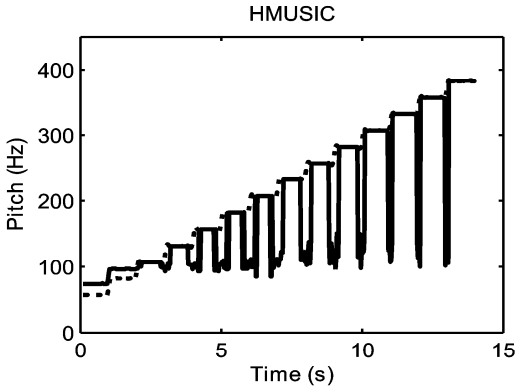


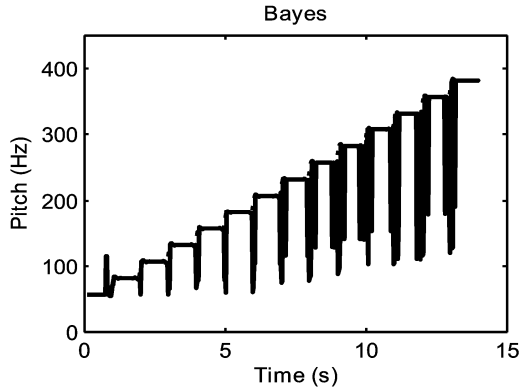


	Missrate	Accuracy
Pattern match	0.22	0.22
HMUSIC	0.27	0.36
Bayes	0.18	0.51

Synthetic pitch 2, envelope 3, no noise, 15 harmonics





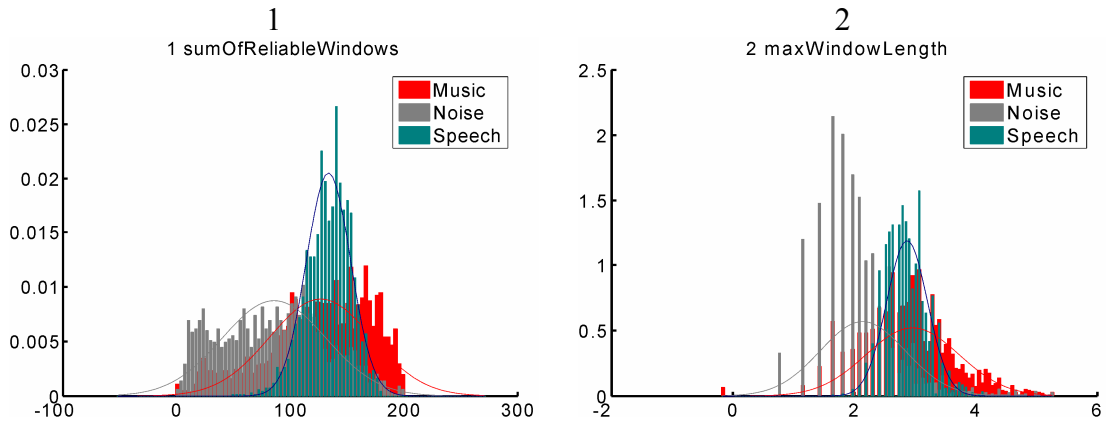


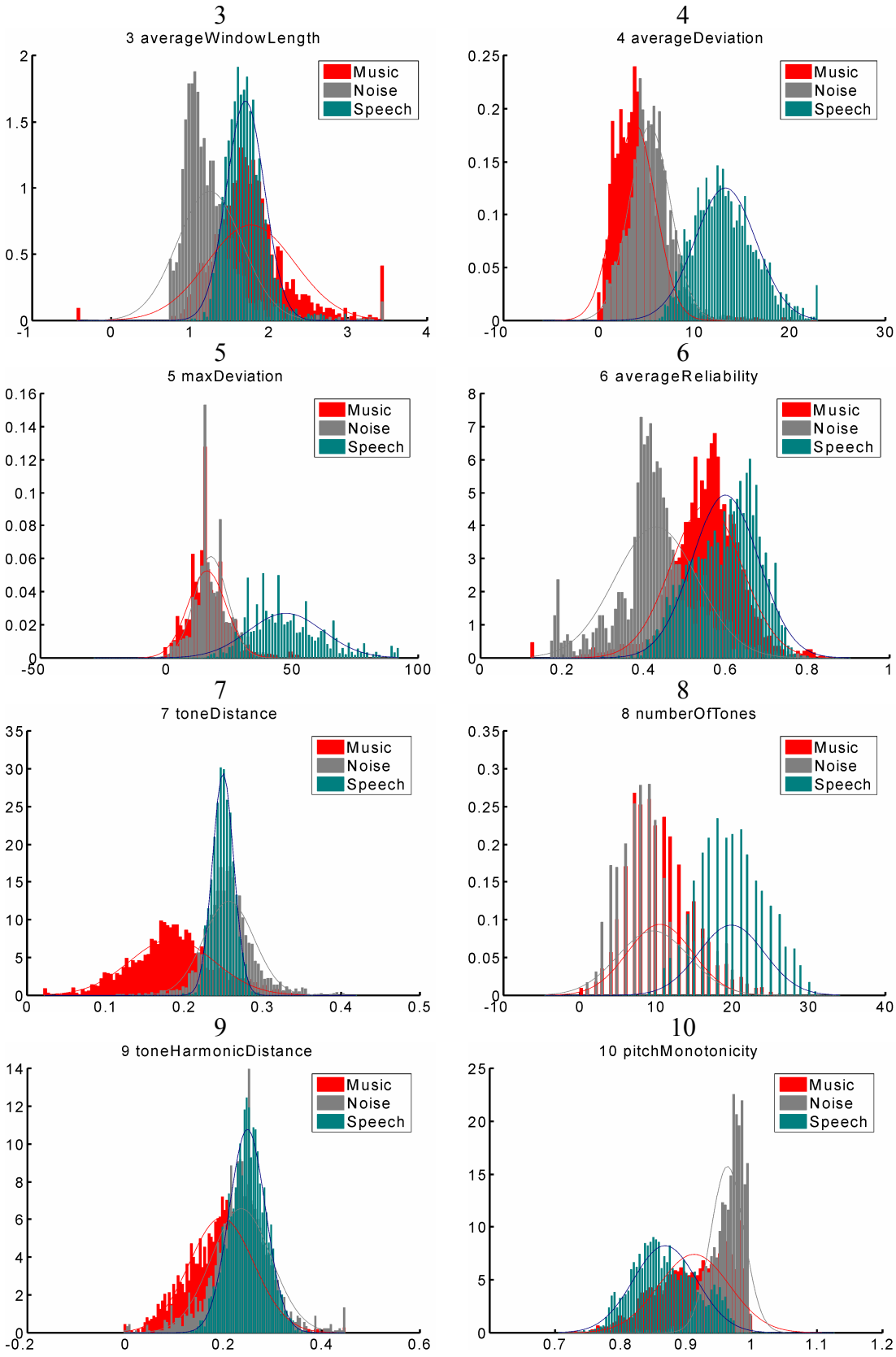
	Missrate	Accuracy
Pattern match	0.00	0.17
HMUSIC	0.38	0.45
Bayes	0.12	0.26

F List of implemented features

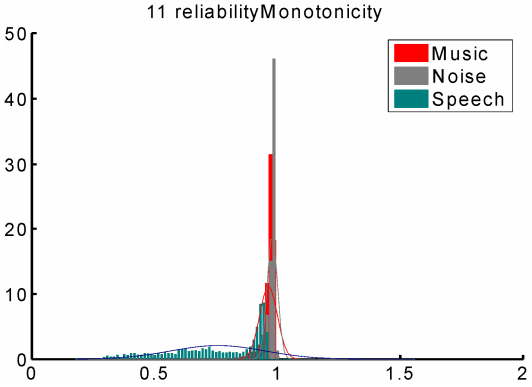
1 sumOfReliableWindows
2 maxWindowLength
3 averageWindowLength
4 averageDeviation
5 maxDeviation
6 averageReliability
7 toneDistance
8 numberOfTones
9 toneHarmonicDistance
10 pitchMonotonicity
11 reliabilityMonotonicity
12 genericMean
13 genericDev
14 genericAbsDiffMean
15 genericAbsDiffDev
16 genericAbsDiff1
17 genericAbsDiff2
18 genericAbsDiff3
19 genericAbsDiff4
20 genericAbsDiff5
21 genericAbsDiff6
22 genericAbsDiff7
23 genericAbsDiff8
24 genericMCR
25 genericToneDistance
26 genericNumberOfTones
27 genericReliabilityMean
28 genericReliabilityDev

G Feature plots

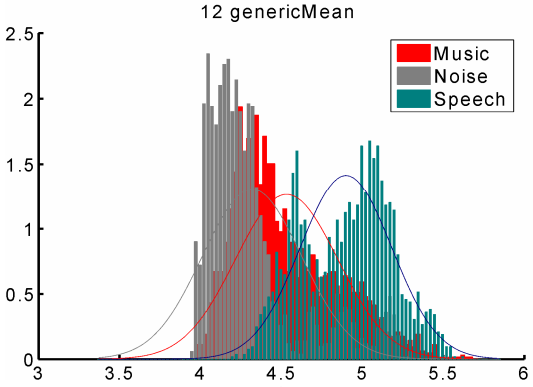



3
3 averageWindowLength
Music
Noise
Speech
4
4 averageDeviation
Music
Noise
Speech
5
5 maxDeviation
Music
Noise
Speech
6
6 averageReliability
Music
Noise
Speech
7
7 toneDistance
Music
Noise
Speech
8
8 numberOfTones
Music
Noise
Speech
9
9 toneHarmonicDistance
Music
Noise
Speech
10
10 pitchMonotonicity
Music
Noise
Speech

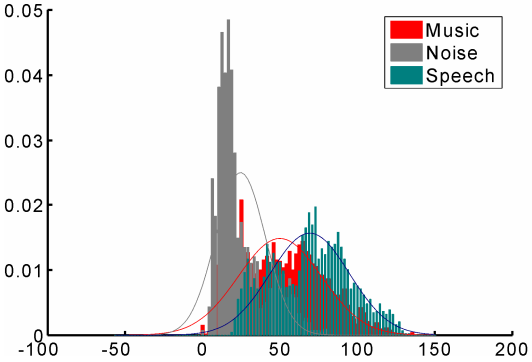
11



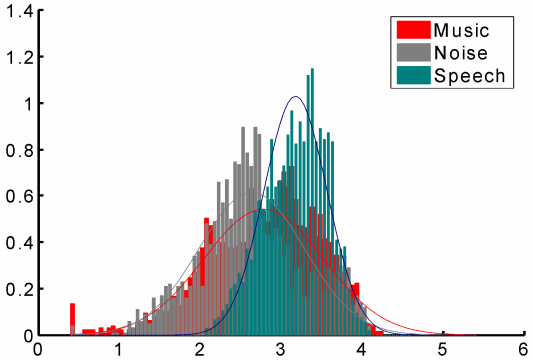
12



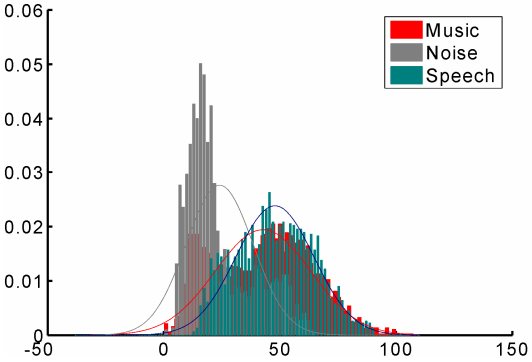
13



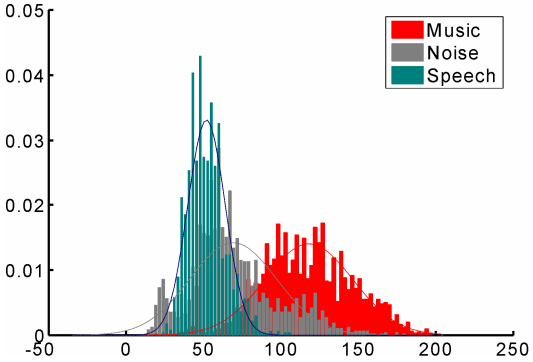
14



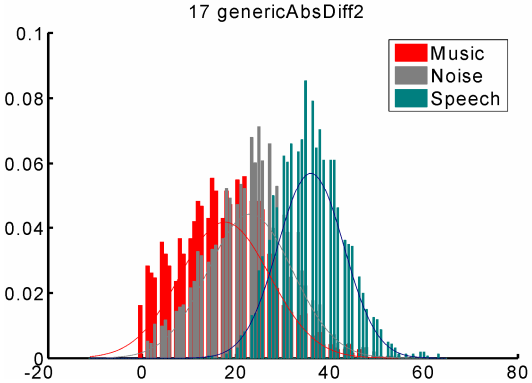
15



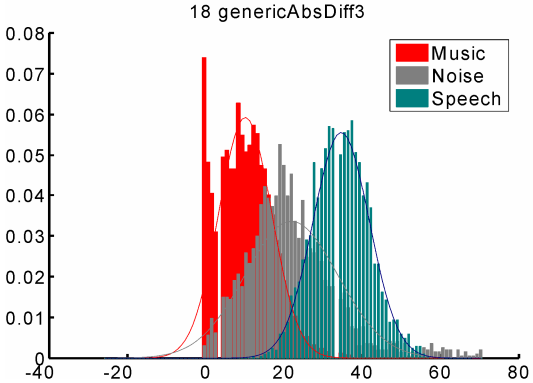
16



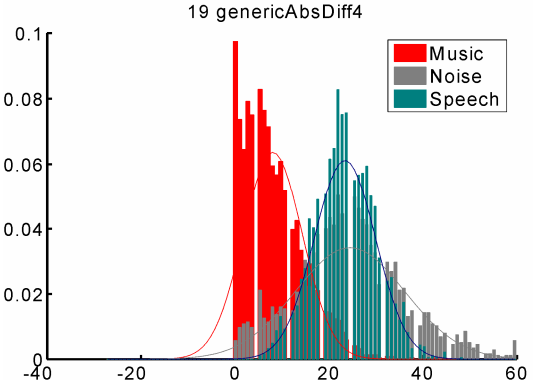
17



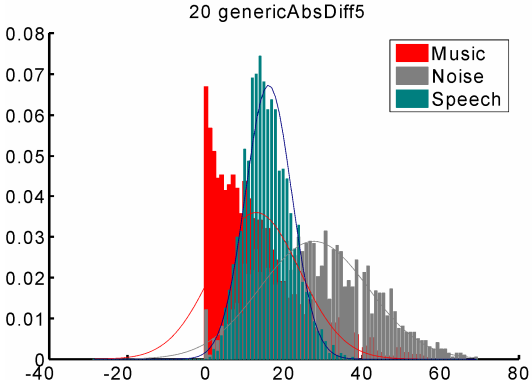
18



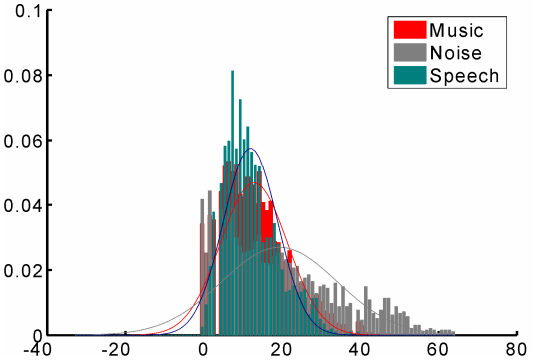
19



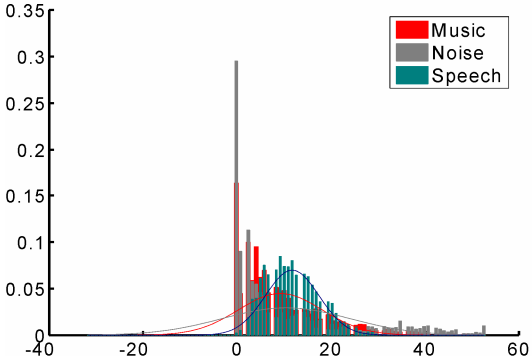
20



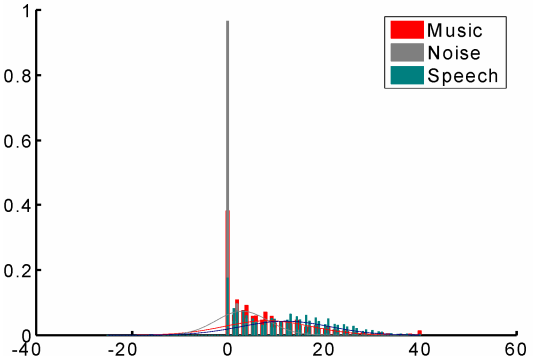
21



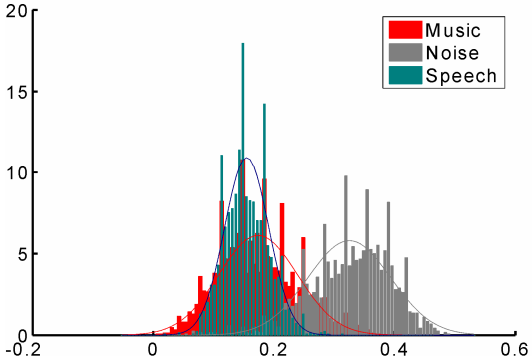
22



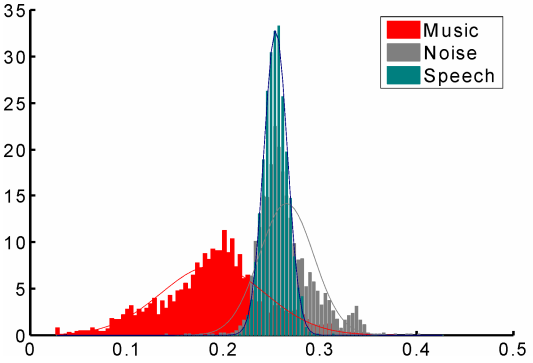
23



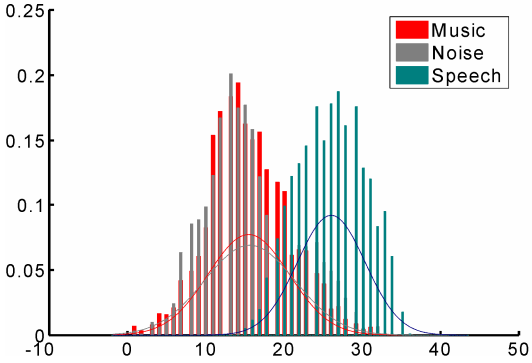
24

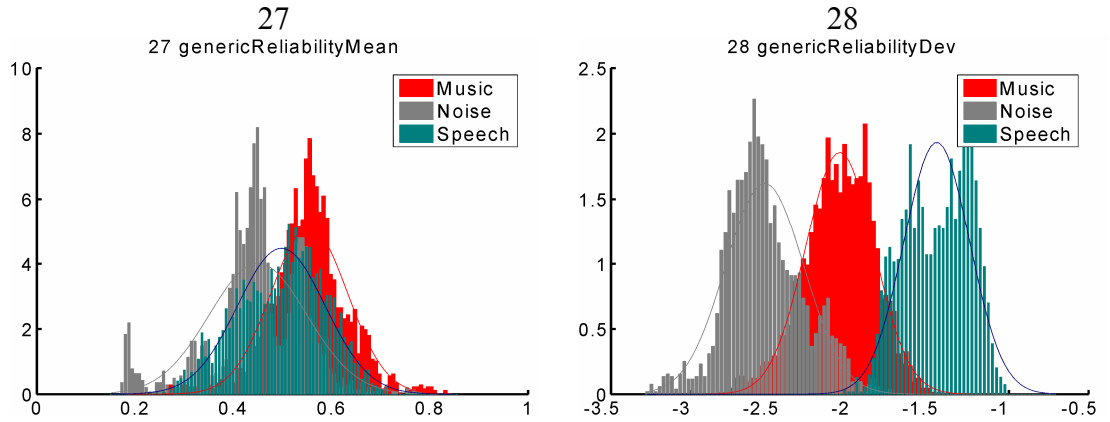


25



26





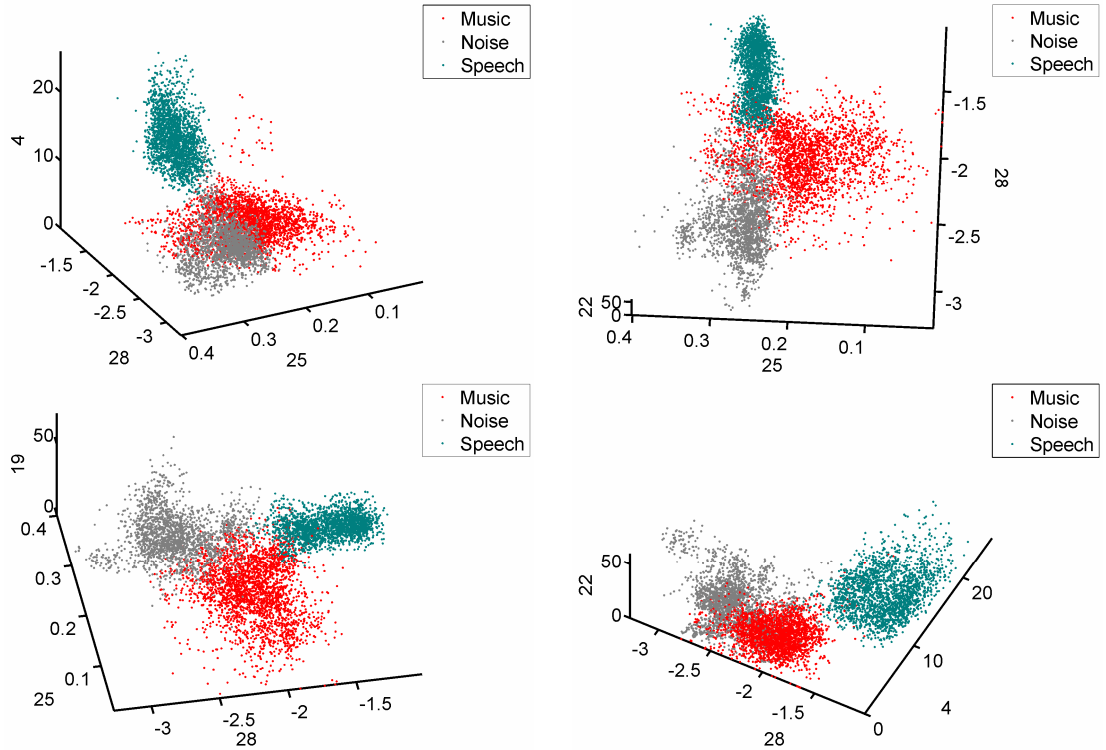
H Derivation of equation (5.2.12)

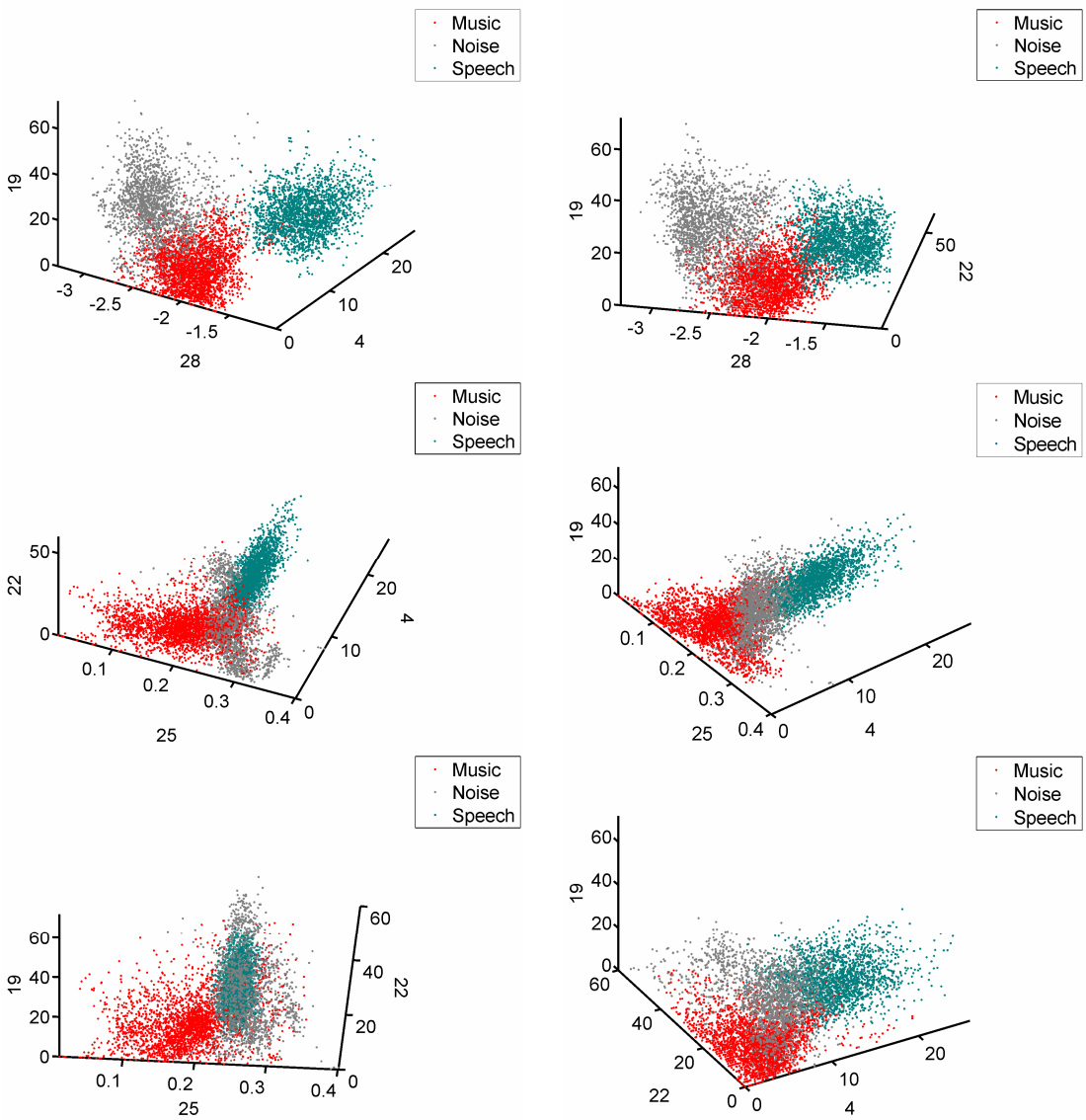
$$\frac{\partial E}{\partial p(c)} = \sum_{n=1}^{N} \frac{1}{\sum_{c'=1}^{C} \frac{p(\boldsymbol{x}_n \mid c')p(c')}{p(\boldsymbol{x}_n \mid c_n)p(c_n)}} \left(\frac{p(\boldsymbol{x}_n \mid c)}{p(\boldsymbol{x}_n \mid c_n)p(c_n)} \delta_{c \neq c_n} - \left(\sum_{c'=1}^{C} \frac{p(\boldsymbol{x}_n \mid c')p(c')}{p(\boldsymbol{x}_n \mid c)p(c)^2} - \frac{p(\boldsymbol{x}_n \mid c)p(c)}{p(\boldsymbol{x}_n \mid c)p(c)^2} \right) \delta_{c=c_n} \right)$$

$$= \sum_{n=1}^{N} \frac{p(\boldsymbol{x}_n \mid c)}{\sum_{c'=1}^{C} p(\boldsymbol{x}_n \mid c')p(c')} \delta_{c \neq c_n} - \frac{1}{p(c)} \left(1 - \frac{p(\boldsymbol{x}_n \mid c)p(c)}{\sum_{c'=1}^{C} p(\boldsymbol{x}_n \mid c')p(c')} \right) \delta_{c=c_n}$$

$$= \frac{1}{p(c)} \sum_{n=1}^{N} p(c \mid \boldsymbol{x}_n) \delta_{c \neq c_n} - (1 - p(c \mid \boldsymbol{x}_n)) \delta_{c=c_n} = \frac{1}{p(c)} \sum_{n=1}^{N} p(c \mid \boldsymbol{x}_n) - \delta_{c=c_n}$$

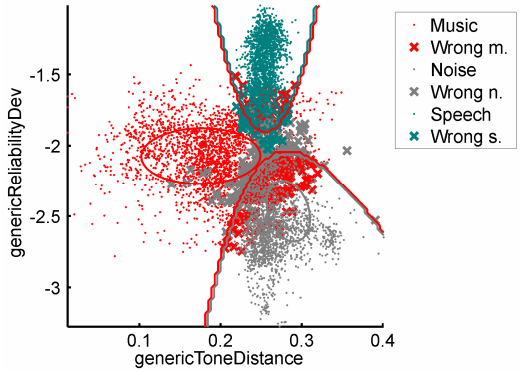
I 3-D comparisons of final features



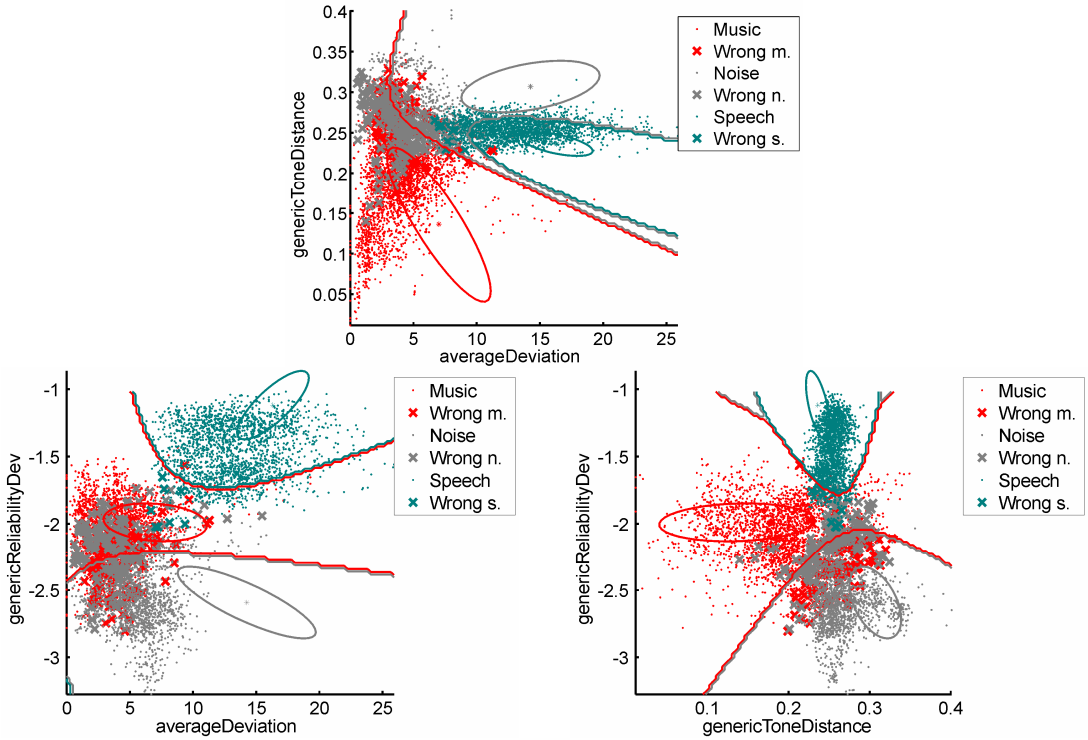


J 2-D feature comparisons of final model

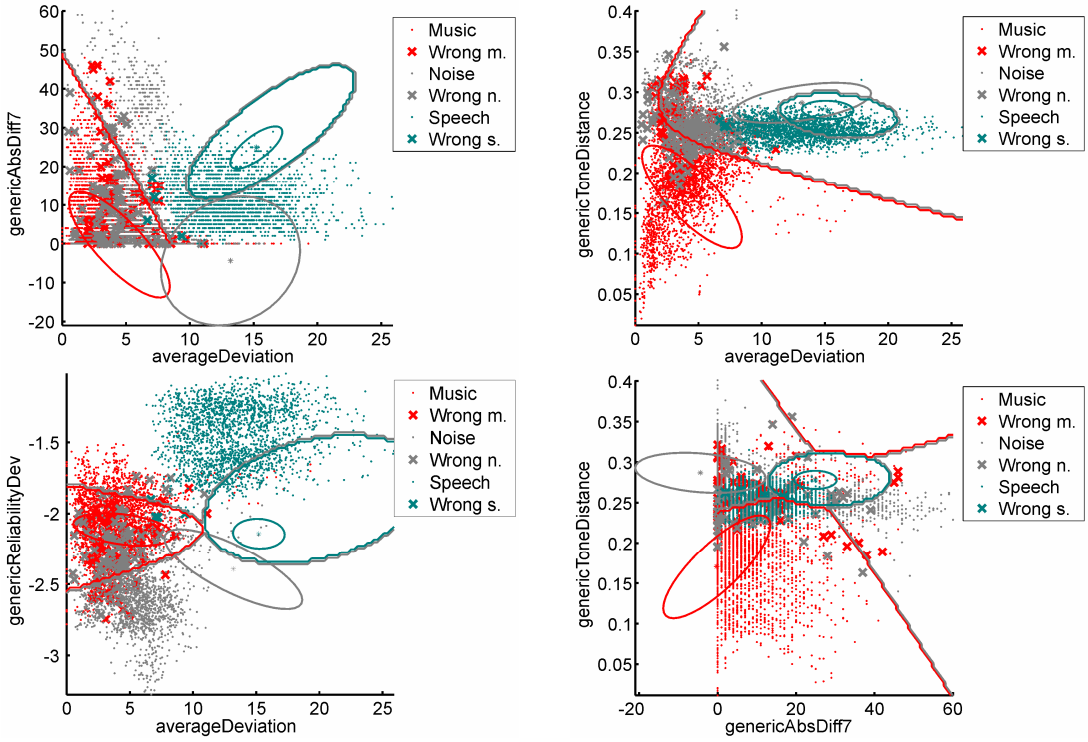
2 features

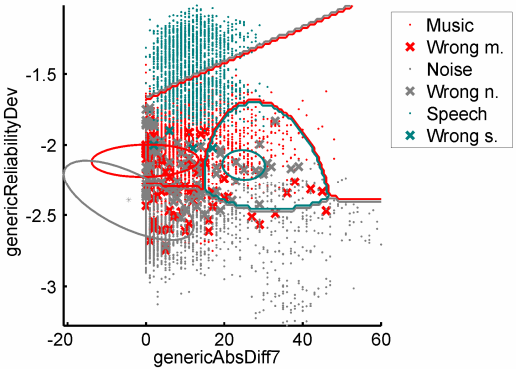


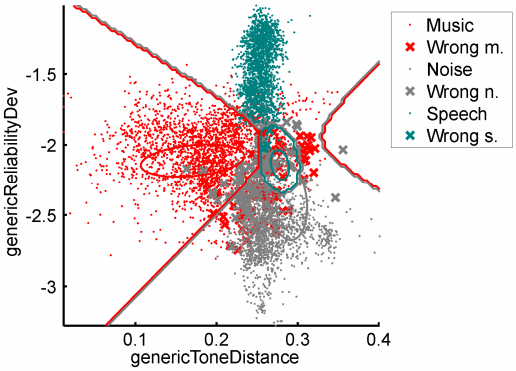
3 features



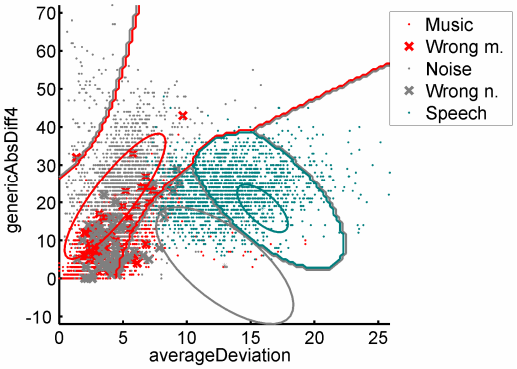
4 features

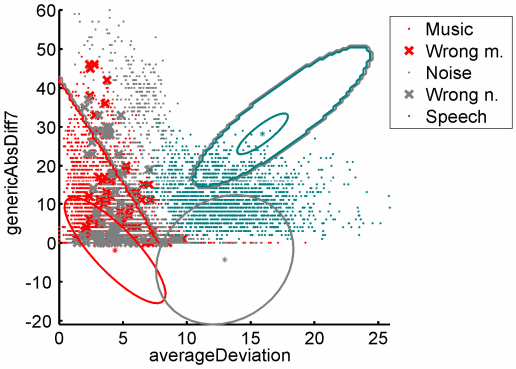


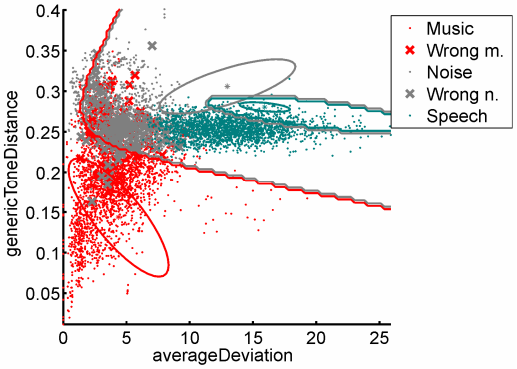

genericReliabilityDev
genericAbsDiff7
Music
Wrong m.
Noise
Wrong n.
Speech
Wrong s.

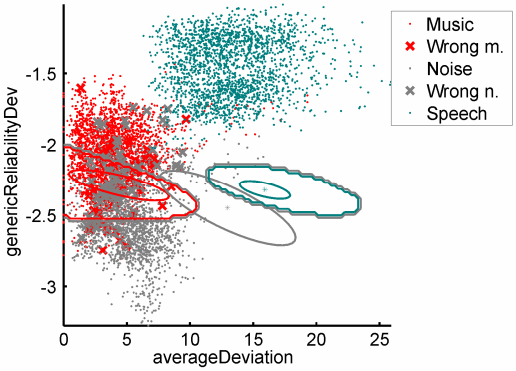

genericReliabilityDev
genericToneDistance
Music
Wrong m.
Noise
Wrong n.
Speech
Wrong s.

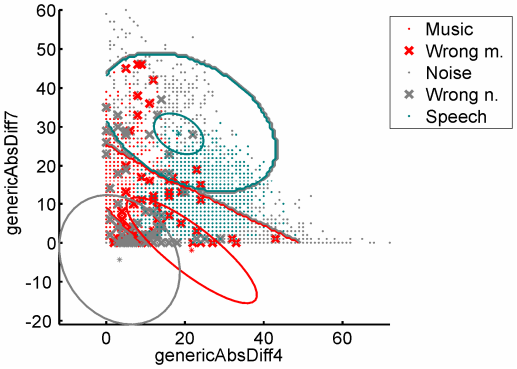
5 features

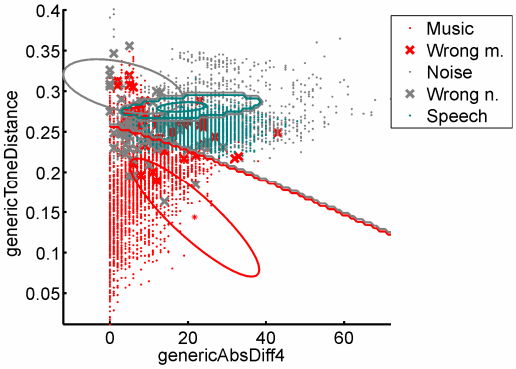

genericAbsDiff4
averageDeviation
Music
Wrong m.
Noise
Wrong n.
Speech

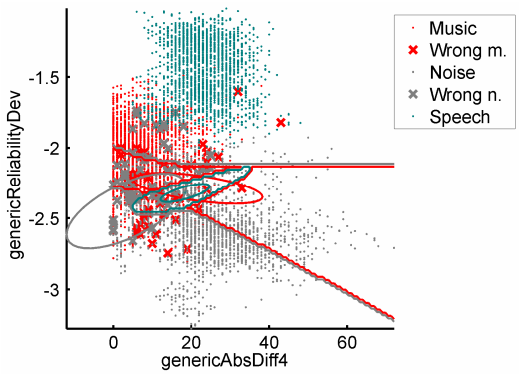

genericAbsDiff7
averageDeviation
Music
Wrong m.
Noise
Wrong n.
Speech

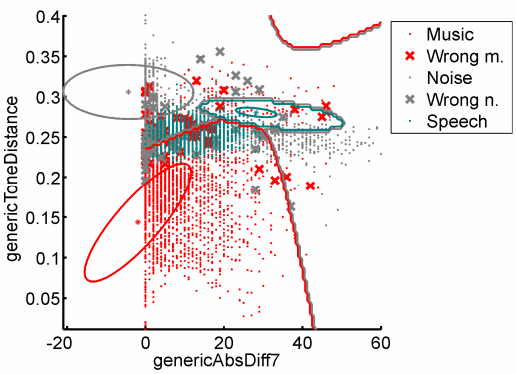

genericToneDistance
averageDeviation
Music
Wrong m.
Noise
Wrong n.
Speech

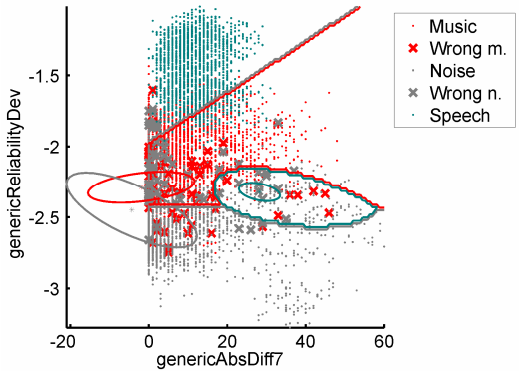

genericReliabilityDev
averageDeviation
Music
Wrong m.
Noise
Wrong n.
Speech

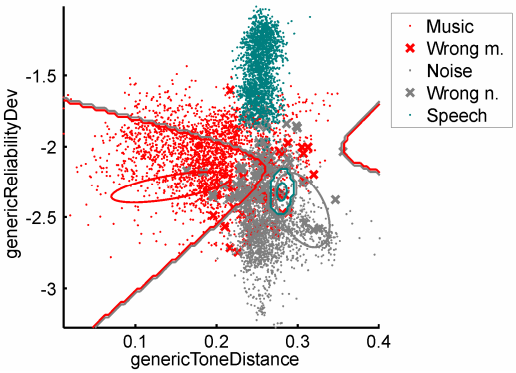

genericAbsDiff7
genericAbsDiff4
Music
Wrong m.
Noise
Wrong n.
Speech


genericToneDistance
genericAbsDiff4
Music
Wrong m.
Noise
Wrong n.
Speech


Music
Wrong m.
Noise
Wrong n.
Speech
genericReliabilityDev
genericAbsDiff4


Music
Wrong m.
Noise
Wrong n.
Speech
genericToneDistance
genericAbsDiff7


Music
Wrong m.
Noise
Wrong n.
Speech
genericReliabilityDev
genericAbsDiff7


Music
Wrong m.
Noise
Wrong n.
Speech
genericReliabilityDev
genericToneDistance