Analysis of Acoustic Feature Extraction Algorithms in Noisy Environments

by

Weiyang Cai

Submitted in Partial Fulfillment

of the

Requirements for the Degree

Master of Science in Electrical Engineering

Supervised by

Professor Wendi Beth Heinzelman

Department of Electrical and Computer Engineering
Arts, Sciences and Engineering
Edmund A. Hajim School of Engineering and Applied Sciences

University of Rochester
Rochester, New York

2013

Biographical Sketch

The author was born in Nanfeng, China on June 30, 1989. He attended Hong Kong University of Science and Technology from 2007 to 2011, and graduated with a Bachelor of Engineering degree in Electronic Engineering in 2011. He came to the University of Rochester in the Fall of 2011. He joined the Wireless Communication and Networking Group in Summer 2012, working on "Project Bridge" under the supervision of Professor Wendi Heinzelman.

Acknowledgments

I would like to express my greatest gratitude to the people who have helped and supported me. First, I would like to thank Professor Wendi Heinzelman for her continuous guidance and invaluable advice on my thesis. Many thanks to my parents for their undivided support and encouragement. I would also like to thank Na Yang and He Ba from the Wireless Communication and Networking Group for providing advice on my thesis.

This research was part of the Bridge project, which was supported by funding from National Institute of Health NICHD (Grant R01 HD060789).

Abstract

Acoustic feature extraction algorithms play a central role in many speech and music processing applications. However, noise usually prevents acoustic feature extraction algorithms from obtaining the correct information from speech and music signals. Thus, the robustness of acoustic feature extraction algorithms is an area worth studying. In this thesis, we consider two important acoustic features: pitch and speaking rate. For each acoustic feature, we introduce several classic and state-of-the-art feature extraction algorithms and evaluate the performance of each of them in noisy environments. We analyze the results and provide possible explanations why some feature extraction algorithms outperform the others in noisy environments.

Table of Contents

List of Tables

List of Figures

1 Introduction

Nowadays, speech and music technologies are receiving increasing attention. Applications such as speech recognition and automatic music transcription play an important role in human computer interactions and are widely used in a large number of mobile devices. For example, mobile applications such as Sound Hound use so-called "query-by-humming" technology to find the song a user sings to his/her iPhone [9]. Standard testing organizations such as ETS have developed some speaking rate estimation programs to automatically measure the examinees' language fluency [10]. Some speech-based emotion classification systems [11] [12] use the statistics of acoustic feature statistics of speech samples such as pitch to classify the emotion of a speech sample.

Extracting accurate acoustic features from signals is crucial for the functionalities of these applications. However, noise usually has a negative impact on speech and music. Many acoustic feature extraction algorithms can perform very well in a quiet environment. However, the performance of these algorithms in noisy environments is unknown. Hence, it is necessary to evaluate the performance of existing feature extraction algorithms in noisy environments. The evaluation results can provide good reference for developing noise-resilient software. This thesis achieves this goal by implementing several classic as well as state-of-the-art feature extraction algorithms for pitch and speaking rate and evaluating the

performance of these algorithms in noisy environments.

1.1 Thesis Outline

Chapter 2 introduces some classic and state-of-the-art pitch estimation algorithms and speaking rate estimation algorithms. In Chapter 3, databases for the evaluation are introduced and then a thorough comparison is conducted to compare the performances of pitch estimation algorithms and speaking rate algorithms in noisy environments for each database. Chapter 4 presents the conclusions and potential future work.

1.2 Contributions

Although we can find some examples in the literature that describe the evaluation of pitch estimation algorithms in noisy environments [13][14], my thesis includes two recent pitch estimation algorithms, BaNa [1] and SAFE [6]. In addition, I test pitch estimation algorithms not only on noisy speech signals but also on noisy music, which provides more comprehensive evaluation results. Few studies have been done on evaluating speaking rate estimation algorithms in noisy environments. Hence, my work on evaluating the performance of speaking rate is novel.

2 Feature Extraction Algorithms

This chapter first introduces windowing, a basic technique for signal analysis. Then two basic acoustic features (pitch and speaking rate) are introduced. A number of classic and state-of-the-art pitch estimation algorithms and speaking rate estimation algorithms are introduced as the foundation of our algorithm evaluations in Chapter 3.

2.1 Windowing

Since acoustic signals that need to be analyzed often last for a few seconds or minutes and the features (e.g., pitch, loudness, formant) are usually varying over time, a common technique called windowing is used to analyze the signal over a short period of time. The window size is chosen such that the feature is expected to remain unchanged in this period.

A Hamming window [15] is widely used in speech and music processing because it has a narrower main lobe and smaller side lobes compared with other windows such as a triangular window [16]. Thus the Hamming window is closer to an impulse in the frequency domain than other windows.

Window size and window shift size are two variables that need to be considered

when utilizing windowing of an acoustic signal. A larger window provides better frequency resolution because it includes more samples for analysis. However, with a larger window, the time sensitivity is reduced. Window shift size determines the amount of overlap of consecutive frames.

2.2 Pitch Estimation

Pitch is defined as the property of an acoustic signal that is determined by the frequency of the waves producing it: pitch thus represents the highness or lowness of sound. Strictly speaking, the term "pitch" should be regarded as the auditory perception of tone. Pitch is an inherently subjective quantity and cannot be directly measured from the acoustic signal. It is a nonlinear function of the signal's spectral and temporal energy distribution [17].

Fundamental frequency ($F0$) is the quantity that is measured by almost all pitch estimation algorithms. $F0$ is defined as the inverse of the period of a signal. The perceived pitch of a signal is highly correlated with its fundamental frequency.

Pitch estimation plays a vital role in many speech and music processing applications. For speech, pitch estimation tracks the pitch contour of the speech, and hence pitch works as an important component of speech recognition algorithms. For music, pitch estimation maps detected frequency to certain musical notes. Many pitch estimation algorithms have been developed [17]. However, none of the existing methods can guarantee the detection accuracy in the presence of high levels of additive noise.

In this section, a number of classic and state-of-the-art pitch estimation algorithms are introduced. The evaluation of these algorithms in the presence of noise is presented in Chapter 3.

The commonly used pitch estimation algorithms often have three stages: 1) pre-processing of the acoustic signal, 2) possible pitch candidates generation, and

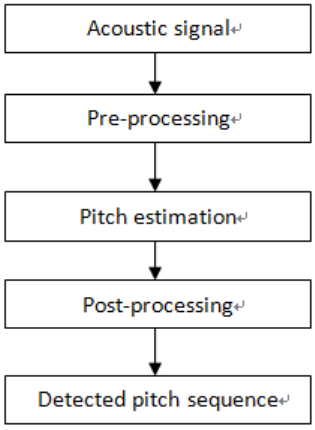

Figure 2.1: The work flow of a typical pitch estimation algorithm.

3) post-processing to select the best choice among the candidates to optimize the estimation of $F0$ [18].

2.2.1 Pre-processing

The purpose of pre-processing is to eliminate interfering signal components.

Filtering:

Filtering is the most common technique used in the pre-processing stage of pitch estimation. For example, the pitch range for speech is 50Hz to 600 Hz. A band pass filter can effectively remove frequencies outside the desired bandwidth, which improves $F0$ estimation performance.

Non-linear operations:

One non-linear technique that is often used is to set a threshold in order to remove some low level noise. Another commonly used technique is center-clipping.

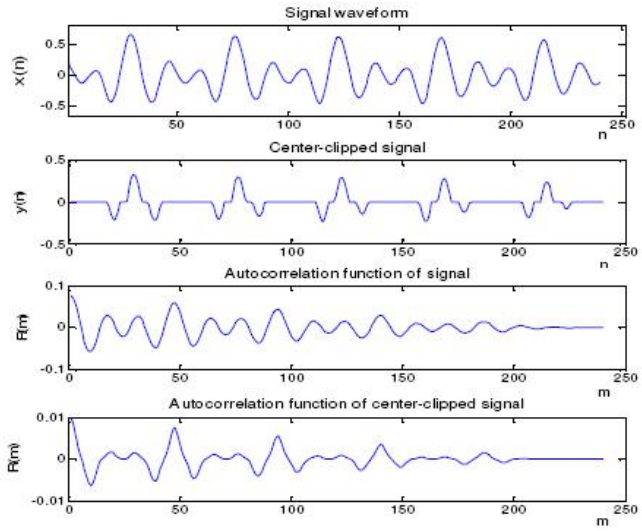

Figure 2.2: An illustration of the effect of center clipping, reprinted from [2].

The center clipping function is defined as follows [2],

$$y(n) = clc[x(n)] = \begin{cases} (x(n) - C_L) & x(n) \geq C_L \\ 0 & |x(n)| < C_L \\ (x(n) + C_L) & x(n) \leq -C_L \end{cases} \tag{2.1}$$

where C_L is the clipping threshold. Generally, C_L is about 50% of the maximum absolute value of the input signal. This technique tends to flatten the spectrum of the signal passed to the candidate generators, to be discussed in Section 2.2.2.

Figure 2.2 shows an example of a voiced frame, its center clipped version, and the difference between the autocorrelation function calculated from the original signal frame and the center-clipped signal frame. We can see that the desired peaks become more clear, while the other irrelevant peaks are flattened.

In speech processing, the pre-processing step also includes a voiced/unvoiced classification function that distinguishes voiced and unvoiced segments of the signal. Speech is composed of phonemes, which are produced by the vocal cords and vocal tract. Voiced signals are produced when the vocal cords vibrate during

the pronunciation of a phoneme. Unvoiced signals, by contrast, do not entail the use of the vocal cords [19]. Only voiced signals are evaluated by pitch estimation algorithms.

2.2.2 Generation of Pitch Candidates

Pitch estimation algorithms can be divided into three categories.

- Time domain approaches
- Frequency domain approaches
- Statistical approaches

Time Domain Approaches

Zero-crossing Rate:

The zero-crossing rate (ZCR) represents the rate of sign-changes in the signal (i.e., the rate at which the signal changes from positive to negative and vice versa) [20]. This technique is very simple and computationally inexpensive. Zero-crossing rate is defined as

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1}\Pi\{s_t s_{t+1} < 0\} \tag{2.2}$$

where s_t is the value of the signal at time t, and $\Pi\{A\}$ is 1 if the argument A is true and 0 otherwise. The frequency of the signal (F_{signal}) is given by

$$F_{signal} = \frac{zcr * fs}{2} \tag{2.3}$$

where zcr is calculated in Equation 2.2 and fs is the sampling frequency of the signal. However, under noisy conditions, the performance of this technique is very poor because low level noise can easily change the sign of a signal.

Autocorrelation:

This method calculates the dot-product of the original signal and a shifted version. The autocorrelation function $r(\tau)$ of a signal with time lag τ is defined as follows,

$$r(\tau) = \frac{1}{N} \sum_{n=0}^{N-1} x(n)x(n+\tau) \tag{2.4}$$

The autocorrelation function always has a global maximum for $\tau = 0$. If the signal is periodic, the autocorrelation function should have global maxima at multiples of the period of the signal T_0 such that $r_x(nT_0) = r_x(0), n = 1, 2, 3...$ [3]. In practice, $x(t)$ is usually a non-periodic windowed signal. Hence, no global maxima can be found outside $\tau = 0$. However, there can still be some local maxima. If the highest of the local maxima is at a time lag τ, and the value at this point is above a threshold, the signal is said to have a periodic part [3]. The fundamental frequency $F0$ is estimated to be $1/\tau$.

YIN:

The YIN algorithm uses a difference function based on the autocorrelation method [21]. While the autocorrelation function aims to maximize the product between the waveform and its shifted version, the difference function $d_t(\tau)$ aims to minimize the difference between the waveform and the shifted version.

$$d_t(\tau) = \sum_{j=1}^{W} (x_j - x_{j+\tau})^2 \tag{2.5}$$

where W is the size of the window.

In order to handle the quasi-periodic nature of pitch in real signals, the YIN algorithm normalizes the difference function by its cumulative mean and sets a value of 1 for $\tau = 0$, as

$$d'_t(\tau) = \begin{cases} 1 & \text{if } \tau = 0 \\ d_t(\tau)/[(1/\tau)\sum_{j=1}^{\tau} d_t(j)] & \text{otherwise} \end{cases} \tag{2.6}$$

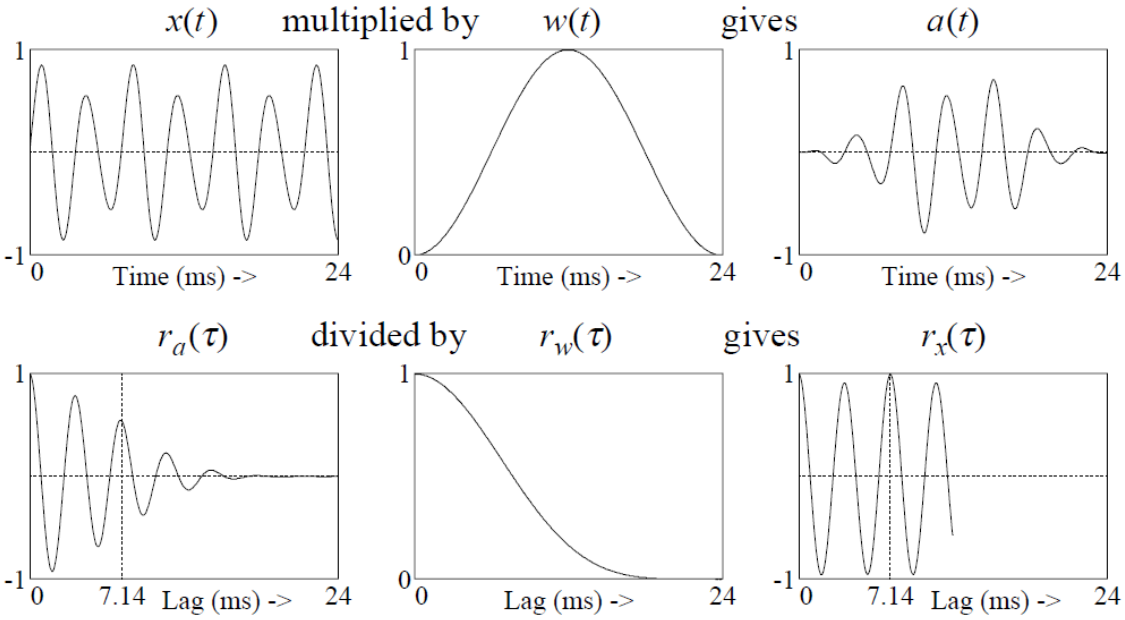

Figure 2.3: Flowchart of the Praat algorithm, reprinted from [3].

The last three steps involve placing a threshold on the smallest value of τ that is accepted. Also, parabolic interpolation is used to refine the peak location and searching around the initial pitch markers to further refine the estimate [17].

Praat:

The basic principle behind Praat is the autocorrelation method. As shown in Figure 2.3, a small segment of the normalized signal $x(t)$ is first multiplied by a (Hamming, Hanning) window $w(t)$ resulting in $r_\alpha(\tau)$. The autocorrelation function $r_\alpha(\tau)$ is then calculated using Equation 2.4. However, if we use the peak picking method introduced in the "autocorrelation method" section, an incorrect peak will be chosen. In the left bottom figure, the first peak is chosen instead of the correct peak at 7.14ms. $r_w(\tau)$ is the normalized autocorrelation of the window function $w(t)$. To handle the problem, the autocorrelation function $r_\alpha(t)$ is divided by $r_\omega(\tau)$. Then time lag T_0 of the maximum peak is estimated to be

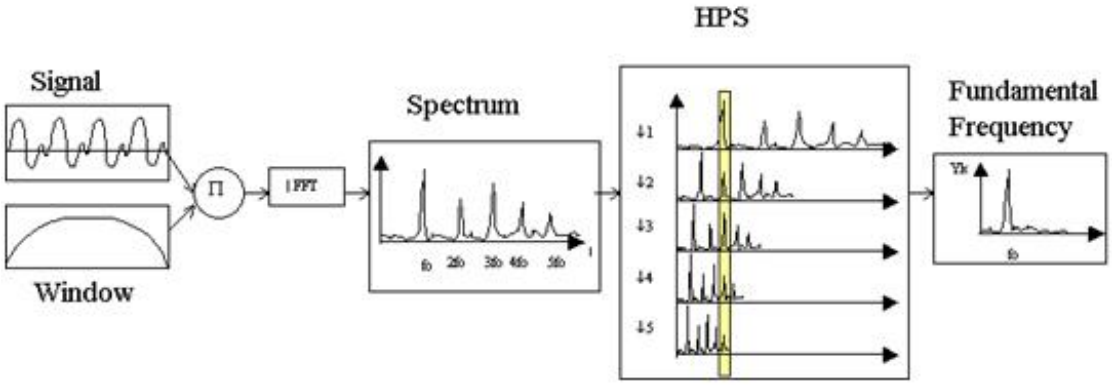

Figure 2.4: Flowchart of the HPS algorithm, reprinted from [4].

the period of $x(t)$ [3].

$$F0 = \frac{1}{\text{time lag of the maximum peak} \times \text{sampling frequency}} \tag{2.7}$$

Frequency Domain Approaches

Harmonic Product Spectrum (HPS):

HPS is a classic pitch detection method based on the fact that the peaks of the frequency spectrum are located at multiples of the fundamental frequency [4]. In HPS, the original frequency spectrum is down-sampled by $N\,(N = 2, 3, 4...)$, and then all of these spectra are multiplied together. The maximum peak indicates the fundamental frequency. Figure 2.4 presents the basic work flow of the HPS algorithm. After downsampling and spectra multiplication, only one prominent peak is left, which indicates the fundamental frequency.

Cepstrum:

The cepstrum is defined as the inverse DFT of the log magnitude of the DFT of the signal [5].

$$C(n) = \mathcal{F}^{-1} log|\mathcal{F}(x[n]) \tag{2.8}$$

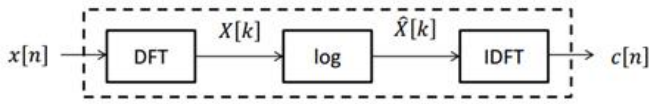

Figure 2.5: Flowchart of Cepstrum, reprinted from [5].

Table 2.1: Tolerance range for harmonic ratios [1].

Ratios	F_0	F_1	F_2	F_3
F_1	[1.9 2.1]			
F_2	[2.8 3.2]	[1.42 1.59]		
F_3	[3.8 4.2]	Discarded	[1.29 1.42]	
F_4	[4.8 5.2]	[2.4 2.6]	[1.59 1.8]	[1.15 1.29]

The cepstrum tends to have local maxima at $k * T$, corresponding to integer multiples of the glottal period. The log operator on the speech magnitude spectrum tends to flatten the harmonic peaks in the spectrum and thus leads to more distinct period peaks in the cepstrum [5]. The independent variable of a cepstrum graph is called the quefrency. The quefrency is a measure of time, though not in the sense of a signal in the time domain. For example, if the sampling rate of an audio signal is 44,100 Hz and there is a large peak in the cepstrum whose quefrency is 100 samples, the peak indicates the presence of a pitch that is 44100/100 = 441 Hz [22].

BaNa:

The BaNa pitch detection algorithm is a recently developed pitch detection algorithm that combines the idea of peak finding and Cepstrum to select the $F0$ candidates. BaNa also introduces the idea of a confidence score to select the pitch among several pitch candidates, and it incorporates the Viterbi Algorithm into post-processing to eliminate some unlikely candidates.

The algorithm first searches for harmonic peaks above a certain threshold and then selects the five peaks out of the set that have the lowest frequency. Then it

calculates the ratio of frequencies for every two harmonic peaks. If any calculated ratio falls into the tolerance range of the harmonic ratios shown in Table 2.1, a potential pitch candidate $\tilde{F}$ can be obtained by dividing the harmonic $\hat{F}$ by its ratio to $F0$: $\tilde{F} = \hat{F}/m$. The tolerance range idea is inspired by the fact that the ratios of the frequencies of harmonic peaks are not always integer. In addition, the pitch value found using the Cepstrum algorithm is also included as one of the pitch candidates [1].

The BaNa algorithm also has a variation called BaNa_music, which is specialized in detecting pitch for music. BaNa_music takes advantage of the fact that the harmonics of music have a wider frequency range so that instead of using an amplitude threshold and selecting the five lowest frequency peaks with an amplitude above the threshold, the five harmonic peaks with the highest amplitudes are chosen. The advantages of this change will be explained in Chapter 3.

Statistical Approaches

SAFE is an algorithm that utilizes a statistically-based soft-decision multi-band method. Here soft-decision means that the decision is made by combining information from different frequency bands of the signal. On the contrary, hard-decision means the decision is determined by the most reliable band of the signal [6]. The algorithm first extracts information from voiced frames. Then it estimates the value of $F0$ in a statistical way. Figure 2.6 presents the work flow of the SAFE algorithm.

After the routine pre-processing stage, the power spectrum of a voiced frame is denoted by $\mathbf{Y}$ and the noise level $\mathbf{N}$ is estimated by measuring the initial and final frames in the utterance. The maximum likelihood estimation of $F0$ is:

$$\hat{f}_0 = \arg\max_{f_0 \in S_{F0}} P(f_0|Y, N) \tag{2.9}$$

where $S_{F0} = \{f_{min}, f_{min} + \Delta, ..., f_{max}\}$ is a set of $F0$ candidates.

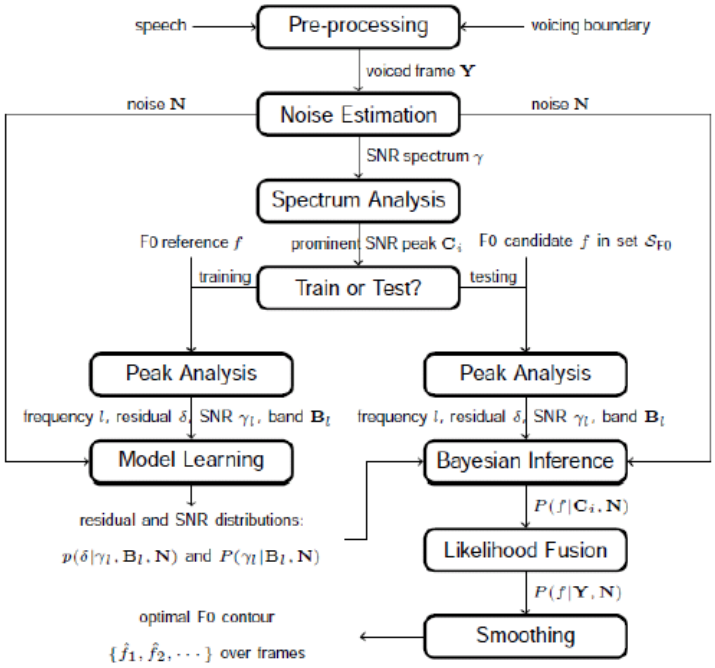

Figure 2.6: Flowchart of the SAFE algorithm, reprinted from [6].

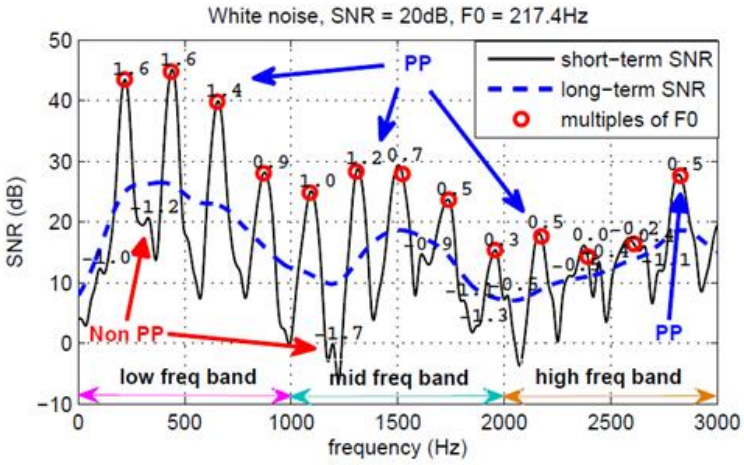

Figure 2.7: The SNR spectrum for white noise SNR-20dB, $F0$=217.4Hz, reprinted from [6].

Let Y_f and N_f denote the power spectrum of the noisy voice frame Y and noise N at frequency f, respectively. The posteriori SNR at frequency f denoted

by γ_f is:

$$\gamma_f = 10 \log_{10} \frac{Y_f}{N_f}. \tag{2.10}$$

We use γ_f^S and γ_f^L to represent the short-term smoothed and the long-term smoothed SNR spectrum, respectively. Figure 2.7 depicts the short-term smoothed and the long-term smoothed SNR spectrum of a voiced frame. ζ_f denotes the difference of these two spectra. The red circles indicate the prominent peaks.

$$\zeta_f = \gamma_{f_i}^S - \gamma_{f_i}^L, i = 1, ..., M \tag{2.11}$$

$$\bar{\zeta_f} = (\gamma_{f_i} - \mu_\gamma)/\sigma_\gamma, i = 1, ..., M \tag{2.12}$$

$\bar{\zeta_f}$ is the normalized SNR difference. An SNR peak is prominent if $\bar{\zeta_f}$ is greater than a threshold. After the spectrum analysis stage, we should be able to get all the prominent SNR peaks C_i in the spectrum.

Before going into the testing phase, a training model is required to be generated first. In the training phase, since $F0$ and γ_f are known, the overall posterior probability of f_0 is:

$$P(f_0|Y, N) = P(f_0|C_1, ..., C_M, N). \tag{2.13}$$

We assume all the SNR peaks are independent in inferring $F0$, the posterior probability can be approximated by a weighted combination of posterior probabilities $P(f_0|C_i, N)$:

$$P(f_0|Y, N) \approx \sum_{i=1}^{M} w_i P(f_0|C_i, N). \tag{2.14}$$

Given $p(f_0|C_i, N)$ is uniformly distributed, according to Bayes's theorem,

$$P(f_0|C_i, N) = \frac{p(C_i|f_0, N)}{\sum_{f_0 \in S_{F0}} p(C_i|f_0, N)} \tag{2.15}$$

Let f denote the frequency of the local SNR peak C_i. Because f is not usually equal to a multiple of f_0, f can be decomposed into a multiple m and a residual δ as follows [6]:

$$m = [\frac{f}{f_0}], \delta = \frac{f}{f0} - m, \tag{2.16}$$

where $[\frac{f}{f_0}]$ denotes the nearest integer of $\frac{f}{f_0}$ and δ ranges from -0.5 to 0.5.

Given the $F0$, and the noise level, the probability of C_i being the peak corresponding to $F0$ is:

$$p(C_i|f_0, N) = p(m, \delta, \gamma_f, B_f|f, N) = D \cdot p(\delta|\gamma_f, B_f, N)p(\gamma_f|B_f, N), \tag{2.17}$$

where D is a constant, and B_f denotes the frequency band f resides in. The derivation in detail is described in [6].

The residual distribution can be approximated by:

$$p(\delta|\gamma_f, B_f, N) \approx p(\delta|Q_{\gamma_f}, B_f, N) \tag{2.18}$$

where Q_{γ_f} denotes the SNR bin to which γ_f is rounded.

The local SNR distribution can be approximated by:

$$p(\gamma_f|B_f, N) \approx p(Q_{\gamma_f}|B_f, N) \tag{2.19}$$

$p(\delta|\gamma_f, B_f, N)$ and $p(\gamma_f|B_f, N)$ generated in the training stage are applied to the testing stage in order to generate $P(f_0|C_i, N)$. $P(f_0|Y, N)$ can be derived from Equation 2.14. The resulting $F0$ contour can be obtained by applying Equation 2.9.

2.2.3 Post-processing

Detection errors sometimes occur in the selection of the pitch of a given frame of the signal. The most common of these errors are a doubling or a halving of the fundamental period [18]. Post-processing can help compensate the effect of such errors. Common methods include smoothing and dynamic programming.

Smoothing: Generally, smoothing is implemented by a median-filter. Cepstrum incorporates smoothing into the post-processing stage to avoid doubling or halving errors.

Dynamic Programming: The earliest use of dynamic programming in pitch estimation was reported in [23], Later, it was clearly outlined in [24] how dynamic programming can be applied to the joint problem of estimating and smoothing speech parameters. The YIN, Praat, BaNa and SAFE algorithms all introduce their own cost function and minimize the total cost along the path by using the Viterbi algorithm.

Let $\tilde{F}_i^n$ denote the i^{th} pitch candidate of frame n and N_{frame} denote the number of frames in the given speech segment. Let p_n denote the index of the chosen pitch candidate for the n^{th} frame. Hence, $\{p_n | 1 \leq n \leq N_{frame}\}$ defines a path through the candidates. For each path, the path cost is defined to be [1]

$$PathCost\left(\{p_n\}\right) = \sum_{n=1}^{N_{frame}-1} Cost\left(\tilde{F}_i^n, \tilde{F}_j^{n+1}\right), \tag{2.20}$$

where the function $Cost$ is used to calculate the cost of adjacent frames.

There are many ways to define the function $Cost$. In BaNa, $Cost$ is defined by using the pitch differences between the adjacent frames and the confidence score of the candidates. The confidence score of a candidate is determined by the number of candidates within a threshold value of each other. The larger the pitch difference among neighboring frames, the higher the $Cost$ should be. The complete cost function is defined mathematically as

$$Cost\left(\tilde{F}_i^n, \tilde{F}_j^{n+1}\right) = \left|\log_2 \frac{\tilde{F}_i^n}{\tilde{F}_j^{n+1}}\right| + w \times \frac{1}{V_i^n}, \tag{2.21}$$

where V_i^n is the confidence score of the i^{th} pitch candidate of the n^{th} frame [1].

The Viterbi algorithm is also used in Praat, YIN and SAFE. Each pitch estimation algorithm defines a different cost function to find the minimum path. The implementation of the cost function in detail for each of these algorithms can be found in [3][21][6].

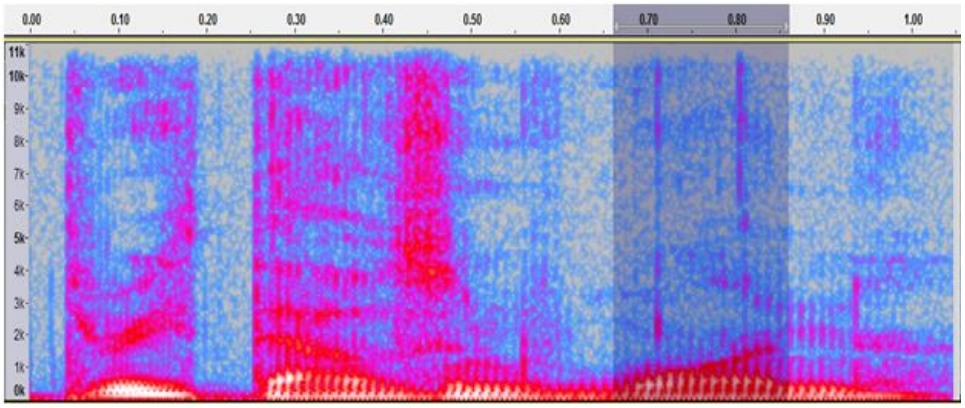

Figure 2.8: Spectrogram of an utterance "Three thousand one."

2.3 Speaking Rate Estimation

Speaking rate is a metric that measures how many words a speaker has pronounced within a certain time duration. The variation of speaking rate has a negative impact on the accuracy of automatic speech recognition. However, speech rate can convey important information to enhance speech understanding. For example, speech rate is the best predictor of subjective fluency.

A popular method to measure speaking rate involves counting the number of phonetic elements per time unit. Syllable-based rate estimation is the most widely used technique for speech rate researchers [25].

Words can be cut into units called syllables. A syllable is typically vowel centric, and adjacent vowels are separated by consonants. According to the Handbook of the International Phonetic Association (IPA), generally vowels form the nucleus of syllables, whereas consonants forms the boundaries in between [26]. For example, "window" contains two syllables, "win" and "dow". "w" and "d" are consonants which separate vowels "in" and "ow". Though there are some exceptions like "react" where two vowels are concatenated together, the number of vowels often equals the number of syllables. Hence, counting the number of syllables is basically equivalent to counting the number of vowels. In contrast to

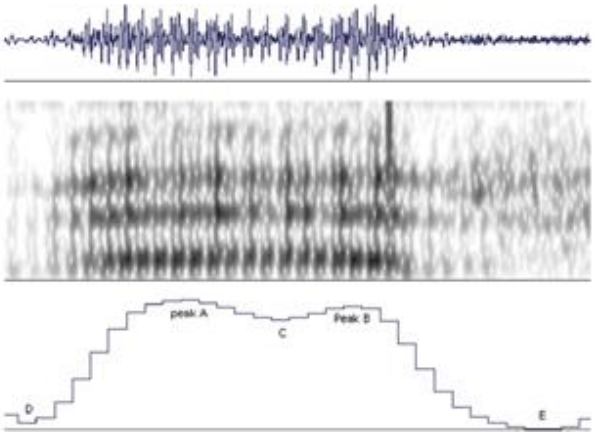

Figure 2.9: The waveform, spectrogram and the energy envelop of an utterance "Bad." Illustration of a case that a single syllable displays two peaks, reprinted from [7].

consonants, vowels exhibit clear formant structure in the spectrum. This characteristic is the foundation of syllable detection. Figure 2.8 is an illustration of the spectrogram of a speech signal. Brighter colors represent larger amplitudes. We can see from the figure that the energy is mainly distributed in the vowel sound and the lower frequency bands.

A simple way to do syllable counting is to perform energy analysis and count the number of prominent peaks of the long-term energy envelope. The energy envelope is composed of the energy value within each frame in time order. The energy within each frame with size w is calculated as follows,

$$E = \sum_{j=1}^{w} x_j^2 \tag{2.22}$$

where x_j is the jth sample inside the frame

However, under noisy environments, many false peaks will appear in the energy envelope, which leads to false detections.

N.H. de Jong and T. Wempe developed an algorithm based on this simple approach to automatically measure speaking rate [25]. The algorithm avoids false

peak detection by setting several thresholds. First, the median value of the energy envelop is chosen as the peak amplitude threshold. Only peaks above the threshold are considered as syllable candidates. The algorithm also takes advantage of the fact that adjacent syllables are often separated by a consonant so that they cannot be very close. Hence, a peak temporal distance is set to eliminate one of the two peaks that are too close. In addition, voiced/unvoiced information provided by a pitch estimation algorithm can help remove peaks that appear in unvoiced frames.

The energy envelope only considers full band energy. As a result, formant structure or power spectral density, which are very important for syllable identification, are ignored. Morgan and Fosler-Lussier [27] proposed a sub-band based method that computes a trajectory that is the average product over all pairs of compressed sub-band energy trajectories.

$x_i(n)$ represents the compressed energy envelope of the i^th spectral band. A new trajectory $y(n)$ is defined as

$$y(n) = \frac{1}{M} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} x_j(n) x_i(n). \tag{2.23}$$

where N is the number of bands, and $M = N(N-1)/2$ is the number of unique pairs. The new trajectory is often called a correlation envelope.

Since a vowel has a clearer formant structure, the energy should be more evenly distributed in each sub-band. In contrast, the energy of a consonant mostly concentrates in the low frequency band. Hence, the correlation amplitude of a vowel is maximized while not significantly increasing the contribution to the envelope from the consonants. As a result, the neighboring syllables should have a deeper gap in between [7].

D. Wang extends the sub-band idea to the time domain based on the fact that vowels and sonorant consonants, which constitute the major body of a syllable, extend over several tens of milliseconds [7]. For each desired frequency sub-band,

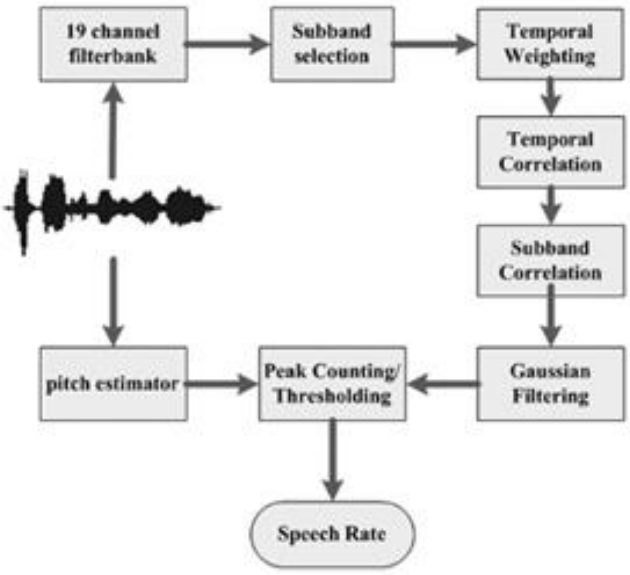

Figure 2.10: Flowchart of the hybrid speaking rate estimator, reprinted from [7].

the energy in each time frame is recalculated using the following equation,

$$y_t = \sqrt{\frac{1}{2K(K-1)} \sum_{j=0}^{K-2} \sum_{p=j+1}^{K-1} x_{t+j} * x_{t+p}} \tag{2.24}$$

where K is the temporal window length, and x_t, x_{t+1},..., x_{t+K-1} represent an increasing time order of sub-band energy vectors with length K. In this way, each syllable should have a peak at its center [7].

Wang also points out in [7] that instead of performing spectral correlation on the full bands, it is better to concentrate on the prominent sub-bands where the formant structure lies. Figure 2.10 illustrates the work flow of Wang's proposed speaking rate estimation algorithm. The choice of sub-bands will be discussed in Chapter 3.

In Chapter 3, all speaking rate methods that appear in this section will be tested and evaluated under noisy environments.

3 Evaluating Acoustic Feature Extraction Algorithms in Noisy Environments

In this chapter, we investigate the acoustic feature extraction algorithms described in Chapter 2 in a range of noisy environments. In particular, we experiment with a number of pitch estimation and speaking rate detection algorithms in environments where the acoustic signal is corrupted with additive noise, including white noise, babble noise, pink noise, and others for different levels of signal-to-noise (SNR) ratios.

3.1 Speech and Music Signals

In this thesis, we only study monophonic signals, which means that at any time there only exists one correct pitch value. For speech, only one speaker is speaking. For music, only one musical note is playing. Polyphonic signals are out of the scope of our study. Hence, the databases we use are carefully chosen.

Table 3.1: Evaluated speech databases and their features [1].

Speech databases	**Emotion**	**Number of speakers**	**Number of selected samples (M = male, F = female)**	**Has pitch ground truth?**
LDC [28]	various	7	20 (10M/10F)	No
Arctic [29]	neutral	4	10 (5M/5F)	No
Harvard Sentences [30]	neutral	6	10 (5M/5F)	No
CSTR [8]	neutral	2	100 (50M/50F)	Yes

3.1.1 Speech Databases

All speech samples are taken from four widely used English speech databases: LDC (Linguistic Data Consortium) [28], Arctic [29], Harvard Sentences [30], and CSTR [8]. We choose approximately the same number of speech samples from male and female speakers. In addition, we include as many speakers as possible. Table 3.1 [1] presents the specifications of each of the speech databases we use in this thesis.

The LDC database is the Emotional Prosody Speech and Transcripts Database from Linguistic Data Consortium (LDC). It includes speech samples with strong emotions such as angry, happy and disgust, for which the pitch values may change dramatically even within a short utterance [1].

The Arctic database consists of around 1150 utterances carefully selected from out-of-copyright texts from Project Gutenberg. The database includes US English male and female speakers as well as other accented speakers (Midwest, Ontario, South Eastern Scottish) [29].

The Harvard Sentences database is a collection of sample phrases that are used

for standardized testing of voice over IP (VOIP), cellular, and other telephone systems. They are phonetically-balanced sentences that use specific phonemes at the same frequency they appear in English [30].

The CSTR database is provided by the University of Edinburgh. This database includes 50 sentences each from a male and a female speaker. The total length of all 100 sentences is 5 minutes and 32 seconds. The database also includes the laryngograph of each sentence, which acts as the $F0$ ground truth [8]. Ground truth here means the absolute frequency of the speech signal at a given time. It serves as the reference for measuring the detection accuracy.

3.1.2 Music Databases

Due to the versatility of timbres generated by various musical instruments, we experiment with some music samples that can represent all music. Generally, musical instruments can be classified into the following categories: string, brass, woodwind, keyboard. To show the overall performance of the pitch detection algorithms for music, we pick one instrument from each category. Music samples from violin, trumpet, clarinet and piano are selected to represent string, brass, woodwind and keyboard instruments, respectively. These music samples can be downloaded from [31]. All the music samples were recorded in a quiet environment, and they were transcribed manually so that we know exactly which music note is playing at any moment.

3.2 Noise Samples

To test the noise resilience of the investigated acoustic feature extraction algorithms, eight types of noise are added to the original acoustic signals (speech or music) with different SNR levels. The noise database we use is from [32], in

which we choose six different types of real life background noise: speech babble (labeled as babble in the figures), destroyer engine room noise (engine), destroyer operations room noise (operation), factory floor noise (factory), vehicle interior noise (vehicle), high frequency radio channel noise (highfreq), white noise (white) and pink noise (pink). To generate noisy speech with a certain SNR value, the signal energy is calculated only on the voiced part, and the noise is amplified or attenuated to a certain level to meet the target SNR value [1].

3.3 Pitch Estimation for Noisy Acoustic Signals

In this section, we investigate the performance of several state-of-the-art pitch estimation algorithms on speech and music signals in various noisy environments and for a wide range of SNR values. The following six algorithms are chosen because of their popularity and good performance for clean signals: BaNa [1], YIN [21], HPS [4], Praat [3], Cepstrum [22], and SAFE [6]. These algorithms have been described in detail in Chapter 2. The source code for BaNa, YIN, HPS, Cepstrum, and SAFE can be obtained from [33][34][35]. Praat is a free software available at [36].

3.3.1 Experiment Settings

In order to detect pitch in an acoustic signal, the signal is broken up into overlapping frames, and a pitch value is estimated for each frame. In our experiments, we set the frame length to 60 ms, with a frame shift set to 10 ms for all algorithms except YIN in order to obtain smooth pitch detection results [1]. For YIN, we use its default settings of which the minimum frequency is set to 30Hz, the maximum frequency is set to one fourth of the sampling frequency, the frame length is set to the sampling frequency fs divided by the minimum frequency, and the frame shift is set to 32 samples.

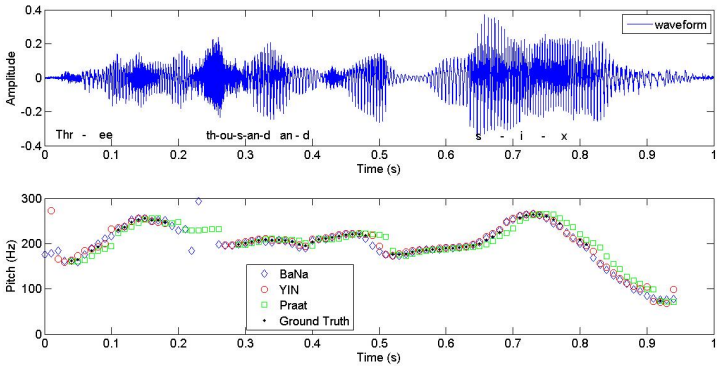

Figure 3.1: Speech waveform and the auto-labeled ground truth derived from three algorithms for one speech utterance.

We extract the pitch contour of a speech or music signal using each algorithm and then compare the results with the ground truth pitch values. For speech, since CSTR is the only database that provides $F0$ ground truth, we must generate the ground truth from clean versions of the speech for the other databases. For music, since we know what notes are played at different times, we can obtain the ground truth from the notes. Given the ground truth, we can evaluate the performance of the different pitch estimation algorithms under noisy environments.

Figure 3.1 presents the pitch contours of a clean utterance "three thousand six" generated from BaNa, YIN and Praat. For one frame, if the estimated pitch values of the three algorithms are within 10% of each other, we assume that this frame is voiced. For each voiced frame, $F0$ ground truth is determined by taking the average the three estimated pitch values.

The evaluation metric we used in pitch estimation is defined as follows,

$$accuracy = \frac{\text{Number of correctly estimated voiced frames}}{\text{Total number of voiced frames}} \times 100\% \qquad (3.1)$$

For each frame, the pitch is estimated correctly, if

$$|\frac{\text{pitch estimated}}{F0 \text{ ground truth}} - 1| < \text{error tolerance} \qquad (3.2)$$

For the Arctic, Harvard Sentences and LDC databases, the error tolerance is set to 10%. For the CSTR database, the error tolerance is set to 20% in order to be consistent with the that used in the SAFE experiments [1].

3.3.2 Pitch Estimation for Speech

We tested the BaNa, YIN, HPS, Cepstrum, and Praat pitch estimation algorithms on the speech signals in the Arctic, Harvard Sentences, CSTR and LDC databases. For BaNa, Cepstrum and Praat, according to the frequency range of human speech, we set the minimum frequency to 50Hz and the maximum frequency to 600Hz. For YIN, we use its default settings described in Section 3.4.1. No parameters need to be set for HPS. Pitch estimation accuracies of all tested algorithms are evaluated at 0dB, 3dB, 7dB, 10dB, 15dB and 20dB SNR levels for each of the 8 noise types. The accuracy values are averaged over all 8 types of noise in the figures. The SAFE algorithm and the BaNa algorithm are two pitch estimation algorithms designed to be robust to noise. We tested both of these algorithms on the CSTR database at 0dB, 10dB and 20dB .

Figure 3.2 presents the pitch estimation performance of all the pitch estimation algorithms on the LDC database. BaNa achieves the best accuracy of 70.0% at 0dB, an improvement over the second best algorithm YIN of around 10%. To show the benefit of adding the pitch candidate found by the Cepstrum method to the candidates derived from the harmonic ratios, we also show the pitch detection accuracy of the LDC database for the BaNa algorithm without the Cepstrum candidate, which is denoted as BaNa-noCepst in the figure. The accuracy of BaNa-noCepst is around 68%, which is still better than YIN's 59% and Praat's 35% at 0dB. YIN should have room for improvement if the parameter settings such as frame length, max frequency are optimized such that its performance is close to Bana's.

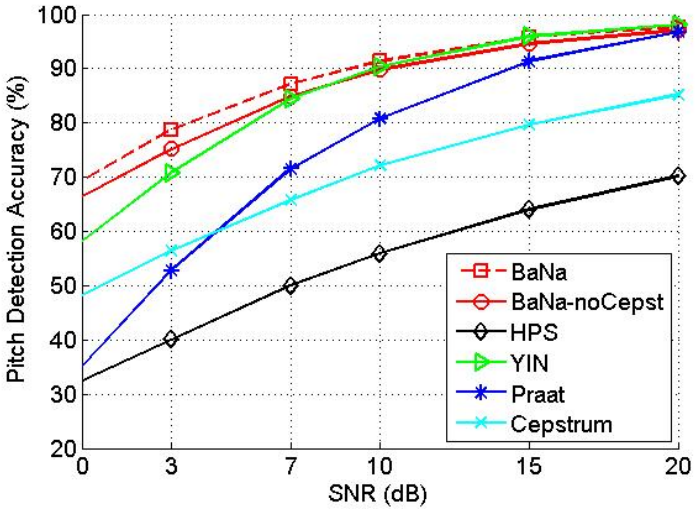

Figure 3.2: Pitch detection accuracy of the different algorithms for the LDC database.

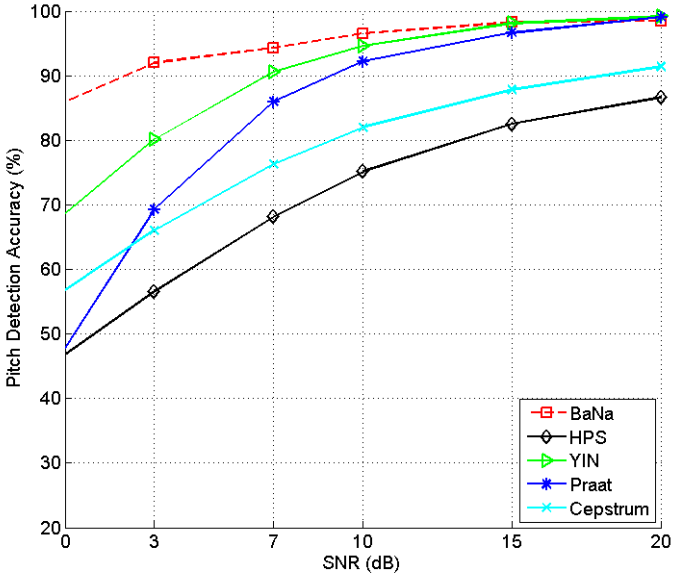

Figure 3.3: Pitch detection accuracy of the different algorithms for the Arctic database.

In Chapter 2, we discussed how both BaNa and HPS exploit the fact that spectral peaks appear at mutiples of the fundamental frequency. From Figure 3.2,

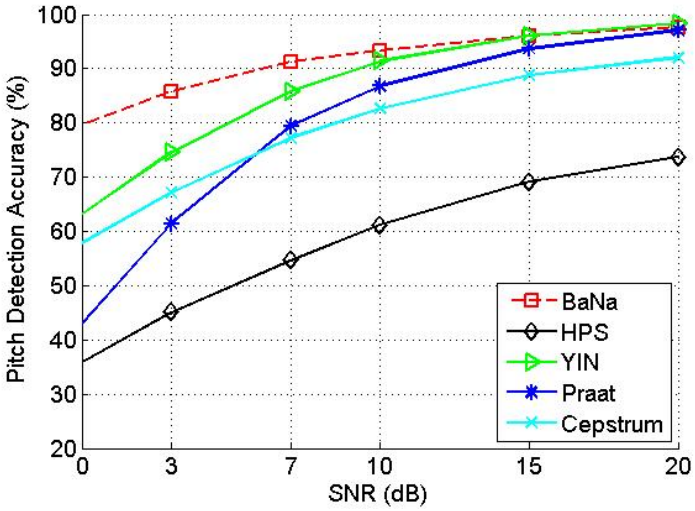

Figure 3.4: Pitch detection accuracy of the different algorithms for the Harvard Sentences database.

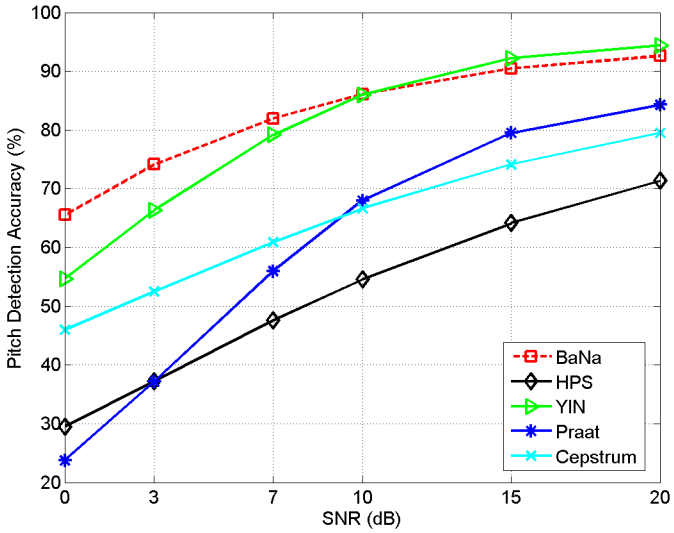

Figure 3.5: Pitch detection accuracy of the different algorithms for the CSTR database.

we can see that HPS's accuracy is much worse than BaNa. At 20dB SNR, the accuracy of HPS is only around 70% compared with BaNa's 98%. One reason why HPS performs poorly is that it assumes all peaks should be at multiples

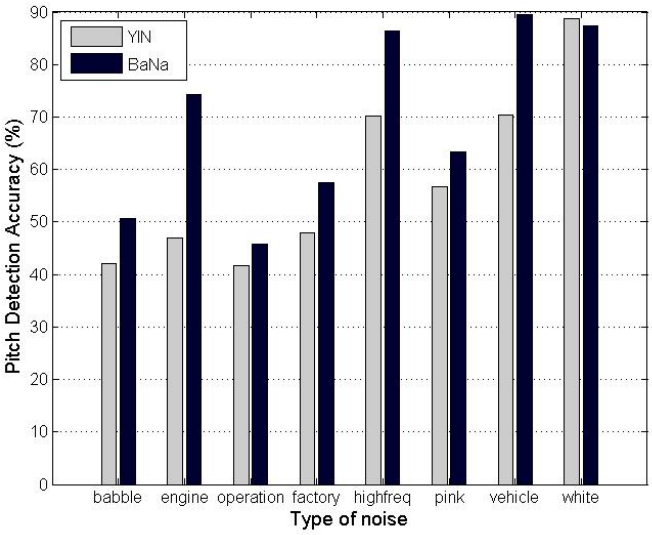

Figure 3.6: Pitch detection accuracy of BaNa and YIN for the LDC database with eight types of noise at 0dB.

of the fundamental frequency. However, in real speech, the ratios of harmonic frequencies are not exactly integers. In contrast, BaNa introduces the idea of a tolerance range (Table 2.1) to find the real harmonic peaks.

Praat achieves very good pitch estimation accuracy when the SNR value is high, reducing the accuracy by a mere 5% and 3% compared with BaNa's accuracy value at 15dB and 20dB, respectively. However, the performance of Praat degrades most quickly among all the selected algorithms. Thus, Praat may not be as robust as the other competing algorithms to noisy environments. In addition, Praat has an in-built voiced/unvoice detection function. At low SNR levels, some voiced frames will be detected as unvoiced if we keep using the settings for a clean environment. This is one of the reasons why Praat does not perform well at low SNR levels. For Cepstrum, since the algorithm is designed for a clean environment and no special techniques have been used to handle noise, it is reasonable that Cepstrum does not have as good a performance as BaNa and YIN.

Figure 3.6 shows a performance comparison between the two top performing algorithms (BaNa and YIN) for all eight types of noise at 0dB. It is quite clear from this graph that BaNa outperforms YIN for all eight types of noise.

Figure 3.3, Figure 3.4 and Figure 3.5 present the evaluation results for the Arctic, Harvard Sentences and CSTR databases, respectively, which are similar to the results for the LDC database. At 0dB SNR, the pitch estimation accuracy of BaNa is 86% for the Arctic database, 79% for the Harvard Sentences database and 65% for the CSTR database.

Hence, we conclude that BaNa is the most accurate and robust algorithm for speech pitch estimation among all the selected algorithms under a range of noise conditions.

SAFE is a recently developed pitch estimation algorithm that claims to achieve a very high accuracy for low SNR values. Here, we compare SAFE with BaNa, the most robust algorithm in the previous evaluations. The CSTR database is used for the evaluation of the SAFE algorithm in the original paper describing SAFE [6], so we use the same database here. Because the SAFE algorithm needs to apply a training model specifying the noise type and SNR level before testing new samples, we simply use the training models provided by the author. In [6], the SAFE algorithm is only trained on white noise and babble noise, so we only compare the results for these two types of noisy speech. In the comparison, we also adopt the SNR values provided by the SAFE paper. In order to ensure that the comparison is fair, all the settings of the two algorithms such as frame length, frame skip size, minimum frequency and maximum frequency are set to the same values as the previous evaluations. The error tolerance is set to 20% to be consistent with the SAFE paper [6].

The results are presented in Figure 3.7 and Figure 3.8, for white and babble noise, respectively. In addition to reporting the results from the SAFE paper [6], we rerun the SAFE algorithm using the code downloaded from [35]. The rerun

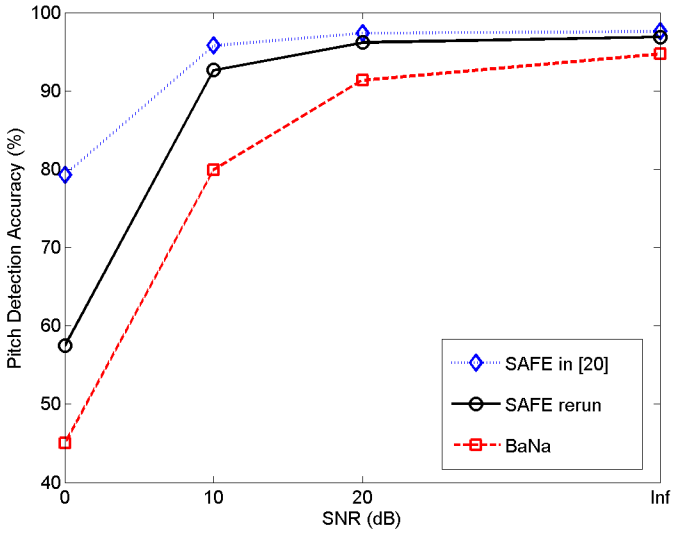

Figure 3.7: Pitch detection accuracy of BaNa and SAFE for the CSTR database [8] for speech with babble noise.

results of SAFE (denoted as "SAFE rerun") are slightly worse than the results presented in [6]. One possible reason is that the settings of SAFE used in [6] are different from those in our experiment. Even so, the rerun results of SAFE are better than the results of BaNa, especially for speech corrupted by babble noise. However, before running the SAFE algorithm, we need to know the noise type and SNR level in order to apply the correct training model, while BaNa does not require any prior knowledge of the noise type or SNR level. Therefore, SAFE can provide more accurate pitch detection accuracy in noisy environments, but BaNa has the advantage of being suitable for a range of environments with no training or parameter setting required based on the noise type or the SNR level.

3.3.3 Pitch Estimation for Music

The spectral characteristics of speech and music are quite different. Music has a much wider frequency range (30Hz-4000Hz) than speech (50Hz-600Hz). Addi-

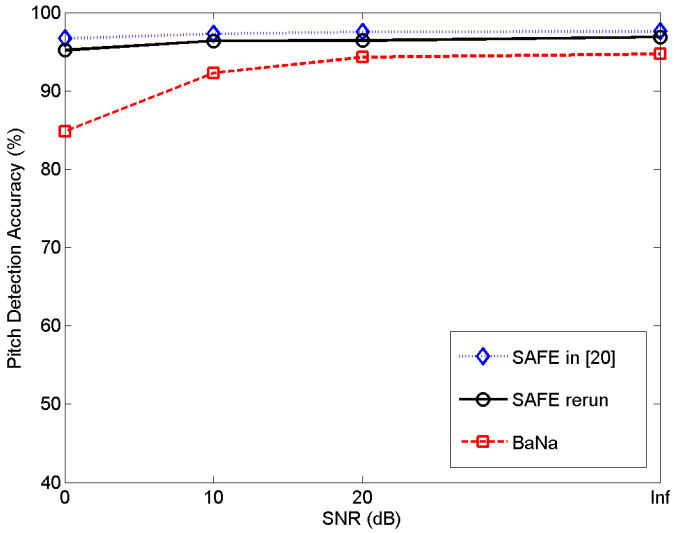

Figure 3.8: Pitch detection accuracy of BaNa and SAFE for the CSTR database [8] for speech with white noise.

tionally, the vibration model of each instrument is different and thus produces different timbre.

For pitch estimation for music, the ground truth is manually transcribed by mapping the music signal at a given time to a musical note. The absolute frequency of each musical note is fixed. For example, A4 maps to 440Hz. We assume that the frequency is constant from the starting point to the ending point of any musical note. We also remove frames in which two or more fundamental frequencies exist due to echo or cords. Thus, we ensure that the music we evaluate is mostly monophonic. Later we talk about fast tempo music having overlapping $F0$ due to dying out of the previous note.

Since the purpose of music pitch estimation is to map sound to a musical note, the frequency difference between the estimated pitch and the ground truth should not exceed a minor second (for example, C to C#), otherwise an error occurs. Hence, the error tolerance when evaluating pitch detection for music is set to a more stringent 2.5% compared to the 10% we used for speech evaluation. 2.5%

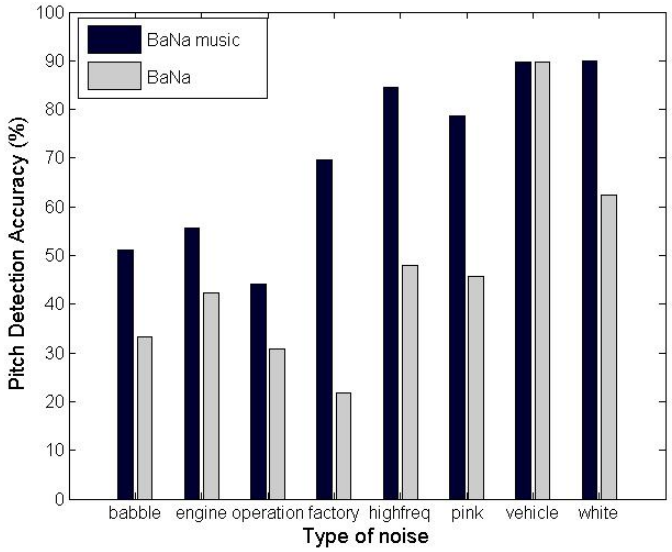

Figure 3.9: Pitch detection accuracy of BaNa and BaNa music for a piece of violin music with eight types of noise at 0dB.

frequency deviation is around half of a minor second.

As mentioned in Chapter 2, the BaNa_music algorithm is a variation of the original BaNa algorithm. The only difference is that instead of using an amplitude threshold and selecting the five lowest frequency peaks with an amplitude above a threshold, the five harmonic peaks with the highest amplitudes are chosen. The reason for the change is that noise can generate a lot of peaks with high amplitude in low frequency band. Thus the harmonic peaks may not be selected by the original BaNa algorithm. Fig 3.9 depicts the pitch detection accuracy of BaNa and BaNa_music for a piece of violin music with eight types of noise at 0dB. We can see that this minor change significantly improves the estimation accuracy for almost all types of noisy music. Hence, we use BaNa_music instead of BaNa for music evaluation.

We use the same experiment settings as that used in the evaluation of pitch estimation in noisy speech, except that we change the estimation range for BaNa,

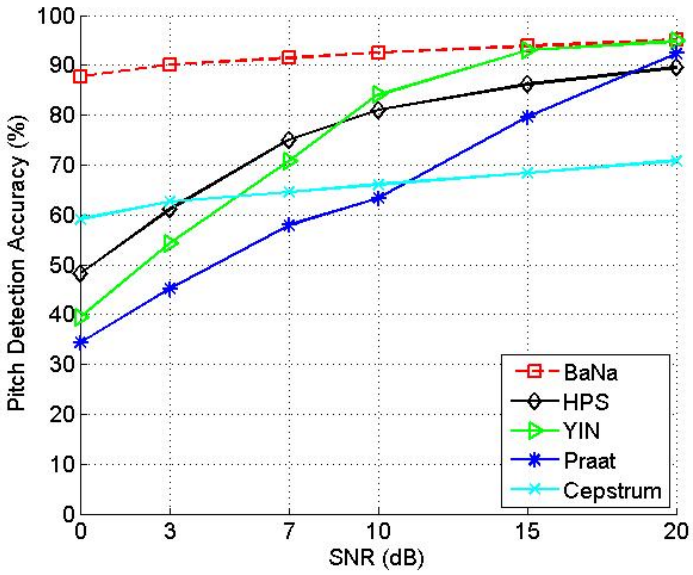

Figure 3.10: Pitch detection accuracy of the different algorithms for a piece of clarinet music.

Cepstrum and Praat from 50Hz-600Hz to 50Hz-4000Hz. For YIN, we still use its default settings described in Section 3.4.1. No parameters need to be set for HPS. SAFE is not evaluated for music because there is no discussion about music in [6], thus no training model for music is provided.

Figures 3.10 to 3.12 show the pitch detection accuracy of the different algorithms for clarinet, trumpet and violin, respectively, averaged over the eight types of noise. All figures show that the BaNa algorithm has the highest pitch estimation accuracy among all the algorithms. At 0dB SNR level, BaNa leads the second highest accuracy by 30%, 30%, and 20%, respectively, for clarinet, trumpet and violin. BaNa is also the best performer for all SNR levels.

In order to see how the tempo of music affects the pitch estimation accuracy, we select one piece of fast tempo piano music and one piece of slow tempo piano music. The piece of fast tempo piano has 3.4 notes per second, and the piece of slow tempo piano has one note per second [1]. Figure 3.13 shows the pitch

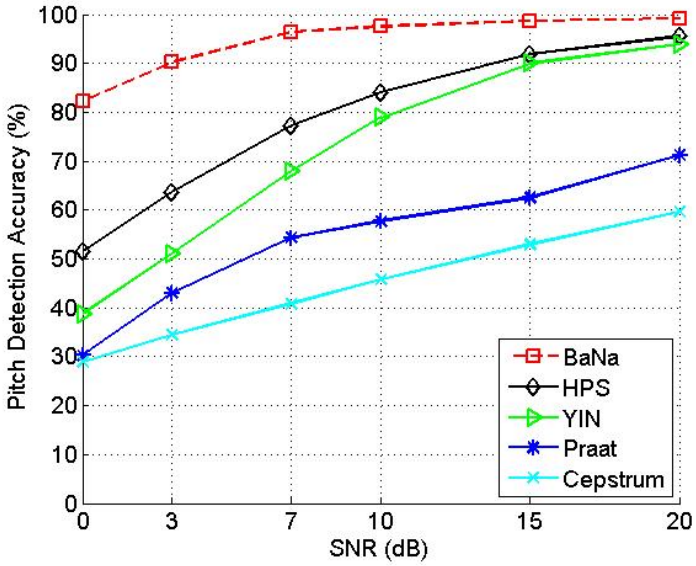

Figure 3.11: Pitch detection accuracy of the different algorithms for a piece of trumpet music.

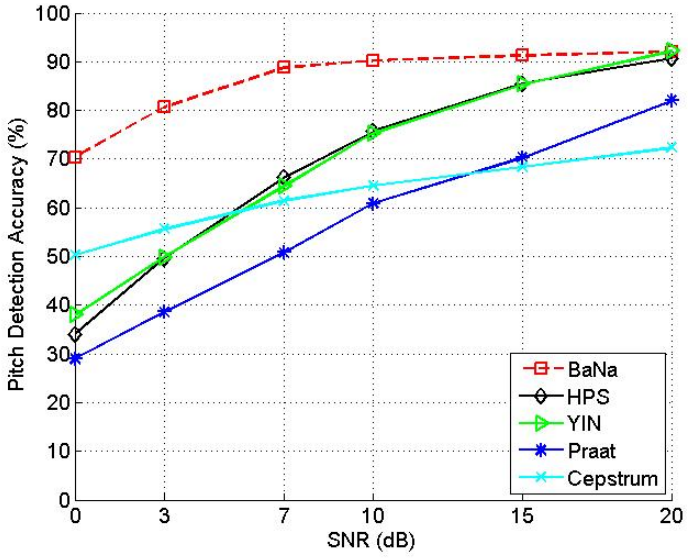

Figure 3.12: Pitch detection accuracy of the different algorithms for a piece of violin music.

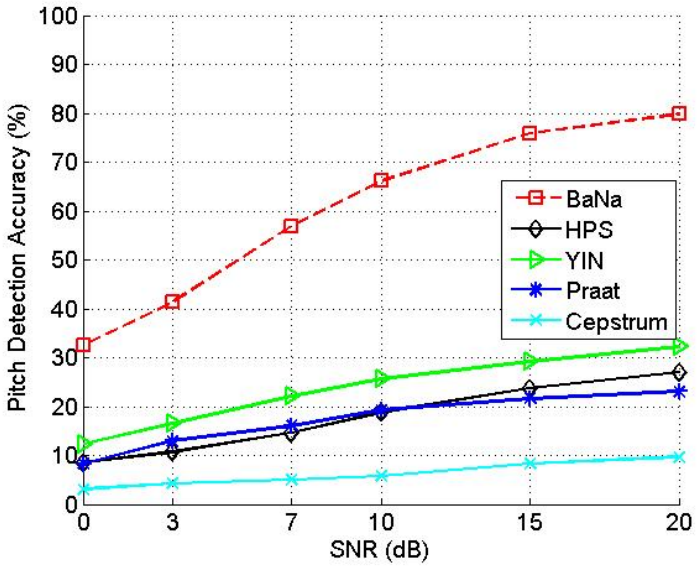

Figure 3.13: Pitch detection accuracy of the different algorithms for a piece of fast tempo piano music.

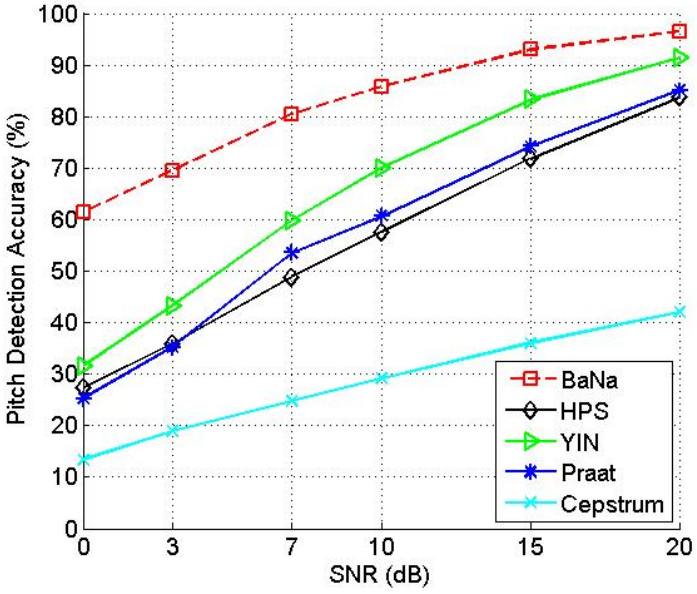

Figure 3.14: Pitch detection accuracy of the different algorithms for a piece of slow tempo piano music.

estimation accuracy for fast tempo piano music. The results show that BaNa performs much better than all the other algorithms. The average gap between BaNa and the second best algorithm is around 25%. At 20dB, BaNa achieves around 80% accuracy, while all the other algorithms are below 40%. One possible reason is that the music is too fast to be completely monophonic. Echoes of all notes are mixed together so that pitch estimation becomes very difficult. This result shows that BaNa is the best pitch estimation algorithm for fast tempo music.

We also observe that HPS achieves a huge improvement for pitch estimation in music compared with speech. We believe that the improvement is caused by more clear structure of harmonics in music compared with speech. We choose BaNa, YIN and HPS for a detailed comparison with eight different types of noised at 0dB SNR value. From Figures 3.15 to 3.19, we can see that BaNa achieves the highest pitch estimation accuracy for almost all pieces of music with all eight types of noise. As shown in Fig 3.18, BaNa achieves over 70% accuracy for high frequency noise and white noise, while the other algorithms only achieve below 35%. Therefore, we conclude that the BaNa algorithm is also the best pitch estimator for noisy music among all selected algorithms.

3.4 Speaking Rate Estimation for Noisy Acoustic Signals

In Chapter 2, we discuss several approaches to measure the speaking rate of a speech signal. Although researchers have tested the different algorithms on speech databases, in most of the experiments, the testing data is clean. In this section, we investigate all the speaking rate estimators described in Chapter 2 in noisy environments.

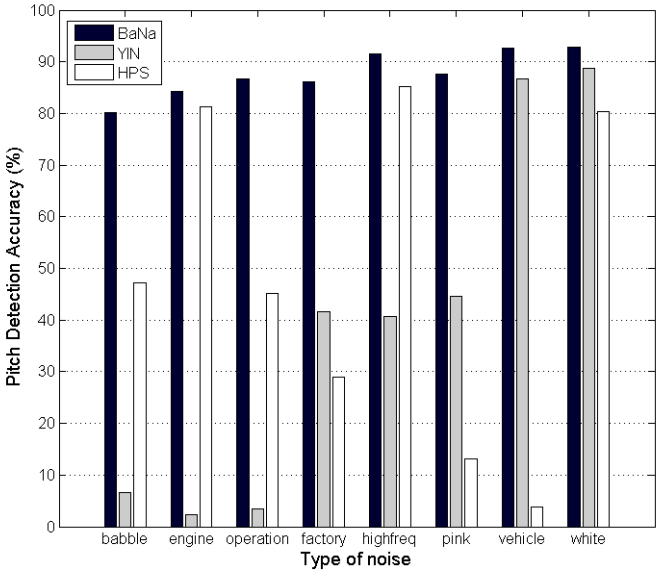

Figure 3.15: Pitch detection accuracy of BaNa, YIN and HPS for a piece of clarinet music with eight types of noise at 0dB.

Much of the research on speaking rate estimation uses the ICSI Switchboard corpus [37]. This corpus consists of 5757 utterances lasting for four hours in total. All utterances were phonetically hand transcribed by linguists. However, due to limited computation power, it takes a huge amount of time to process all the data. Also, we came across some compatibility problems when trying to read the WAV files from the ICSI Switchboard corpus. Therefore, instead we decide to choose the CSTR database introduced in Section 3.1 as our testing corpus. The CSTR database is composed of 100 utterances with transcriptions. We manually count the number of syllables for each utterance to determine the ground truth. The number of syllables of all 100 utterances ranges from 5 to 20. Each utterance lasts for around 2 to 4 seconds.

The noisy speech files generated in Section 3.2 are used here. We only select those speech signals that are corrupted by babble and white noise at 0dB, 10dB and 20dB SNR. Thus, including the clean speech files, a total of 700 files are

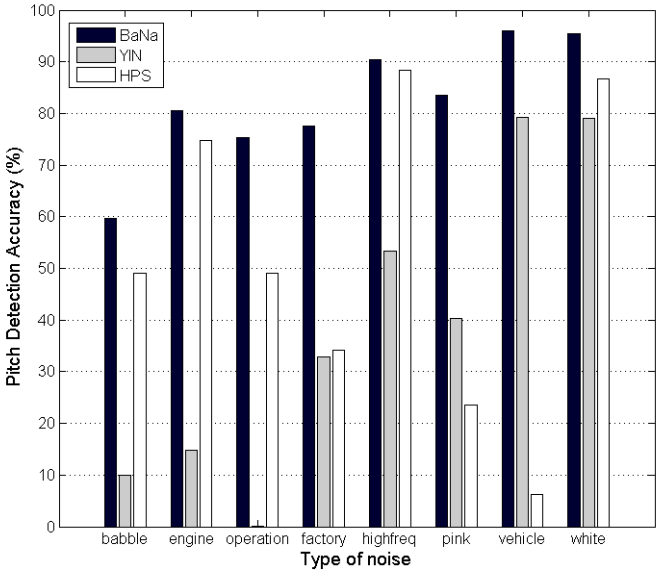

Figure 3.16: Pitch detection accuracy of BaNa, YIN and HPS for a piece of trumpet music with eight types of noise at 0dB.

evaluated (100 clean files plus 300 files for each type of noise).

Three speaking rate estimators are introduced in Chapter 2. The first estimator developed by N. H. de Jong and T. Wempe is written in Praat script. The method uses the full-band energy to generate the energy envelope, so we call this approach the "full band estimator." The Praat code for speaking rate estimation is available here [38]. The second speaking rate estimator, developed by N. Morgan and E. Fosler-Lussier takes advantage of formant structure information of human voice and adopts a sub-band energy correlation method. We call this approach the "sub-band estimator." The final speaking rate estimator, developed by D. Wang, extends the correlation idea from the frequency domain to the time domain, thus we call this approach "hybrid estimator." The code for the sub-band and hybrid estimators is not available on the web. Hence, we develop MATLAB code based on the algorithms described in [27] and [7]. The MATLAB code for all speaking rate estimation algorithms will be available at [33].

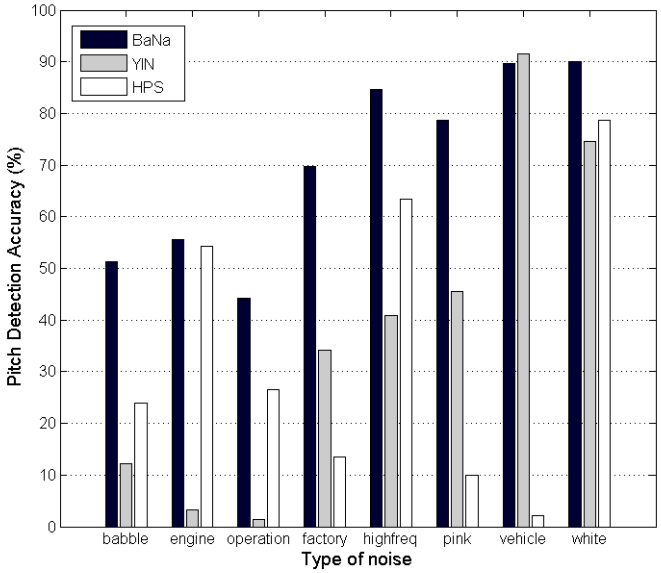

Figure 3.17: Pitch detection accuracy of BaNa, YIN and HPS for a piece of violin music with eight types of noise at 0dB.

3.4.1 Experiment Settings

In Section 2.3, we find that before using a selected speaking rate estimator, we need to determine several parameters. For all three estimators, we need to set a peak amplitude threshold and a threshold defining the amplitude difference between a peak and its preceding local minimum. For the sub-band and hybrid estimators, the number of selected sub-bands also needs to be determined. For the hybrid estimator, temporal correlation window length needs to be set.

We use the default settings for the full band estimator as provided by the author of the Praat script. For the sub-band and hybrid estimators, the parameter values are consistent with the settings in [7].

In [7], the author points out that the number of selected sub-bands (M) can affect the performance of the speaking rate estimator. Therefore, we test two sub-band estimators with $M = 4$ and $M = 12$. The hybrid estimator also uses 12

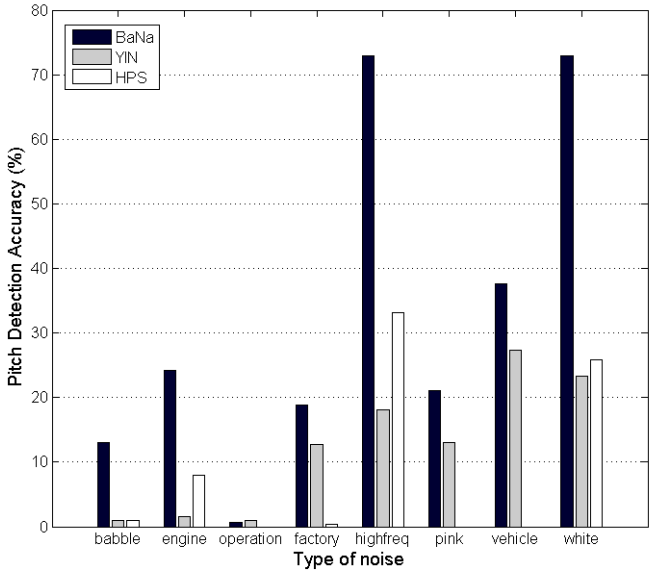

Figure 3.18: Pitch detection accuracy of BaNa, YIN and HPS for a piece of fast tempo piano music with eight types of noise at 0dB.

sub-bands. These two parameters are also used in [27] and [7]. Details about the boundary of each sub-band can be found in [27] and [39].

3.4.2 Evaluation Metrics

The ground truth indicates the number of syllables in the corresponding sentence. The size of the ground truth is 100 because there are 100 sentences in the CSTR database. Three evaluation metrics are used in determining the performance of the speaking rate estimation algorithms. All three metrics measure how the estimated results deviate from the ground truth.

The correlation coefficient measures the similarity between the estimated syllable count generated by an estimator and the ground truth. The more the estimated result matches the ground truth, the closer the correlation coefficient approaches a value of 1.

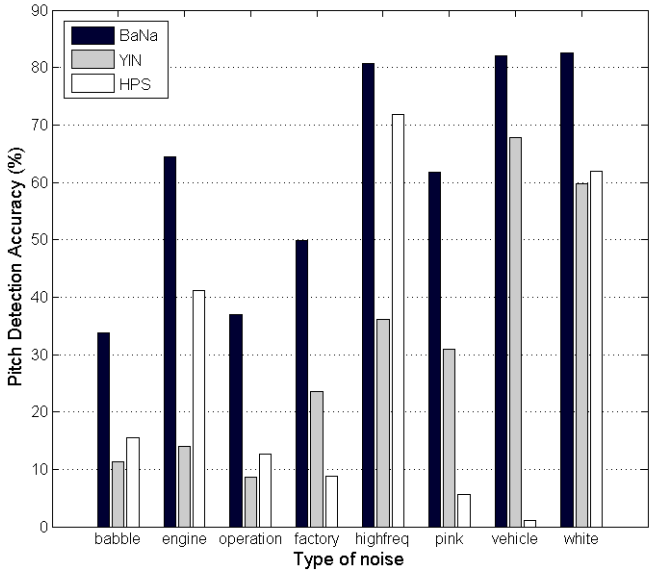

Figure 3.19: Pitch detection accuracy of BaNa, YIN and HPS for a piece of slow tempo piano music with eight types of noise at 0dB.

We also compute the mean square error between the syllable count and the ground truth, which can best be described as the accuracy of the estimator.

$$MSE\% = (\frac{\text{Estimated syllable count} - \text{Ground truth}}{\text{Ground truth}})^2 \times 100\% \tag{3.3}$$

The standard deviation of syllable count difference of all 100 sentences is also calculated to measure the stability of the different estimators.

3.4.3 Evaluation of the Speaking Rate Estimation Algorithms

We test all three estimators on clean and noisy speech files and calculate the three evaluation metrics described in Section 3.4.2. The results are summarized in Tables 3.2 3.4. The bold number in each row indicates the best estimator

Table 3.2: Correlation Coefficient, the more accurate the estimator is, the closer the value approaches one.

Correlation	**Full band**	**Sub-band M=4**	**Sub-band M=12**	**Hybrid**
Clean	0.8971	**0.9043**	0.9027	0.8782
0dB babble noise	0.6256	0.5278	0.4312	**0.6582**
10dB babble noise	**0.82**	0.7751	0.7863	0.5915
20dB babble noise	0.7944	**0.8648**	0.8453	0.8035
0dB white noise	**0.5996**	0.5974	0.5953	0.5856
10dB white noise	0.882	0.8368	**0.8946**	0.842
20dB white noise	0.9072	**0.9149**	0.9113	0.8459

for a certain situation (e.g., 0dB white noise). It is surprising that no single estimator demonstrates a big advantage over the other estimators. For example, in Table 3.3, the best estimators for 0dB babble noise, 10dB babble noise, and 20dB babble noise are the full-band estimator, the sub-band estimator $M = 12$, and the sub-band estimator $M = 4$, respectively. Since the sub-band $M = 4$, the sub-band $M = 12$ and the hybrid estimator all use sub-band correlation, we can treat them as one category to compare against the full band estimator. We see from Table 3.3 that the full band estimator achieves the highest accuracy for 0dB babble noise only. Thus, speaking rate estimation can benefit from sub-band correlation in noisy speech.

The mean square error (MSE) of the hybrid estimator [7] is 5.32% using the optimal parameters. In our experiments, the MSE% of the full-band, the sub-band $M = 4$, the sub-band $M = 12$, and the hybrid estimator are 5.17%, 4%, 3.99% and 4.04%, respectively.

Table 3.3: Mean Square Error. Ideal value will be 0 if the estimator is perfect.

Mean square error(%)	**Full band**	**Sub-band M=4**	**Sub-band M=12**	**Hybrid**
Clean	5.17	4	**3.99**	4.04
0dB babble noise	**11.78**	22.79	24.57	36.02
10dB babble noise	8.10	7.46	**6.85**	25.57
20dB babble noise	8.83	**6.74**	7.19	8.47
0dB white noise	44.53	47.18	50.43	**25.55**
10dB white noise	8.93	7.54	**5.27**	9.09
20dB white noise	5.48	**2.84**	3.30	5.79

Table 3.4: Standard Deviation. A stabler estimator should give smaller value.

Standard deviation	**Full band**	**Sub-band M=4**	**Sub-band M=12**	**Hybrid**
Clean	1.5692	1.1987	**1.1794**	1.2988
0dB babble noise	**2.231**	3.243	3.2474	2.66
10dB babble noise	**1.4859**	1.7984	1.7628	2.3462
20dB babble noise	1.75	**1.2589**	1.4101	1.4426
0dB white noise	3.165	3.2208	3.2275	**2.4303**
10dB white noise	1.8015	1.9574	**1.57**	1.6098
20dB white noise	1.5859	**1.0333**	1.1254	1.3371

We observe that at the 10dB and 20dB SNR levels, the estimation accuracy (measured by MSE%) of all four estimators for babble and white noise is just slightly worse than that for clean speech. For some odd cases, the accuracy for

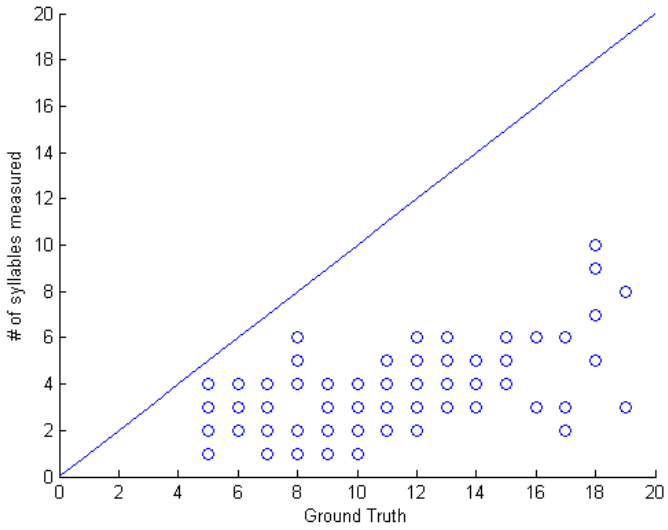

Figure 3.20: Scatter plots of detected syllable numbers versus the ground truth for the 100 utterances corrupted with 0dB white noise. Full band estimator.

noisy speech is even better than that for clean speech. For example, in Table 3.3, the MSE of white noise 20dB for the sub-band $M = 4$ estimator is 2.84%, better than the 4% MSE for clean speech. We can infer that the accuracy of the estimators is not reliable to detect all syllables correctly. The noise causes a small number of false detections, which helps improve the accuracy.

We also notice that at the 0dB SNR level, all estimators have very poor performance, especially for white noise. Figures 3.20 to 3.23 present scatter plots of the detected syllable number versus the ground truth for the 100 utterances corrupted with 0dB white noise.

From Figures 3.20 to 3.23, we can see that, with the exception of hybrid estimator, the other three estimators under count the syllables for all 100 testing cases. One possible explanation is that the parameter settings mentioned in Section 3.4.1 are only optimal for clean data. At 0dB SNR level, the noise adds many false peaks with large amplitude between two syllable peaks, such that syllable peaks are not detected because of settings such as minimum peak distance.

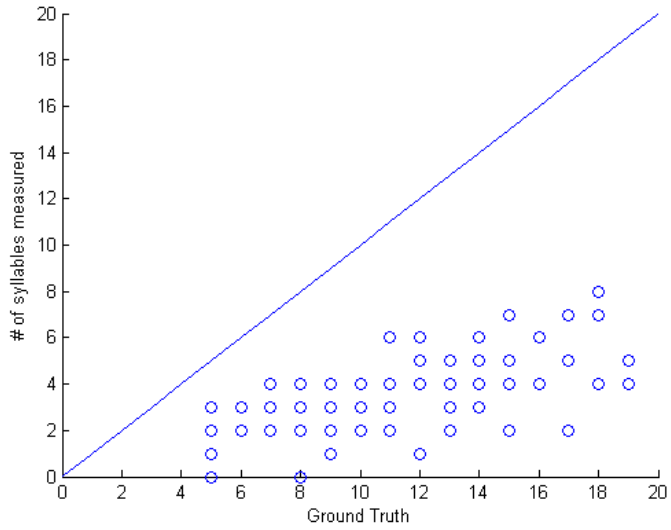

Figure 3.21: Scatter plots of detected syllable numbers versus the ground truth for the 100 utterances corrupted with 0dB white noise. Sub-band M=4 estimator.

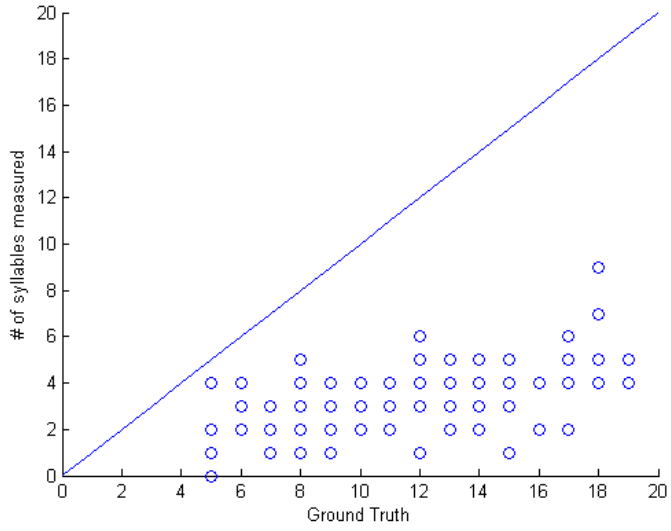

Figure 3.22: Scatter plots of detected syllable numbers versus the ground truth for the 100 utterances corrupted with 0dB white noise. Sub-band M=12 estimator.

The hybrid estimator is the most complex among all four estimators evaluated. However, its performance is not the best in our experiment, which contradicts the

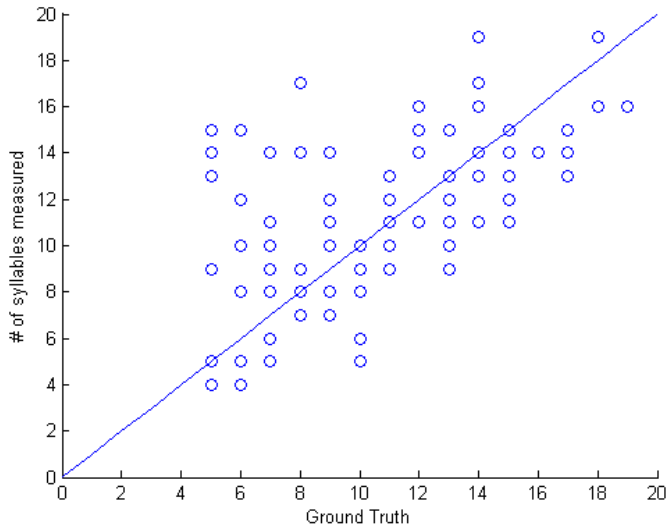

Figure 3.23: Scatter plots of detected syllable numbers versus the ground truth for the 100 utterances corrupted with 0dB white noise. Hybrid estimator.

testing results presented in [7]. Since we have written our own code to implement the estimator, there may exist some differences between our code and the original code, so the optimal settings in [7] may not apply to our case. To show the robustness of the hybrid estimator, a noisy utterance corrupted with 0dB white noise is picked from the speech corpus randomly to evaluate.

Figure 3.24 and Figure 3.25 display the waveforms of the utterance "I can't move my legs." Figure 3.24 is the clean version, while Figure 3.25 is the version that is corrupted by 0dB white noise. Figure 3.26 to 3.29 present the envelopes of the noisy utterance generated by the full band estimator, the sub-band $M = 4$ estimator, the sub-band $M = 12$ estimator, and the hybrid estimator, respectively. There should be five detectable syllables for this utterance. The envelope generated by all estimators except the hybrid one has numerous fake peaks. Since the hybrid estimator uses a temporal correlation window to smooth the envelope, only the five dominant syllable peaks are detected. Theoretically, the hybrid estimator should have the best performance in our experiments given optimal parameter

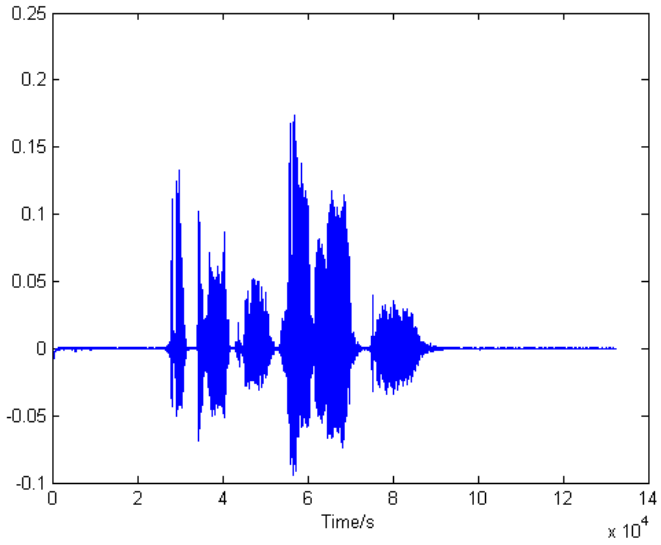

Figure 3.24: Clean speech. "I can't move my legs."

settings. Parameter selection is a tedious process and takes a large amount of time. Hence, we leave optimal parameter selection for speaking rate estimation as future work. In addition, the CSTR database is relatively small compared with the ICSI Switchboard speech corpus, which contains more utterances and has more speaking rate variation. We will test the ICSI Switchboard speech corpus in the future as well to validate our results.

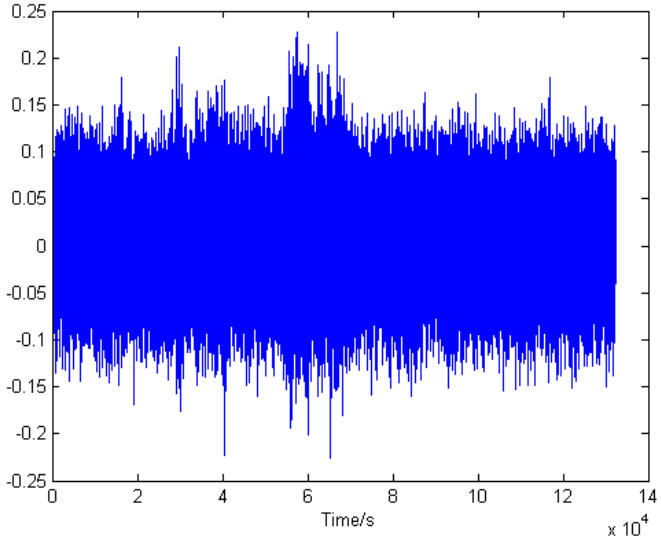

Figure 3.25: Noisy speech with 0dB white noise. "I can't move my legs."

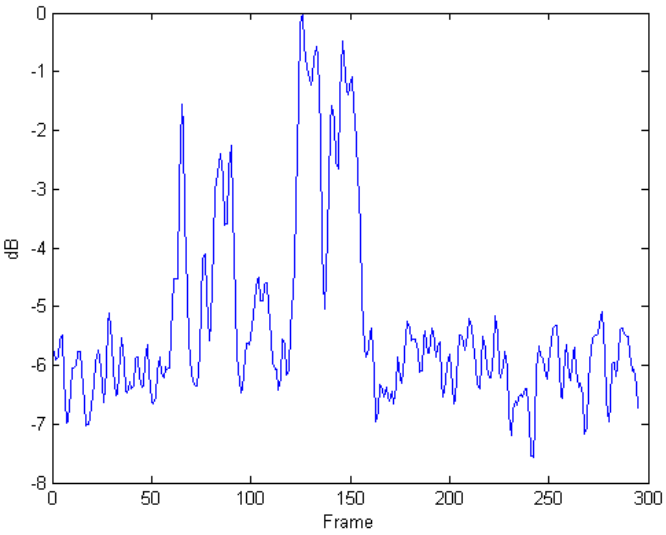

Figure 3.26: Envelope of the full band estimator.

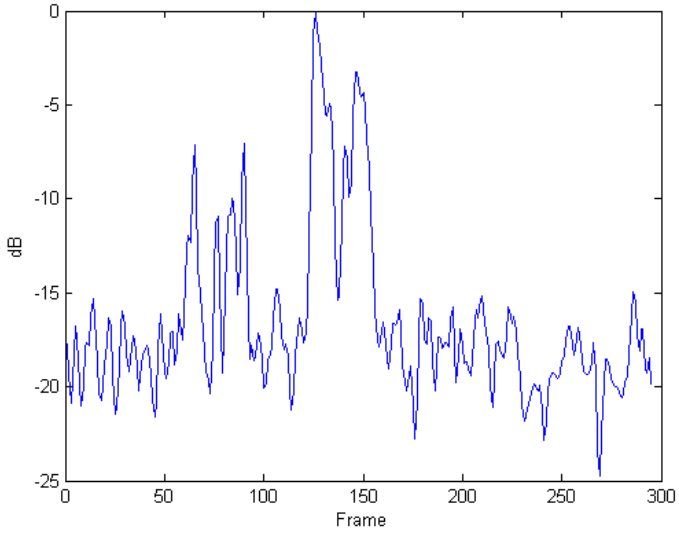

Figure 3.27: Envelope of the sub-band M=4 estimator.

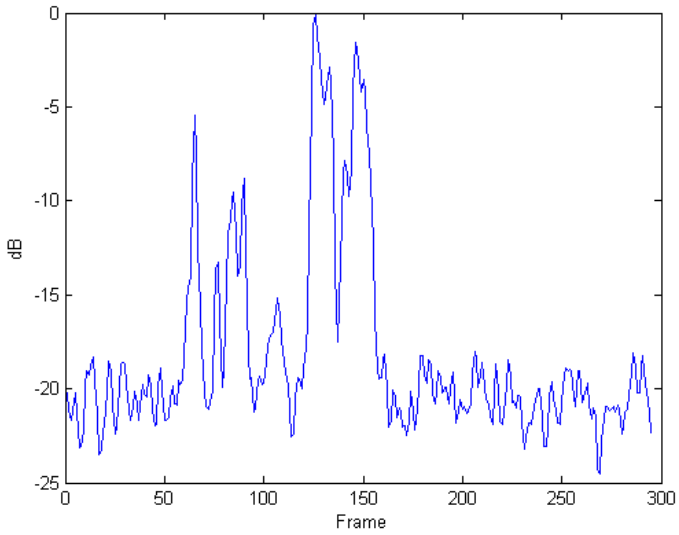

Figure 3.28: Envelope of the sub-band M=12 estimator.

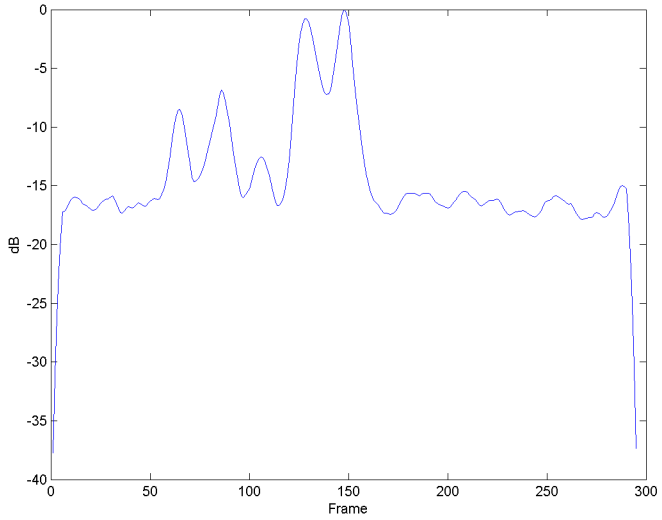

Figure 3.29: Envelope of the hybrid estimator.

4 Conclusions and Future Work

In this thesis, we discussed two important acoustic features of speech and music: pitch and speaking rate. We introduced several classic and state-of-the-art feature extraction algorithms for both pitch estimation and speaking rate estimation. We evaluated and compared the performance of the selected pitch and speaking rate estimation algorithms in noisy environments.

For the evaluation of pitch estimation algorithms in noisy environments, we tested BaNa, YIN, HPS, Cepstrum, and Praat on speech and music signals. We concluded that BaNa achieves the highest pitch estimation accuracy for all tested speech databases and music databases. We also compared the performance of BaNa and the recently developed SAFE algorithm on the CSTR database. We concluded that SAFE can provide more accurate pitch detection accuracy in noisy environments, but BaNa has the advantage of being suitable for a range of environments with no training or parameter setting required based on the noise type or the SNR level.

For the evaluation of speaking rate estimation algorithms in noisy environments, we discussed the full-band estimator, the sub-band estimator, and the hybrid estimator. From the experiments, we concluded that speaking rate estimation can benefit from using sub-band correlation. Although the hybrid estimator

did not achieve the highest accuracy among all the speaking rate estimators in the experiment because of non-optimal parameter settings, we demonstrated a case that only the hybrid estimator can effectively avoid fake peak detection under low SNR noisy environments, such that the performance is improved compared with the other speaking rate estimators.

Many things need to be improved in the future. For the evaluation of the pitch and speaking rate estimation algorithms, we only evaluated the estimation accuracy. However, so far we have not evaluated the time complexity of each algorithm, which is crucial for software systems. For the evaluation of speaking rate estimation algorithms, we did not use the optimal parameter settings for all selected estimators. Thus, our experiment may be biased. Also, we need to evaluate the performance of the algorithms on a larger database such as the ICSI Switchboard in the future to validate our experiment results.

Formant, another important acoustic feature, is not discussed here. Since not many databases provide the ground truth for formants, and Labeling the formants manually from the spectrogram requires a huge amount of time, we leave the evaluation of formant detection in noisy environments as future work.

Research such as speech emotion detection extracts pitch and speaking rate statistics in order to classify the emotions of the speech. We can apply different combinations of feature extraction algorithms to see how robust feature extraction algorithms help improve the performance of speech emotion detection in noisy environments.

Bibliography

[1] H. Ba, N. Yang, I. Demirkol, and W. Heinzelman, "Bana: A hybrid approach for noise resilient pitch detection," in *Statistical Signal Processing Workshop (SSP), 2012 IEEE*. IEEE, 2012, pp. 369–372.

[2] "Performance evaluation of pitch detection algorithms," http://access.feld.cvut.cz/view.php?cisloclanku=2009060001.

[3] P. Boersma, "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound," in *Proceedings of the institute of phonetic sciences*, vol. 17, no. 1193. Amsterdam, 1993, pp. 97–110.

[4] "Harmonic Product Spectrum," http://cnx.org/content/m11714/latest/.

[5] "Cepstrum analysis," http://research.cs.tamu.edu/prism/lectures/sp/l9.pdf.

[6] W. Chu and A. Alwan, "Safe: a statistical algorithm for f0 estimation for both clean and noisy speech," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[7] D. Wang and S. S. Narayanan, "Robust speech rate estimation for spontaneous speech," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 8, pp. 2190–2201, 2007.

[8] P. C. Bagshaw, S. Hiller, and M. A. Jack, "Enhanced pitch tracking and the processing of f0 contours for computer aided intonation teaching." 1993.

[9] "SoundHound Homepage," http://www.soundhound.com/.

[10] "ETS speakingRater," http://www.ets.org/research/topics/as_nlp/speech/.

[11] D. Bitouk, R. Verma, and A. Nenkova, "Class-level spectral features for emotion recognition," *Speech Communication*, vol. 52, no. 7, pp. 613–625, 2010.

[12] V. Sethu, E. Ambikairajah, and J. Epps, "Empirical mode decomposition based weighted frequency feature for speech-based emotion classification," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 5017–5020.

[13] L. Rabiner, M. Cheng, A. Rosenberg, and C. McGonegal, "A comparative performance study of several pitch detection algorithms," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 24, no. 5, pp. 399–418, 1976.

[14] K. Oh and C. Un, "A performance comparison of pitch extraction algorithms for noisy speech," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84.*, vol. 9. IEEE, 1984, pp. 85–88.

[15] "Hamming Window," https://ccrma.stanford.edu/~jos/sasp/Hamming_Window.html.

[16] "Triangular Window," https://ccrma.stanford.edu/~jos/sasp/Bartlett_Triangular_Window.html.

[17] D. Sharma and P. A. Naylor, "Evaluation of pitch estimation in noisy speech for application in non-intrusive speech quality assessment," in *Proc European Signal Processing Conf.* Citeseer, 2009, pp. 2514–2518.

[18] D. Talkin, "A robust algorithm for pitch tracking (rapt)," *Speech coding and synthesis*, vol. 495, p. 518, 1995.

[19] "Voiced/unvoiced detection," http://www.ee.ucla.edu/dsplab/vus/over.html.

[20] R. Amado *et al.*, "Pitch detection algorithms based on zero-cross rate and autocorrelation function for musical notes," in *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on.* IEEE, 2008, pp. 449–454.

[21] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, p. 1917, 2002.

[22] "Cepstrum Example," http://en.wikipedia.org/wiki/Cepstrum.

[23] H. Ney, "A dynamic programming technique for nonlinear smoothing," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'81.*, vol. 6. IEEE, 1981, pp. 62–65.

[24] H. Ney, "Dynamic programming algorithm for optimal estimation of speech parameter contours," *Systems, Man and Cybernetics, IEEE Transactions on*, no. 2, pp. 208–214, 1983.

[25] N. H. de Jong and T. Wempe, "Automatic measurement of speech rate in spoken dutch," *ACLC Working Papers*, vol. 2, no. 2, pp. 49–58, 2007.

[26] I. P. Association, *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet.* Cambridge Univ Pr, 1999.

[27] N. Morgan and E. Fosler-Lussier, "Combining multiple estimators of speaking rate," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 2. IEEE, 1998, pp. 729–732.

[28] "Emotional prosody speech and transcripts database from Linguistic Data Consortium (LDC)," http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2002S28.

[29] "CMU Arctic Database," http://www.festvox.org/cmu_arctic/.

[30] "Harvard Sentences Database in Open Speech Repository," http://www.voiptroubleshooter.com/open_speech/american.html.

[31] "Freesound website for short pieces of music download," http://www.freesound.org/.

[32] A. Varga and H. J. Steeneken, "Assessment for automatic speech recognition: Ii. noisex-92: A database and an experiment to study the effect of additive noise on speech recognition systems," *Speech Communication*, vol. 12, no. 3, pp. 247–251, 1993.

[33] "Wireless Communication and Networking Group," http://www.ece.rochester.edu/projects/wcng/code.html.

[34] "Source code for the YIN algorithm," http://audition.ens.fr/adc/.

[35] "Download page for the SAFE toolkit," http://www.ee.ucla.edu/~weichu/safe/.

[36] "Source code for the Praat algorithm," http://www.fon.hum.uva.nl/praat/.

[37] J. J. Godfrey, E. C. Holliman, and J. McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, vol. 1. IEEE, 1992, pp. 517–520.

[38] "De Jong's Praat Script for speaking rate estimation," https://sites.google.com/site/speechrate/.

[39] K. N. Stevens and G. Von Bismarck, "A nineteen-channel filter bank spectrum analyzer for a speech recognition system," 1967.